DP-420 Q&As

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

Pass Microsoft DP-420 Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

https://www.geekcert.com/dp-420.html

100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by Microsoft Official Exam Center

- **Instant Download** After Purchase
- **100% Money Back** Guarantee
- **365 Days** Free Update
- **800,000+** Satisfied Customers

QUESTION 1

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.

Correct Answer: AEF

QUESTION 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

Correct Answer: A

QUESTION 3

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: BD

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 workload1. The integrated cache can reduce the RU consumption and latency of read operations by serving them from

the cache instead of the backend containers2.

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 4

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: CDF

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 5

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}]

Correct Answer: A

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