

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.pass2lead.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 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: D

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 2

HOTSPOT

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.

Hot Area:

Answer Area

Statements	Yes	No
The query performs a cross-partition query	<input type="radio"/>	<input type="radio"/>
The query uses an index	<input type="radio"/>	<input type="radio"/>
Recreating the container with the partition key set to /categoryId will improve the performance of the query	<input type="radio"/>	<input type="radio"/>

Correct Answer:

Answer Area

Statements	Yes	No
The query performs a cross-partition query	<input type="radio"/>	<input checked="" type="radio"/>
The query uses an index	<input type="radio"/>	<input checked="" type="radio"/>
Recreating the container with the partition key set to /categoryId will improve the performance of the query	<input checked="" type="radio"/>	<input type="radio"/>

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 3

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

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 4

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`.

Correct Answer: C

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 5

HOTSPOT

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:

1.

A function named Fn1 that reads the change feed of cn1

2.

A function named Fn2 that reads the change feed of cn2

3.

A function named Fn3 that reads the change feed of cn3 You perform the following actions:

1.

Delete an item named item1 from cn1.

2.

Update an item named item2 in cn2.

3.

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.

Hot Area:

Answer Area

Statements	Yes	No
Fn1 will receive item1 from the change feed	<input type="radio"/>	<input type="radio"/>
Fn2 can check the <code>_etag</code> of item2 to see whether the item is an update or an insert	<input type="radio"/>	<input type="radio"/>
Fn3 will receive item3 from the change feed	<input type="radio"/>	<input type="radio"/>

Correct Answer:

Answer Area

Statements	Yes	No
Fn1 will receive item1 from the change feed	<input type="radio"/>	<input checked="" type="radio"/>
Fn2 can check the <code>_etag</code> of item2 to see whether the item is an update or an insert	<input type="radio"/>	<input checked="" type="radio"/>
Fn3 will receive item3 from the change feed	<input checked="" type="radio"/>	<input type="radio"/>

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>

[DP-420 VCE Dumps](#)

[DP-420 Study Guide](#)

[DP-420 Braindumps](#)