

Microsoft.DP-420.vJun-2024.by.Tony.50q

Number: DP-420
Passing Score: 800
Time Limit: 120
File Version: 7.0

Exam Code: DP-420

Exam Name: Designing and Implementing Cloud-Native Applications Using Microsoft Azure Cosmos DB



01 - Litware Azure Subscription

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 Qbutton 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 conproduct container.

QUESTION 1

You need to provide a solution for the Azure Functions notifications following updates to conproduct.

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 leaseCollectionName
- C. Configure the trigger for each function to use a different leaseCollectionName
- D. Configure the trigger for each function to use the same leaseCollectionPrefix

Correct Answer: A, C

Section:

Explanation:

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 conproduct container.

Reference:

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

QUESTION 2

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.

Correct Answer: D

Section:

QUESTION 3

HOTSPOT

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.



Hot Area:

Answer Area

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

con-productVendor: Composite on creditRating and yearFounded
Range on creditRating
Range on yearFounded

Answer Area:

Answer Area

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

con-productVendor: Composite on creditRating and yearFounded
Range on creditRating
Range on yearFounded

Section:

Explanation:

QUESTION 4

HOTSPOT

You need to select the capacity mode and scale configuration for account2 to support the planned changes and meet the business requirements. What should you select? To answer, select the appropriate options in the answer area. NOTE: Each correct selection is worth one point.

Hot Area:

Answer Area

Capacity mode: Provisioned throughput
Provisioned throughput
Serverless

Scale configuration: Autoscale throughput on iotdb and throughput sharing across the containers
Autoscale throughput on con-iot1 and con-iot2
Autoscale throughput on iotdb and throughput sharing across the containers
Manual throughput on con-iot1 and con-iot2

Answer Area:

Answer Area

Capacity mode:

Scale configuration:

Section:

Explanation:

Exam A

QUESTION 1

HOTSPOT

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.

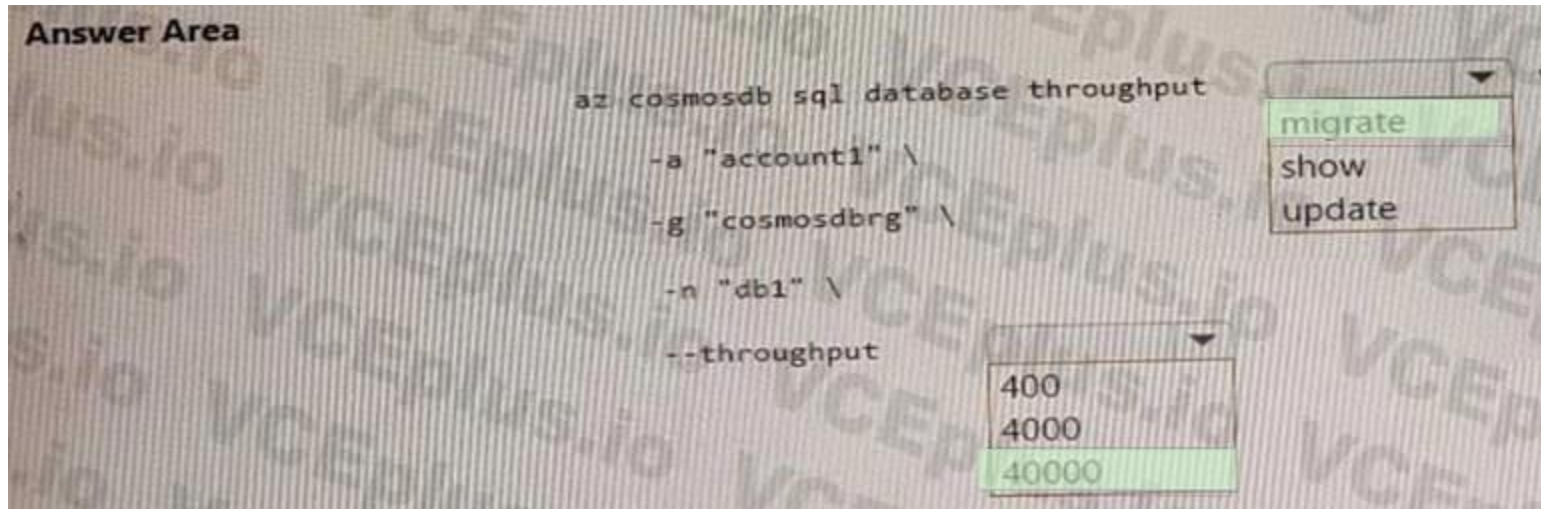


Hot Area:

Answer Area

```
az cosmosdb sql database throughput
```

Answer Area:



Section:

Explanation:

Migrate

40000

According to the Azure CLI reference¹, 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 2

HOTSPOT

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
{
    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;

ClientEncryptionIncludedPath[] 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.

Hot Area:

Statements	Yes	No
You can perform a query that filters on the creditcard property.	<input type="radio"/>	<input type="radio"/>
You can perform a query that filters on the ssn property.	<input type="radio"/>	<input type="radio"/>
An application can be allowed to read the creditcard property while being restricted from reading the ssn property.	<input type="radio"/>	<input type="radio"/>

Answer Area:

Statements	Yes	No
You can perform a query that filters on the creditcard property.	<input type="radio"/>	<input checked="" type="radio"/>
You can perform a query that filters on the ssn property.	<input type="radio"/>	<input checked="" type="radio"/>
An application can be allowed to read the creditcard property while being restricted from reading the ssn property.	<input checked="" type="radio"/>	<input type="radio"/>

Section:

Explanation:

According to the Azure Cosmos DB documentation¹, 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 operations¹.

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 operations¹.

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 property².

QUESTION 3

HOTSPOT

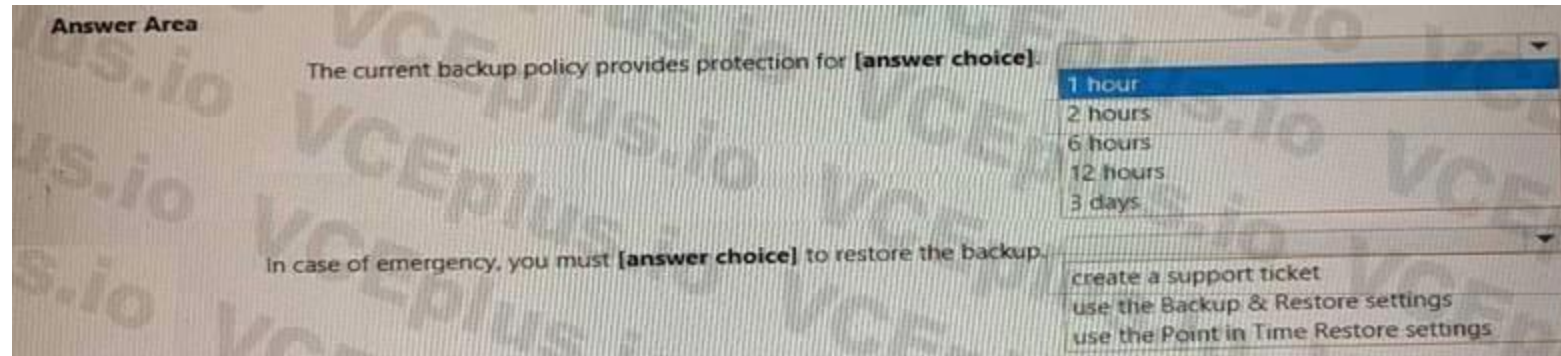
You configure a backup for an Azure Cosmos DB for NoSQL account as shown in the following exhibit.

The screenshot shows the 'Backup & Restore' configuration page for an Azure Cosmos DB account. The 'Backup Interval' is set to 120 minutes. The 'Backup Retention' is set to 12 hours. 'Copies of data retained' is set to 6. Under 'Backup storage redundancy', 'Geo-redundant backup storage' is selected.

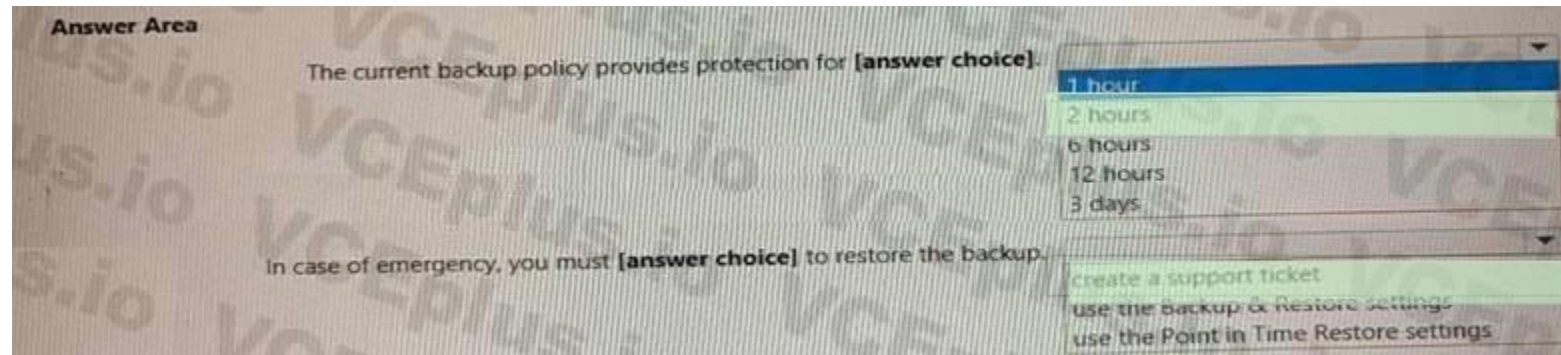
Use the drop-down menus to select the answer choice that completes each statement based on the information presented in the graphic.

NOTE: Each correct selection is worth one point.

Hot Area:



Answer Area:



Section:

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 days¹.

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 days¹.

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 completed².

QUESTION 4

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

Correct Answer: A, B, G

Section:

Explanation:

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

QUESTION 5

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"

Correct Answer: C, D, F

Section:

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"
```

Incorrect Answers:

B: JSON is plain text.

Note, full example:

```
{  
  "name": "cosmosdb-sink-connector",  
  "config": {  
    "connector.class": "com.azure.cosmos.kafka.connect.sink.CosmosDBSinkConnector",  
    "tasks.max": "1",  
    "topics": [  
      "hotels"  
    ],  
    "value.converter": "org.apache.kafka.connect.json.AvroConverter",  
    "value.converter.schemas.enable": "false",  
    "key.converter": "org.apache.kafka.connect.json.AvroConverter",  
    "key.converter.schemas.enable": "false",  
    "connect.cosmos.connection.endpoint": "https://.
```



```
documents.azure.com:443/",
"connect.cosmos.master.key": "",
"connect.cosmos.databasename": "kafkaconnect",
"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 6

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

Correct Answer: C

Section:

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.

Incorrect Answers:

A: Throughput: Set an optional value for the number of RUs you'd like to apply to your CosmosDB collection for each execution of this data flow. Minimum is 400.

B: Write throughput budget: An integer that represents the RUs you want to allocate for this Data Flow write operation, out of the total throughput allocated to the collection.

D: Collection action: Determines whether to recreate the destination collection prior to writing.

None: No action will be done to the collection.

Recreate: The collection will get dropped and recreated

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

QUESTION 7

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

- A. Database Migration Assistant
- B. Azure Data Factory
- C. Azure Migrate

Correct Answer: B

Section:

Explanation:

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.

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 DB3.

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 API4.

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 API5. 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 8

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>

Correct Answer: B, D

Section:

Explanation:

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 cache2.

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

QUESTION 9

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.

Correct Answer: C

Section:

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 complexity1. 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 levels2. Consistent prefix guarantees that reads never see out-of-order writes and that monotonic reads are preserved1. This may be suitable for your application if you can tolerate some eventual consistency.

QUESTION 10

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 rolebased 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

Correct Answer: A

Section:

Explanation:

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

Incorrect Answers:

B: DocumentDB Account Contributor can manage Azure Cosmos DB accounts. Azure Cosmos DB is formerly known as DocumentDB.

C: DocumentDB Account Contributor: Can manage Azure Cosmos DB accounts.

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

QUESTION 11

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"

Correct Answer: A

Section:

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 12

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.

Correct Answer: A

Section:

Explanation:

Reference: <https://docs.microsoft.com/en-us/azure/cosmos-db/provision-account-continuousbackup>

QUESTION 13

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 + ''"`



Correct Answer: C

Section:

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 14

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.

Correct Answer: A, D

Section:

Explanation:

Reference:

<https://github.com/microsoft/MCW-Cosmos-DB-Real-Time-Advanced-Analytics/blob/main/Handson%20lab/HOL%20step-by%20step%20-%20Cosmos%20DB%20realtime%20advanced%20analytics.md>

QUESTION 15

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

Correct Answer: A

Section:

Explanation:

According to the Azure Cosmos DB documentation¹², 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 items². 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:

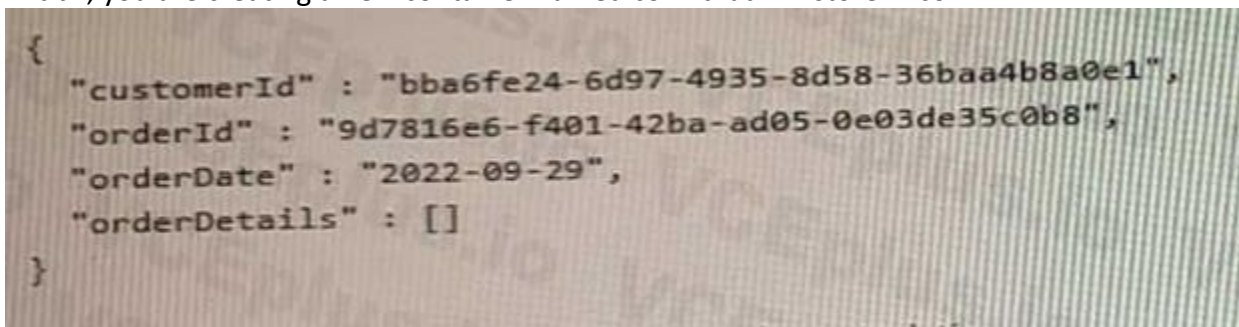
- A. all the properties of the updated items

QUESTION 16

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

You are designing an application that will use db1.

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



```
{
  "customerId" : "bba6fe24-6d97-4935-8d58-36baa4b8a0e1",
  "orderId" : "9d7816e6-f401-42ba-ad05-0e03de35c0b8",
  "orderDate" : "2022-09-29",
  "orderDetails" : []
}
```

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

Correct Answer: C

Section:

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 partitions³. A partition key should have a high cardinality, which means it should have many distinct values, such as hundreds or thousands¹. 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 value¹.

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 ID³.

It aligns with the query patterns of the application, as customers will often view their past order history³.

It minimizes costs, as it reduces the number of cross-partition queries and optimizes the storage and throughput utilization¹.

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 criteria³.

It may result in hot partitions or throttling if some customers create orders more frequently than others or have more data than others¹.

It may not support transactions across multiple customers, as transactions are scoped to a single logical partition².

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 item², or you could use a partition key with a random or pre-calculated suffix to distribute the workload more evenly².

QUESTION 17

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 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.

Correct Answer: B

Section:

Explanation:

Azure Cosmos DB is a multi-model database that supports various data models, such as documents, key-value, graph, and column-family³. 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 way³.

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 ID¹.

It avoids storing redundant data or creating additional containers for users¹.

It allows you to browse by company and browse the users by company with simple queries¹.

It shows a details page for the company and all the related users by fetching a single document¹.

It leverages the benefits of embedding data, such as reducing the number of requests, improving query performance, and simplifying data consistency².

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 MB².

It may not be optimal for scenarios where users need to be associated with more than one company or queried independently from companies².

It may not be scalable for companies that have more than 1,000 users, as it could result in hot partitions or throttling².

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 data², or you could use a graph data model that expresses entities and relationships as vertices and edges.

QUESTION 18

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.

Correct Answer: A

Section:

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 19

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 ConflictResolutionMode 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

Correct Answer: B

Section:

Explanation:

QUESTION 20

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.

```
{
  "indexingMode": "consistent",
  "includedPaths":
  [
    {
      "path": "/product/category/?"
    },
    {
      "path": "/product/brand/?"
    }
  ],
  "excludedPaths":
  [
    {
      "path": "/*"
    },
    {
      "path": "/product/brand"
    }
  ]
}
```

Which path will be indexed?

- A. /product/brand
- B. /product/category
- C. /product/[]/category
- D. /product/brand/tailspin

Correct Answer: A

Section:

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 objects1.

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 name1.

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

QUESTION 21

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. userAssignedIdentatxe
- D. allowedOrxgins

Correct Answer: B

Section:

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 authentication1.

QUESTION 22

HOTSPOT

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.)

```

01 {
02   "connectionStrings": [
03     {
04       "connectionString":
05       "AccountEndpoint=https://contosodbaccount.documents.azure.com:443/;
06       AccountKey=MwUgRnGti4vErT2rFPPFdTFfyI9KyI9Kbe1RPGv70QdHo6VZ2i45TcJzrd4J8OzYxrEATzyZh0M1nJaNFA==;",
07       "description": "Primary SQL Connection String"
08     },
09     {
10       "connectionString":
11       "AccountEndpoint=https://contosodbaccount.documents.azure.com:443/;
12       AccountKey=gfThRnGti4vErT2rFPPFdTFfyI43529Kbe1RPGv70QdHo6VZ2i45TcJzrd4J8OzYxrEATzyZh0M1nJaNFA==;",
13       "description": "Secondary SQL Connection String"
14     },
15     {
16       "connectionString":
17       "AccountEndpoint=https://contosodbaccount.documents.azure.com:443/;
18       AccountKey=W6ykBc1PHJoos6HdErT2rFPPFx9yI9Kbe1RPGv70Q1IQwQNxq6QdOXjxgyLLebXBp8uJu7FyJy3Uv1vuK2A==;",
19       "description": "Primary Read-Only SQL Connection String"
20     },
21     {
22       "connectionString":
23       "AccountEndpoint=https://contosodbaccount.documents.azure.com:443/;
24       AccountKey=k2DZI0oY4Jc7QeUJqVGH3csda6EyI9Kbe1RPGv70QErT2rFPPFtbwTPfKAg19zVxC0NDNn8xPpQrednVYcQ==;",
25       "description": "Secondary Read-Only SQL Connection String"

```



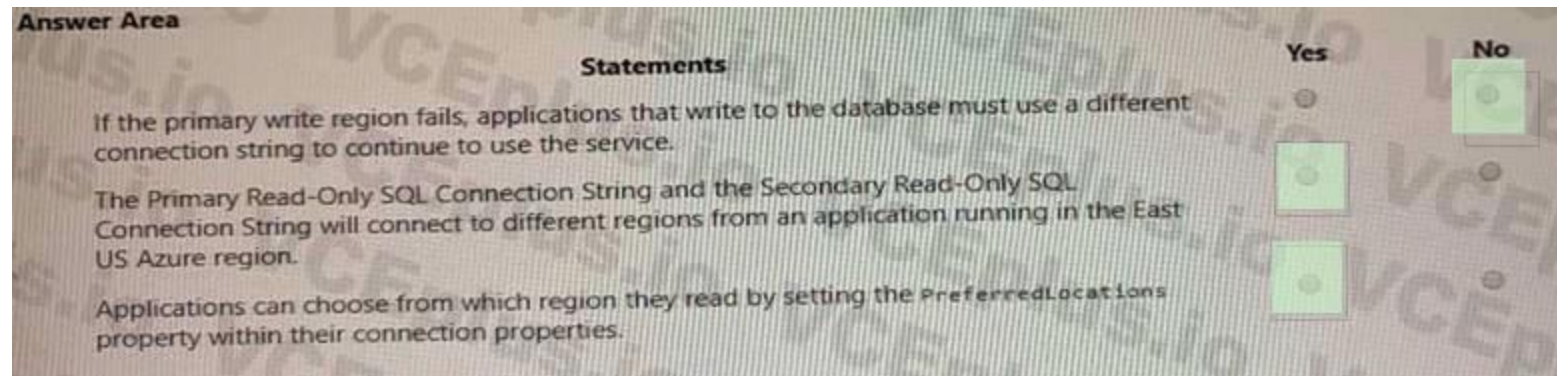
For each of the following statements, select Yes if the statement is true. otherwise, select No.

NOTE: Each correct selection is worth one point.

Hot Area:

Statements	Yes	No
If the primary write region fails, applications that write to the database must use a different connection string to continue to use the service.	<input type="radio"/>	<input checked="" type="radio"/>
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.	<input type="radio"/>	<input type="radio"/>
Applications can choose from which region they read by setting the PreferredLocations property within their connection properties.	<input type="radio"/>	<input type="radio"/>

Answer Area:



Section:

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. = NO

You 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 outage². The application does not need to change the connection string or specify the failover priority³. 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 latency¹.

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 = Yes

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. 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 application². Azure Cosmos DB will route the requests to the appropriate region based on the availability and latency¹. 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 location¹.

QUESTION 23

HOTSPOT

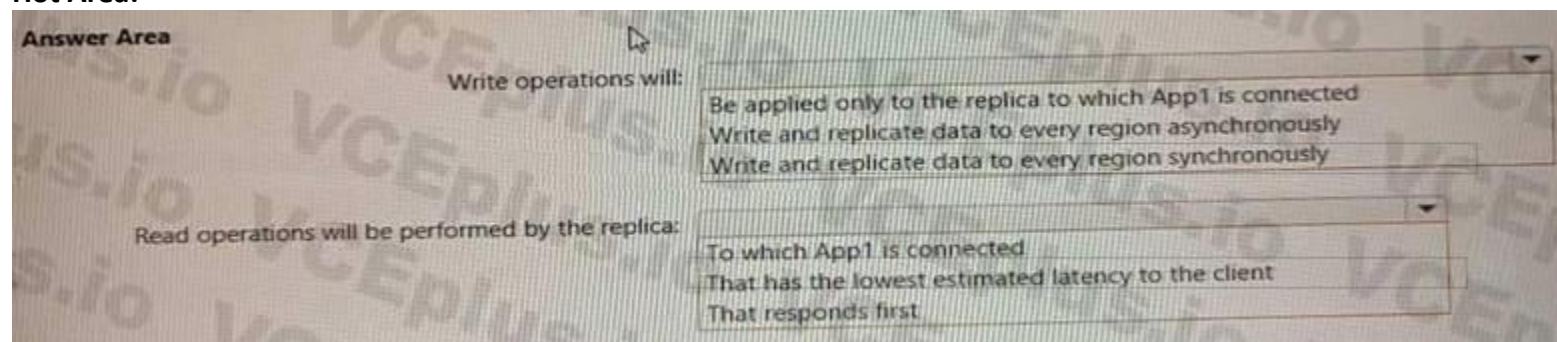
You have a multi-region Azure Cosmos DB account named account1 that has a default consistency level of strong.

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

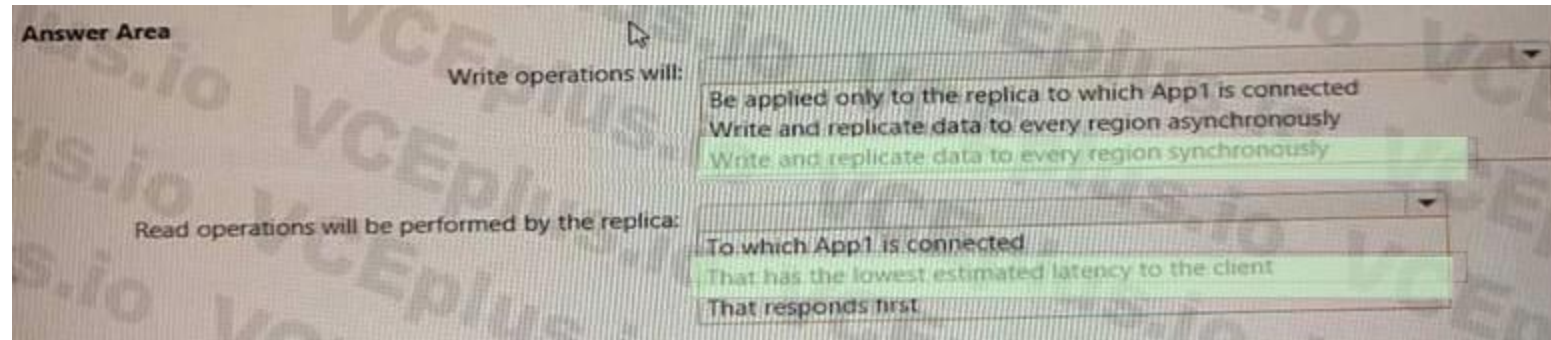
How will the read and write operations of App1 be handled? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Hot Area:



Answer Area:



Section:

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 account2, 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 account1. The request level consistency level of session only applies to the read operations of App11.

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 24

HOTSPOT

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

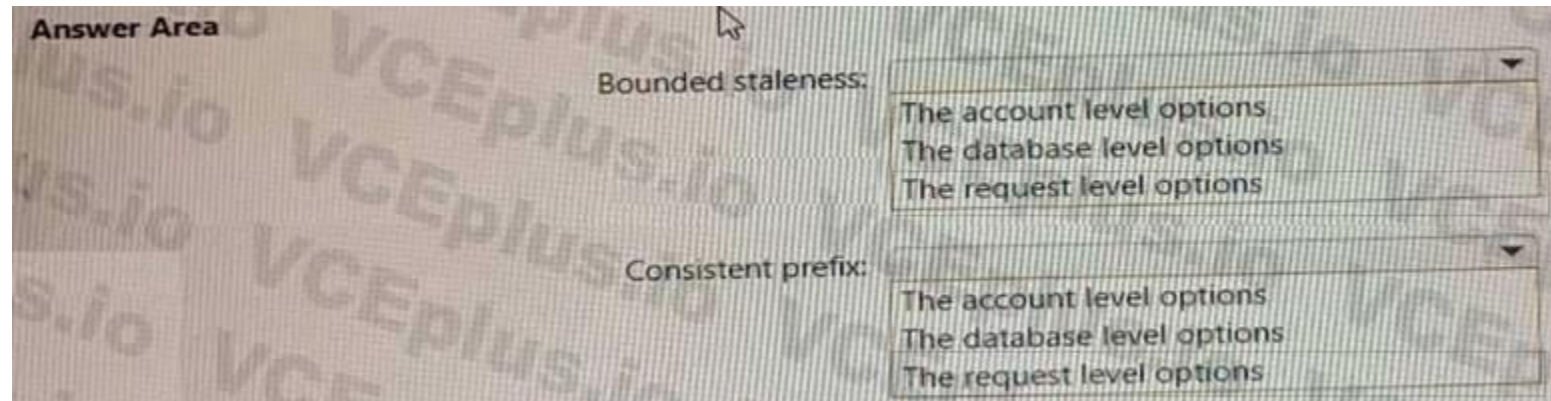
You have an app named App1.

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

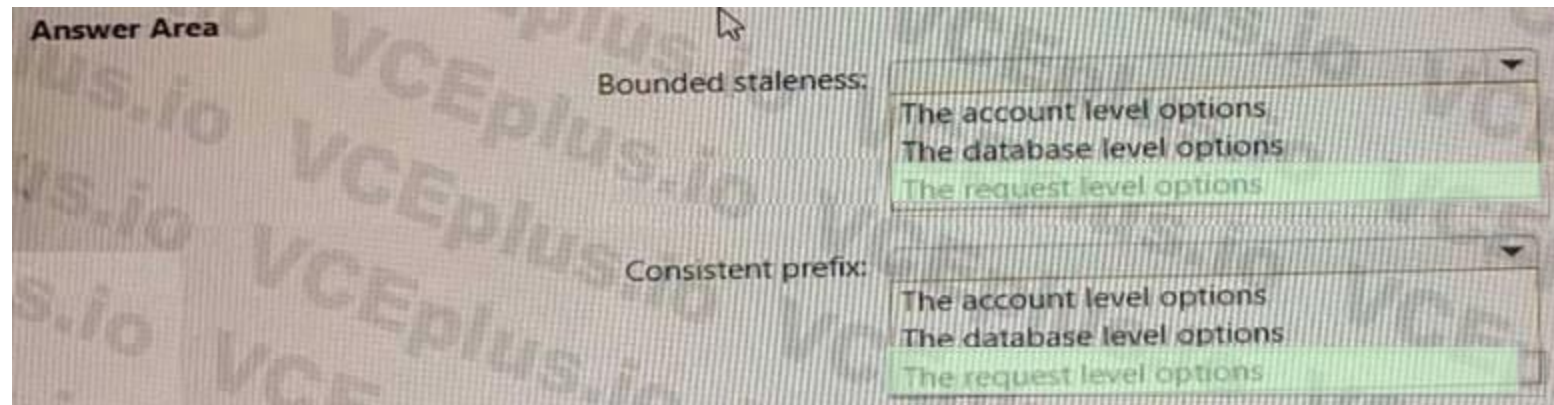
What should you modify for each consistency level? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Hot Area:



Answer Area:



Section:

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 time². The default consistency level applies to all databases and containers under that account¹. You can also override the default consistency level for a specific request by using the request options².

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/enus/azure/cosmos-db/consistency-levels>

QUESTION 25

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 coll2 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 coll1 to Autoscale. Set the throughput for coll2 and coll3 to Manual.
- C. Create a serverless account.

Correct Answer: B

Section:

Explanation:

Azure Cosmos DB offers two different capacity modes: provisioned throughput and serverless¹.

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 consumed¹. 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 data¹.

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 load¹.

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 needs¹.

It allows you to optimize your costs by paying only for the throughput you need for each container¹.

It allows you to enable geo-distribution for your account if you need to replicate your data across multiple regions¹.

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 ratio¹.

It may not be optimal for scenarios where all containers have low or sporadic traffic that does not justify provisioned capacity¹.

It may not support availability zones or multi-master replication for your account¹.

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-distribution¹. 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-distribution¹.

QUESTION 26

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 of 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.

Correct Answer: B

Section:

Explanation:

Azure Cosmos DB is a multi-model database that supports various data models, such as documents, key-value, graph, and column-family³. 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 way³.

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 ID¹.

It avoids storing redundant data or creating additional containers for order items¹.

It allows you to view the order history of a customer with simple queries¹.

It leverages the benefits of embedding data, such as reducing the number of requests, improving query performance, and simplifying data consistency².

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 MB².

It may not be optimal for scenarios where order items need to be queried independently from orders or aggregated by other criteria².

It may not support transactions across multiple orders or customers, as transactions are scoped to a single logical partition².

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 data², or you could use a graph data model that expresses entities and relationships as vertices and edges.

QUESTION 27

HOTSPOT

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.

Hot Area:

Answer Area

```

SELECT * FROM c
WHERE c.city
IN ('Moncton', 'Toronto', 'Montreal')

SELECT * FROM c
WHERE c.city = 'Moncton'
AND c.age > 45
AND c.age < 70

SELECT * FROM c
WHERE c.city = 'Moncton'
AND c.age > 45

```

Composite
Range
Spatial

Composite
Range
Spatial

Composite
Range
Spatial

Answer Area:

Answer Area

```

SELECT * FROM c
WHERE c.city
IN ('Moncton', 'Toronto', 'Montreal')

SELECT * FROM c
WHERE c.city = 'Moncton'
AND c.age > 45
AND c.age < 70

SELECT * FROM c
WHERE c.city = 'Moncton'
AND c.age > 45

```

Composite
Range
Spatial

Composite
Range
Spatial

Composite
Range
Spatial

Section:

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 property¹. Range index also supports any string or number².

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 properties¹. Composite index allows you to specify a list of property paths and sort orders that are used for ordering items².

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 point¹. Spatial index supports point, polygon and linestring data types².

QUESTION 28

HOTSPOT

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"
    }
  ]
}
```

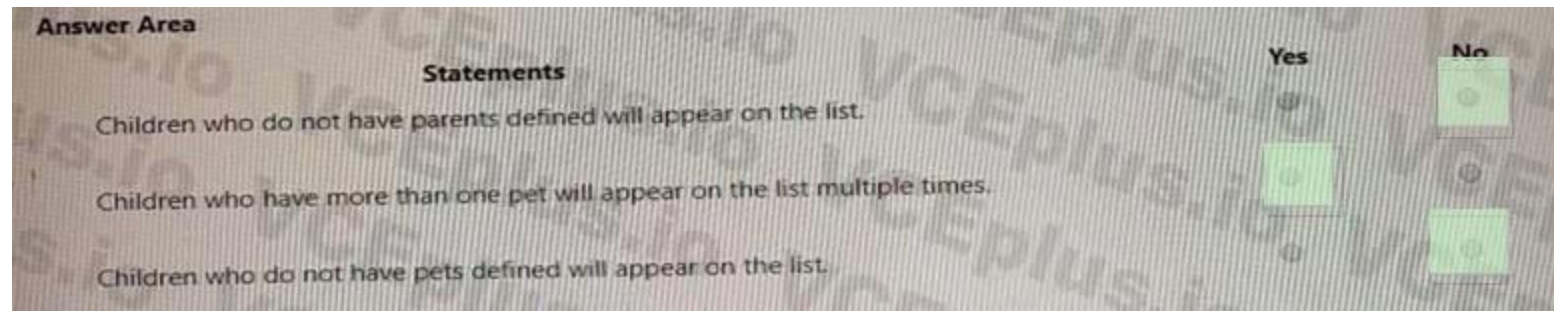


For each of the following statements, select Yes if the statement is true. otherwise, select No.
NOTE: Each correct selection is worth one point.

Hot Area:

Statements	Yes	No
Children who do not have parents defined will appear on the list.	<input type="checkbox"/>	<input type="checkbox"/>
Children who have more than one pet will appear on the list multiple times.	<input type="checkbox"/>	<input type="checkbox"/>
Children who do not have pets defined will appear on the list.	<input type="checkbox"/>	<input type="checkbox"/>

Answer Area:



Section:

Explanation:

Children who do not 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 not 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 29

Your company develops an application named App1 that uses the Azure Cosmos DB SDK and the Eventual consistency level.

App1 queries an Azure Cosmos DB for NoSQL account named account!

You need to identify which consistency level to assign to App1 to meet the following requirements:

- * Maximize the throughput of the queries generated by App1 without increasing the number of request units currently used by the queries.
- * Provide the highest consistency guarantees.

Which consistency level should you identify?

- A. Strong
- B. Bounded Staleness
- C. Session
- D. Consistent Prefix

Correct Answer: A

Section:

QUESTION 30

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 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 that uses the Azure Cosmos DB for NoSQL change feed as a trigger and an Azure event hub as the output.

Does this meet the goal?

- A. Yes

B. No

Correct Answer: A

Section:

QUESTION 31

HOTSPOT

You have an Azure Cosmos DB container named container1.

You need to insert an item into contained. The solution must ensure that the item is deleted automatically after two hours.

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

NOTE: Each correct selection is worth one point.

Hot Area:



Answer Area

```
{
  "id": "1",
  "customerOrderNumber": "XA2342093",
  "orderDate": "2022-04-22T09:34:42",
  "customerNumber": 234223,
  "orderItems":
  [
    {
      "id": 1,
      "quantity": 12,
      "productCode": "82348LK"
    }
  ]
}
"_attachments": "attachments/",
"ttl": 7200,
"defaultTimeToLive": 7200,
"_ttl": 7200,
"_ts": 1551322496
}
```



Answer Area:

Answer Area

```
{
  "id": "1",
  "customerOrderNumber": "XA2342093",
  "orderDate": "2022-04-22T09:34:42",
  "customerNumber": 234223,
  "orderItems":
  [
    {
      "id": 1,
      "quantity": 12,
      "productCode": "82348LK"
    }
  ]
}
"_attachments": "attachments/",
"ttl": 7200,
"defaultTimeToLive": 7200,
"_ttl": 7200,
"_ts": 1551322496
}
```

The logo for Vdumps, featuring a stylized orange 'V' followed by the word 'dumps' in a grey sans-serif font.

Section:

Explanation:

QUESTION 32

You have an Azure Cosmos DB database that contains a container named container 1. The container1 container is configured with a maximum of 20,000 RU/s and currently contains 240 GB of data. You need to estimate the costs of container1 based on the current usage. How many RU/s will be charged?

A. 240

- B. 4,000
- C. 20,000
- D. 24,000

Correct Answer: B

Section:

Explanation:

QUESTION 33

HOTSPOT

You have the Azure Cosmos DB for NoSQL containers shown in the following table.

Name	DefaultTimeToLive
container1	-1
container2	null
container3	60

You have the items shown in the following table.

Name	Container	TimeToLive
item1	container1	60
item2	container2	10
item3	container3	-1

When will each item expire? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Hot Area:



Answer Area

Item1: ▼
Never
10 seconds
60 seconds
10 minutes
60 minutes

Item2: ▼
Never
10 seconds
60 seconds
10 minutes
60 minutes

Item3: ▼
Never
10 seconds
60 seconds
10 minutes
60 minutes

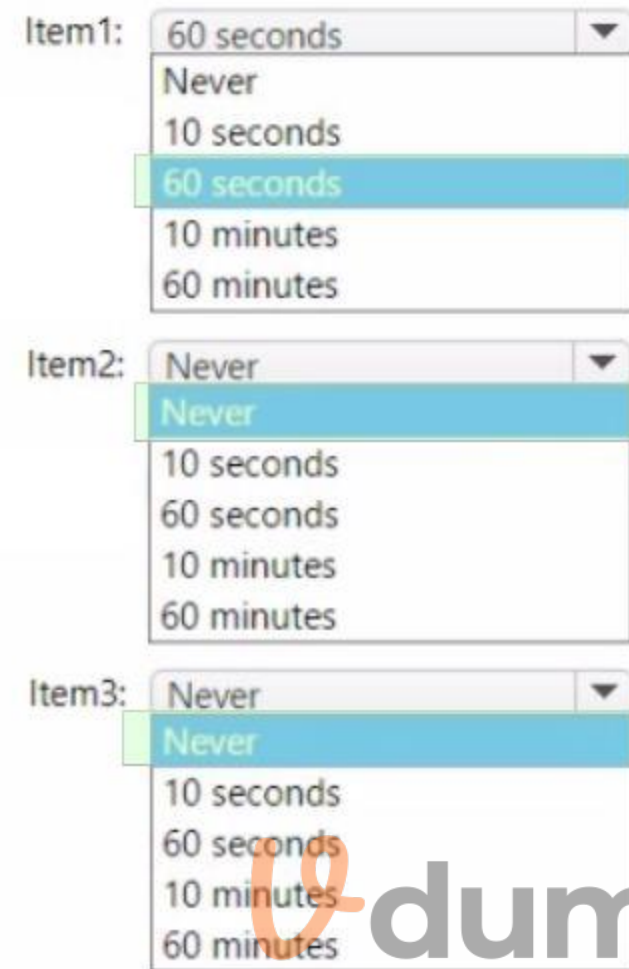
Answer Area:

Answer Area

Item1: 60 seconds
Never
10 seconds
60 seconds
10 minutes
60 minutes

Item2: Never
Never
10 seconds
60 seconds
10 minutes
60 minutes

Item3: Never
Never
10 seconds
60 seconds
10 minutes
60 minutes



Section:

Explanation:

QUESTION 34

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. Provisioned Throughput
- C. Metadata Requests
- D. Data Usage

Correct Answer: C

Section:

Explanation:

QUESTION 35

You have a container named container1 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 rolebased 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 Contributor only
- D. DocumentDB Account Contributor and Cosmos DB Built-in Data Contributor

Correct Answer: A

Section:

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 36

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.

```
Update-AzCosmosDBAccountFailoverPriority -ResourceGroupName "rg1" -Name "account1" -FailoverPolicy @("East US", "West US")
```

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.



Correct Answer: B

Section:

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 order1.

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 regions2. 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 37

HOTSPOT

You have an Azure Cosmos DB for NoSQL account named accounts1.

You plan to implement the integrated cache for account1.

You need to configure the connectivity mode and the consistency level for requests that target account1. The solution must maximize consistency while using the integrated cache.

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

NOTE: Each correct selection is worth one point.

Hot Area:

Answer Area

Connectivity mode:
Direct mode over TCP
Gateway mode with a dedicated gateway
Gateway mode with a standard gateway

Consistency level:
Consistent Prefix
Session
Strong

Answer Area:

Answer Area

Connectivity mode:
Direct mode over TCP
Gateway mode with a dedicated gateway
Gateway mode with a standard gateway

Consistency level:
Consistent Prefix
Session
Strong

Section:

Explanation:

QUESTION 38

HOTSPOT

You have an Azure subscription that contains an Azure Cosmos DB for NoSQL account named account1 and a Log Analytics workspace named Workspace1. Workspace 1 stores the logs of account1.

You need to identify which operations used the most request units per second (RU/s) during the last 24 hours.

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

NOTE: Each correct selection is worth one point

Hot Area:



Answer Area

AzureDiagnostics
AzureActivity
AzureDiagnostics
AzureMetrics

```
| where ResourceProvider=="MICROSOFT.DOCUMENTDB" and Category==  
| where TimeGenerated >= ago(1d)
```

"DataPlaneRequests"
"ControlPlaneRequests"
"DataPlaneRequests"
"PartitionKeyStatistics"

```
| summarize max(responseLength_s), max(requestLength_s), max(requestCharge_s), count = count() by OperationName, requestResourceType_s, userAgent_s, collectionRid_s
```

Answer Area:

Answer Area

AzureDiagnostics
AzureActivity
AzureDiagnostics
AzureMetrics

```
| where ResourceProvider=="MICROSOFT.DOCUMENTDB" and Category==  
| where TimeGenerated >= ago(1d)
```

"DataPlaneRequests"
"ControlPlaneRequests"
"DataPlaneRequests"
"PartitionKeyStatistics"

```
| summarize max(responseLength_s), max(requestLength_s), max(requestCharge_s), count = count() by OperationName, requestResourceType_s, userAgent_s, collectionRid_s
```

Section:

Explanation:

QUESTION 39

TSPOT

You have an Azure subscription that contains an Azure Cosmos DB for NoSQL database named DB1. The shared throughput provisioned for DB1 is 10,000 DTU/s. DB1 contains the containers shown in the following table.

Name	Provisioned throughput
Container1	Share throughput across containers
Container2	4,000 DTU/s

You need to modify the throughput for the containers. The solution must meet the following requirements:

- * The maximum throughput for Container1 must be 4,000 DTU/s.
- * The throughput for Contained must be shared across the containers.
- * Administrative effort must be minimized.

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

Hot Area:

Answer Area

For Container1:
Migrate the data in Container1 to a new container.
Modify the settings of Container1.
Modify the settings of Container2.
Modify the settings of DB1.

For Container2:
Migrate the data in Container2 to a new container.
Modify the settings of Container1.
Modify the settings of Container2.
Modify the settings of DB1.

Answer Area:

Answer Area

For Container1:
Migrate the data in Container1 to a new container.
Modify the settings of Container1.
Modify the settings of Container2.
Modify the settings of DB1.

For Container2:
Migrate the data in Container2 to a new container.
Modify the settings of Container1.
Modify the settings of Container2.
Modify the settings of DB1.

Section:

Explanation:

QUESTION 40

You have an Azure subscription.

You plan to create an Azure Cosmos DB for NoSQL database named DB1 that will store author and book data for authors that have each published up to ten books. Typical and frequent queries of the data will include:

- * All books written by an individual author
- * The synopsis of individual books

You need to recommend a data model for DB1. The solution must meet the following requirements:

- * Support transactional updates of the author and book data.
- * Minimize read operation costs.

What should you recommend?

- A. Create a single container that stores author items and book items, and then items that represent the relationship between the authors and their books.
- B. Create three containers, one that stores author items, a second that stores book items, and a third that stores items that represent the relationship between the authors and their books.
- C. Create two containers, one that stores author items and another that stores book items. Embed a list of each author's books in the corresponding author item.
- D. Create a single container that stores author items and book items. Embed a list of each author's books in the

Correct Answer: D

Section:

QUESTION 41

You provision an Azure Cosmos DB for NoSQL container. You set the throughput to Autoscale, and the maximum request units per second (RU/s) to 20,000. For how many RU/s will you be charged when the actual RU/s usage is zero?

- A. 0
- B. 200
- C. 2,000
- D. 4,000
- E. 10,000

Correct Answer: C

Section:

QUESTION 42

You have operational data in an Azure Cosmos DB 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 to 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

Correct Answer: A

Section:



QUESTION 43

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 transaction.

What should you use to group the items?

- A. logical partitions
- B. physical partitions
- C. databases
- D. containers

Correct Answer: A

Section:

QUESTION 44

HOTSPOT

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.

Hot Area:

Answer Area

The screenshot shows three dropdown menus for consistency levels:

- Reporting:** Eventual (selected)
- Purchasing:** Bounded staleness (selected)
- Fulfillment:** Strong (selected)

Answer Area:



Section:

Explanation:

QUESTION 45

HOTSPOT

You have an Azure Cosmos DB for NoSQL account that contains a container named container1.

You plan to use container1 as a key-value store that will perform point reads on the item ID and partition key.

You need to define an indexing policy for container1. The solution must meet the following requirements:

- * Provide the ability to set DefaultTimeToLive for container1.
- * Minimize implementation time.
- * Minimize costs.

How should you complete the JSON definition of the indexing policy? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Hot Area:



Answer Area

```
{
  "indexingMode": "none",
  "automatic": false,
  "includedPaths": [
    [
      [{"path": "^_etag\/*"}],
      [{"path": "^_ts\/*"}]
    ]
  ],
  "excludedPaths": [{"path": "/"}]
}
```

Answer Area:

Answer Area

```
{
  "indexingMode": "none",
  "automatic": false,
  "includedPaths": [
    [
      [{"path": "^_etag\/*"}],
      [{"path": "^_ts\/*"}]
    ]
  ],
  "excludedPaths": [{"path": "/"}]
}
```

Section:

Explanation:

QUESTION 46

HOTSPOT

You have the following Azure Resource Manager (ARM) template.


```

{
  "type": "Microsoft.DocumentDB/databaseAccount",
  "apiVersion": "2022-08-15",
  "name": "acct01/mydb/mycontainer",
  "properties": {
    "resource": {
      "id": "mycontainer",
      "partitionKey": {
        "paths": [
          "/companyid"
        ],
        "kind": "Hash"
      },
      "indexingPolicy": {
        "indexingMode": "consistent",
        "includedPaths": [
          {
            "path": "/*"
          },
          {
            "path": "/headquarters/country/?"
          }
        ],
        "excludedPaths": [
          {
            "path": "/headquarters/*"
          }
        ],
        "compositeIndexes": [
          [
            {
              "path": "/name",
              "order": "ascending"
            },
            {
              "path": "/startDate",
              "order": "descending"
            }
          ]
        ]
      }
    }
  }
}

```



You plan to deploy the template in incremental mode.

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

Hot Area:

Answer Area

Statements	Yes	No
When the template is deployed, an Azure Cosmos DB account named <code>acct01</code> will be created if the account does NOT exist.	<input type="radio"/>	<input type="radio"/>
When the template is deployed, a container named <code>mycontainer</code> in <code>mydb</code> will be created if the container does NOT exist.	<input type="radio"/>	<input type="radio"/>
When the template is deployed, if <code>mycontainer</code> exists and has a partition key on <code>name</code> , the partition key will change to <code>companyId</code> .	<input type="radio"/>	<input type="radio"/>

Answer Area:

Answer Area

Statements	Yes	No
When the template is deployed, an Azure Cosmos DB account named <code>acct01</code> will be created if the account does NOT exist.	<input type="radio"/>	<input checked="" type="radio"/>
When the template is deployed, a container named <code>mycontainer</code> in <code>mydb</code> will be created if the container does NOT exist.	<input checked="" type="radio"/>	<input type="radio"/>
When the template is deployed, if <code>mycontainer</code> exists and has a partition key on <code>name</code> , the partition key will change to <code>companyId</code> .	<input checked="" type="radio"/>	<input type="radio"/>

Section:

Explanation: