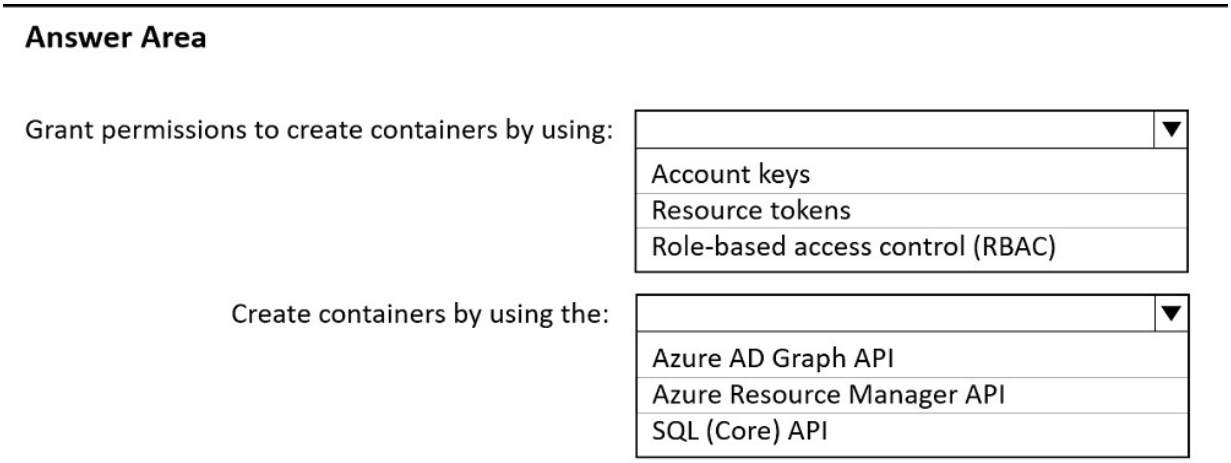
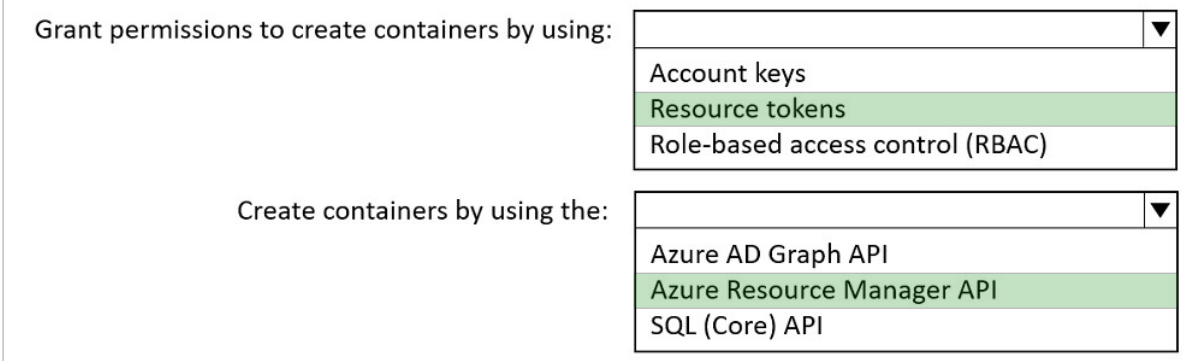
Question #1 *Topic 1*

HOTSPOT -
You have an Azure Cosmos DB Core (SQL) API account named account1 that has the disableKeyBasedMetadataWriteAccess property enabled.
You are developing an app named App1 that will be used by a user named DevUser1 to create containers in account1. DevUser1 has a non-privileged user account in the Azure Active Directory (Azure AD) tenant.
You need to ensure that DevUser1 can use App1 to create containers in account1.
What should you do? To answer, select the appropriate options in the answer area.
NOTE: Each correct selection is worth one point.
Hot Area:



Correct Answer:



Box 1: Resource tokens -
Resource tokens provide access to the application resources within a database. Resource tokens:
Provide access to specific containers, partition keys, documents, attachments, stored procedures, triggers, and UDFs.
Box 2: Azure Resource Manager API
You can use Azure Resource Manager to help deploy and manage your Azure Cosmos DB accounts, databases, and containers.
Incorrect Answers:
The Microsoft Graph API is a RESTful web API that enables you to access Microsoft Cloud service resources.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/secure-access-to-data https://docs.microsoft.com/en-us/rest/api/resources/

lakime Highly Voted 1 year ago
I think it will be "Role-based access control" as Resouce Token doesn't cooperate with AD, regarding second part - ARM is correct
upvoted 15 times

nkav Highly Voted 11 months ago

RBAC is the answer.

upvoted 5 times

essdeecee Most Recent 8 months, 1 week ago

More likely to be SQL (Core) API. Permission for Cosmos is required whereas Azure Resource Manager would need portal permissions.
https://docs.microsoft.com/en-us/dotnet/api/microsoft.azure.cosmos.database.createcontainerifnotexistsasync?view=azure-dotnet
https://github.com/MicrosoftLearning/dp-420-cosmos-db-dev/blob/main/instructions/06-sdk-crud.md#:~:text=Asynchronously%20invoke%20the%20CreateContainerIfNotExistsAsync,CreateContainerIfNotExistsAsync(%22products%22%2C%20%22/categoryId%22%2C%20400)%3B

upvoted 2 times

HOTSPOT -

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

You run the following Azure CLI script.

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

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

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

NOTE: Each correct selection is worth one point.

Hot Area:

Answer Area

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

Correct Answer:

Answer Area

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

Box 1: Yes -

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

Box 2: No -

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

To Configure multi-region set UseMultipleWriteLocations to true.

Box 3: Yes -

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

Reference:

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

nqthien041292 Highly Voted 1 year ago

Vote YNN

upvoted 12 times

wispa2001 Highly Voted 1 year ago

1:No
2:Yes
3:No

upvoted 10 times

Ranzzzan 2 months, 3 weeks ago

Should be YNY. for the third one tried it on pricing calculator, the cost does not change.

upvoted 1 times

ognamala 9 months, 1 week ago

Your first answer is incorrect - in the above, there is a change of the priority zero, effectively changing the write region.

upvoted 1 times

ognamala 9 months, 1 week ago

Your second answer is also incorrect - again the priority 0 is changing which initiates manual failover - so that should be No.

upvoted 1 times

kel7718 Most Recent 5 months, 1 week ago

A failover priority of 0 indicates a write region (https://learn.microsoft.com/en-us/rest/api/cosmos-db-resource-provider/2021-04-01-preview/database-accounts/failover-priority-change?tabs=HTTP)

Knowing this, North Europe is now 0 so write region, first answer is thus Yes. West Europe is now 1 and we did not enable multiple write regions so we can no longer write to west europe and costs haven't increased as it is still a single write region. YNN

upvoted 1 times

Alex22022 3 months, 3 weeks ago

Costs have increased since North Europe is added. Now we have two regions instead of one. Each region is billed: https://learn.microsoft.com/en-us/training/modules/configure-replication-manage-failovers-azure-cosmos-db/4-evaluate-cost-distributing-data-globally

But since the statment is 'The cost ... is unaffected ...', the answer is no.
Therefore: YNN

upvoted 1 times

codingdown 6 months ago

bah , not clear at all

upvoted 2 times

ognamala 9 months, 1 week ago

Yes - triggering a manual failover by changing priority zero causes write and read regions to be swapped
No - triggering a manual failover by changing priority zero causes write and read regions to be swapped
Yes - unaffected only because we are switching between west and north europe, if it was say switzerland it would be more expensive

upvoted 4 times

SQLK 9 months, 2 weeks ago

Why option C is yes??

upvoted 1 times

Alex22022 3 months, 3 weeks ago

C is No because each region is billed additionally. Equivalent emxample from the docs: https://learn.microsoft.com/en-us/training/modules/configure-replication-manage-failovers-azure-cosmos-db/4-evaluate-cost-distributing-data-globally

upvoted 1 times

susejzepol 5 months, 3 weeks ago

I think because they are un the same región. This cause that cost doesn't increase.

upvoted 1 times

Gtus 11 months, 1 week ago

The third statement here is tricky. Normally, Azure resource price is different for regions, this applicable to Cosmos DB as well. However, the price of West Euro and North Euro for Costome DB is just exactly the same. You may check it with Azure Pricing Calculator: https://azure.microsoft.com/en-us/pricing/calculator/

upvoted 4 times

lakime 1 year ago

IMO C should be No

upvoted 3 times

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

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

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

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

```
{
   "name": "apple",
   "type": ["fruit", "exotic"]
   "orders": 10000
}
```

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

NOTE: Each correct selection is worth one point.

A.

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

B.

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

C.

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

D.

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

Correct Answer: *BD*

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

Incorrect Answers:

A: Default sorting ordering is Ascending. Must use Descending order.

C: Order on Orders not on Type.

Reference:

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

virgilpza 2 months, 3 weeks ago

answer is correct

upvoted 1 times

itsmenida1 4 months, 4 weeks ago

correct

upvoted 3 times

HOTSPOT -

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

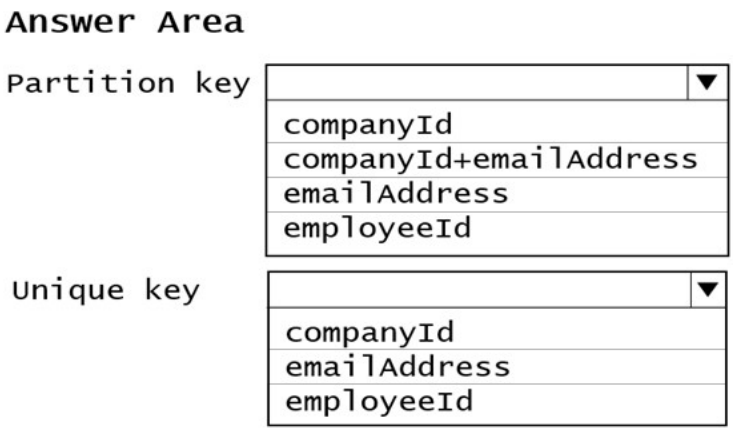
You plan to create a container that will store employee data for 5,000 small businesses. Each business will have up to 25 employees. Each employee item will have an emailAddress value.

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

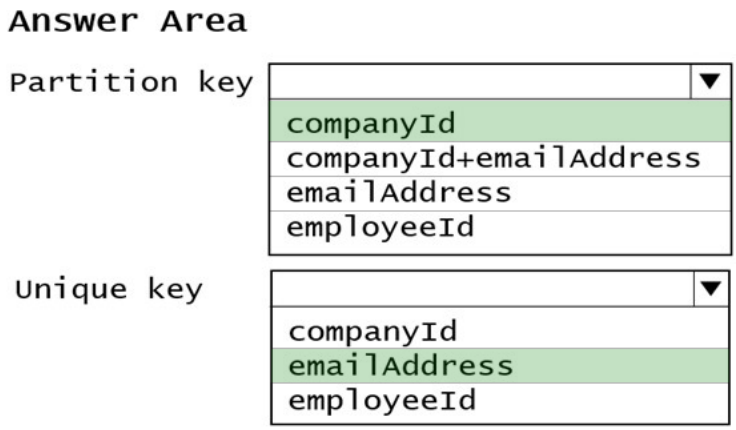
To what should you set the partition key and the unique key? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Hot Area:



Correct Answer:



Box 1: CompanyID -

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

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

Box 2: emailAddress -

Reference:

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

ognamala Highly Voted 9 months, 1 week ago

Solution is correct - companyId for partition key, and emailAddress for unique key

Additional justification why you would definitely set the partition key to companyId, is by also keeping potential queries in mind, typically in this scenario you will always end up with an equality filter on the companyId, hence it makes sense to partition by companyId, which will result in better query performance

UniqueKey will need to be set to emailAddress, since the job of a unique key is to ensure uniqueness of a value within a logical partition
https://docs.microsoft.com/en-us/azure/cosmos-db/unique-keys

upvoted 5 times

lakime Most Recent 1 year ago

Correct - Logical partitions are formed based on the value of a partition key that is associated with each item in a container. All the items in a logical partition have the same partition key value.
So segregating by companyid makes sense
E-mail will be unique

upvoted 2 times

HOTSPOT -

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

The following is a sample of a document in container1.

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

The orderId property is used as the partition key.

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

NOTE: Each correct selection is worth one point.

Hot Area:

Answer Area

Statements	Yes	No

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

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

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

Correct Answer:

Answer Area

Statements	Yes	No

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

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

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

Box 1: Yes -

Records with different OrderIDs will match.

Box 2: Yes -

Records with different OrderIDs will match.

Box 3: No -

Only records with one specific OrderId will match

muks1003 Highly Voted 11 months, 1 week ago

The given answers are correct, OrderId is the partition key and any query not involving the OrderId will be considered as Cross Partition (Or Fan-out) Query.

upvoted 11 times

Torent2005 Most Recent 7 months, 4 weeks ago

NNY

Cross-pratition query means that your query hits only neccessary data from logical partition based on partition key (query has filter eg. c.orderId). If not scan whole data. IF "orderId" is partition key then first two queries are not cross-partition queries.

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

upvoted 1 times

codingdown 6 months ago

noooope at all.
cross partition means it will scan more than 1 partition, easy as that

upvoted 3 times

lakime 1 year ago

As orderid is partition key, And if we will assume that this orderid will appear in different dates and in different customers - then those answers are correct

upvoted 4 times

Question #6 *Topic 1*

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

The following is a sample of the data.

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

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

- Minimizes the partition skew
- Avoids capacity limits
- Avoids hot partitions

What should you do?

A. Use timestamp as the partition key.

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

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

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

Correct Answer: *D*

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

Incorrect Answers:

A: You will also not like to partition the data on λ€DateTimeλ€, because this will create a hot partition. Imagine you have partitioned the data on time, then for a given minute, all the calls will hit one partition. If you need to retrieve the data for a customer, then it will be a fan-out query because data may be distributed on all the partitions.

B: Senser1Value has only two values.

C: All the devices could have the same manufacturer.

Reference:

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

Community vote distribution

D (80%) C (20%)

muks1003 Highly Voted 11 months, 1 week ago

Selected Answer: D

D is the correct answer, I have taken the Microsoft Official Test And the answer is D.

upvoted 7 times

Gall Most Recent 8 months, 3 weeks ago

Selected Answer: D

Because of frequency (1s) we may not want to put too much data into the same container. The maxim quota for logical partition is 20GB, which can exhaust pretty fast.

upvoted 1 times

ognamala 9 months, 1 week ago

Correct answer is D as per this link https://docs.microsoft.com/en-us/azure/cosmos-db/sql/synthetic-partition-keys - in order to avoid a hot partition we have no option but to use a strategy like that suggested in the section "Use a partition key with a random suffix"

upvoted 1 times

chukismit 10 months, 1 week ago

D, it's the right answer because deviceid a with pre-calculated suffixes number (Second when data is inserted) It will create 60 partitions by every deviceid

upvoted 2 times

jfarrell 11 months, 3 weeks ago

I get the reason it is D BUT, there is context missing that would be necessary. For example, does this mean that the devicid for like devices is the same and somehow hardware bound? Also, the suggestion of a random number is not inclusive of how random the number is, obviously if its generated with each send you will create excessive partitions

upvoted 1 times

lakime 1 year ago

Selected Answer: C

Fully agree, I would either select C, or have an answer - deviceid

upvoted 2 times

lakime 1 year ago

Sorry - I'm changing my mind ;) I guess B makes more sense in that case, we got two options (True/False) - and creating synthetic key when probably devideid will have always same manufactor - doesn't make sense

upvoted 1 times

ognamala 9 months, 1 week ago

C and B are both wrong, the both can end up creating a hot partition because of the low cardinality

upvoted 1 times

kdsingh 1 year ago

D, can't be the right answer because deviceid and a random number will create a new partition on every document you will write to the database.

upvoted 2 times

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

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

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

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

What should you include in the solution?

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

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

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

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

Correct Answer: *A*

Store multiple entity types in the same container.

Community vote distribution

C (62%) A (38%)

jfarrell Highly Voted 11 months, 3 weeks ago

This is nonsense. Microsoft;s own documentation clearly states that for groups of data with like PKs (and it would be author here) use the same container. C is the answer

upvoted 13 times

imando Most Recent 1 month, 3 weeks ago

Selected Answer: C

C is correct answer

upvoted 2 times

Ranzzzan 2 months, 3 weeks ago

A

https://learn.microsoft.com/en-us/azure/cosmos-db/nosql/modeling-data#hybrid-data-models

upvoted 1 times

NickKazen 7 months ago

Selected Answer: A

Although it would minimise read operation costs, there is no limit on the number of books authors can write (unfortunately in some cases). Because of that, the size of the actual document can grow enormously. That on its own will cause a huge factor of latency.

https://learn.microsoft.com/en-us/azure/cosmos-db/nosql/modeling-data#when-not-to-embed

upvoted 1 times

codingdown 6 months ago

Then what?
If the number of books becomes very high, retrieving all the books per Author would require a high effort in both scenario (A or C) . Having the Author document embedded in the same container would not cause substantial overhead, (only extra operation to do in C would be to filter out the Author document based on property "type" AFTER retrieving all books (N)/ and author (1) documents having same partition key.

upvoted 1 times

DirectX 6 months, 4 weeks ago

"there is no limit on the number of books authors can write (unfortunately in some cases)".
I think that this is not the case in real life. The ration is 1:few

upvoted 1 times

remz 9 months, 1 week ago

Selected Answer: C

C is the most read cost effective

upvoted 2 times

remz 9 months, 1 week ago

i read it back , A is the Right answer!

upvoted 1 times

ognamala 9 months, 1 week ago

Selected Answer: C

We are explicitly required to minimise cost and latency, A although it would get the job done it would be highly inefficient - the relationship between an author and books is a one to few (1-many but bounded), hence data should be embedded - refer to the below links

https://docs.microsoft.com/en-us/azure/cosmos-db/sql/modeling-data
https://docs.microsoft.com/en-us/azure/cosmos-db/sql/how-to-model-partition-example

upvoted 4 times

grada 10 months, 1 week ago

Selected Answer: A

A is the closest of what I'd model, with a little addon of both books and authors containing collections of not only the authorIds and bookIds (respectively), but also a subset of data from the documents they're referencing (for instance, authorId + authorName, and bookId + bookTitle).

https://docs.microsoft.com/en-us/azure/cosmos-db/sql/modeling-data#hybrid-data-models

upvoted 4 times

essdeecee 8 months, 1 week ago

Perfect. A link to the exact example in the documentation for this exact question. Not clear from the question but logically it [u]is[/u] a N:N relationship.

upvoted 2 times

jfarrell 11 months, 3 weeks ago

Sorry I mean't B with a type field

upvoted 1 times

shachar_ash 11 months, 2 weeks ago

In C you put them in the same container, and embedding to lower the latency

upvoted 2 times

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

Juba1711 5 months, 3 weeks ago

correct

https://learn.microsoft.com/en-us/azure/cosmos-db/nosql/query/is-number

upvoted 2 times

Question #9 *Topic 1*

You need to implement a trigger in Azure Cosmos DB Core (SQL) API that will run before an item is inserted into a container.
Which two actions should you perform to ensure that the trigger runs? Each correct answer presents part of the solution.
NOTE: Each correct selection is worth one point.

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

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

C. Register the trigger as a pre-trigger.

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

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

Correct Answer: *C*
C: When triggers are registered, you can specify the operations that it can run with.
F: When executing, pre-triggers are passed in the RequestOptions object by specifying PreTriggerInclude and then passing the name of the trigger in a List object.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/sql/how-to-use-stored-procedures-triggers-udfs

kdsingh Highly Voted 1 year ago
C and E are correct answers.
upvoted 14 times

SQLK Most Recent 9 months, 3 weeks ago
Is there an option F here
upvoted 3 times

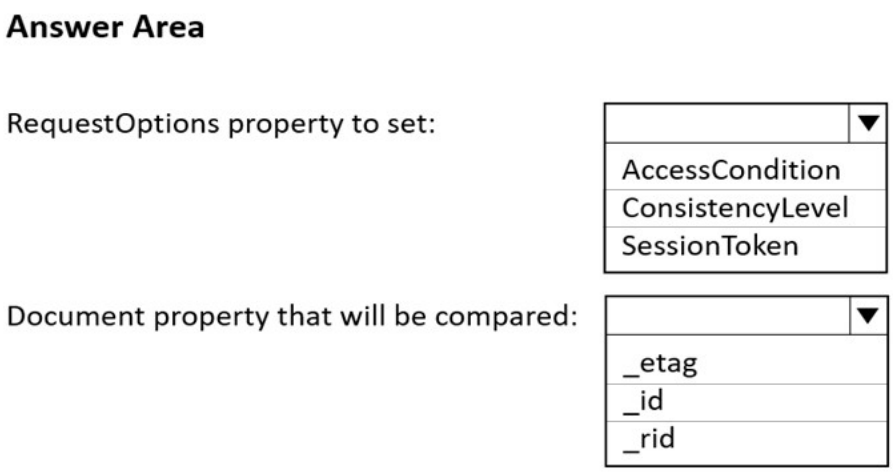
susejzepol 7 months ago
I think that they refer to E when they are mentioning F option.
upvoted 1 times

Aunehwet79 8 months, 3 weeks ago
Supposed to say E
upvoted 1 times

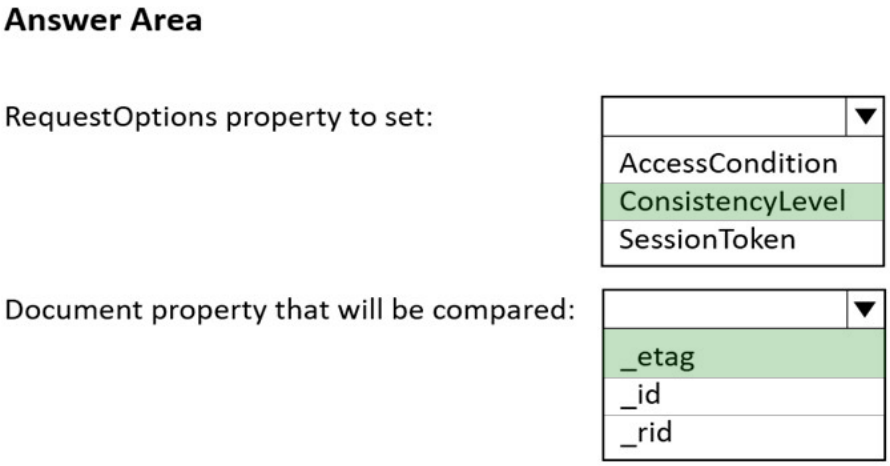
prim 9 months, 3 weeks ago
C and E, in exam
upvoted 3 times

jfarrell 11 months, 3 weeks ago
There is no F option
upvoted 4 times

HOTSPOT -
You have a container in an Azure Cosmos DB Core (SQL) API account.
You need to use the Azure Cosmos DB SDK to replace a document by using optimistic concurrency.
What should you include in the code? To answer, select the appropriate options in the answer area.
NOTE: Each correct selection is worth one point.
Hot Area:



Correct Answer:



Box 1: ConsistencyLevel -
The ItemRequestOptions Class ConsistencyLevel property gets or sets the consistency level required for the request in the Azure Cosmos DB service.
Azure Cosmos DB offers 5 different consistency levels. Strong, Bounded Staleness, Session, Consistent Prefix and Eventual - in order of strongest to weakest consistency.

Box 2: _etag -
The ItemRequestOptions class helped us implement optimistic concurrency by specifying that we wanted the SDK to use the If-Match header to allow the server to decide whether a resource should be updated. The If-Match value is the ETag value to be checked against. If the ETag value matches the server ETag value, the resource is updated.
Reference:
https://docs.microsoft.com/en-us/dotnet/api/microsoft.azure.cosmos.itemrequestoptions https://cosmosdb.github.io/labs/dotnet/labs/10-concurrency-control.html

AT96 Highly Voted 1 year ago
When we then want to send our request to replace a document, we can specify an AccessCondition with the ETag we received when we fetched out our document.
upvoted 16 times

grada 10 months, 1 week ago

This, it's supposed to be either AccessCondition or IfMatchETag directly, consistency levels have nothing to do with optimistic concurrency.

https://docs.microsoft.com/en-us/dotnet/api/microsoft.azure.documents.client.requestoptions.accesscondition?view=azure-dotnet#examples

upvoted 2 times

remz Most Recent 9 months, 1 week ago

Accesscondition ETAG

upvoted 1 times

HOTSPOT -

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

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

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

The application has the following characteristics:

- Users can submit an unlimited number of posts.
- The average number of posts submitted by a user will be more than 1,000.
- Posts can have an unlimited number of comments from different users.
- The average number of comments per post will be 100, but many posts will exceed 1,000 comments.
- Users will be limited to having a maximum of 20 interests.

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

NOTE: Each correct selection is worth one point.

Hot Area:

Answer Area

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

Correct Answer:

Answer Area

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

Box 1: Yes -
Non-relational data increases write costs, but can decrease read costs.

Box 2: Yes -
Non-relational data increases write costs, but can decrease read costs.

Box 3: No -
Non-relational data increases write costs, but can decrease read costs.

avijitd Highly Voted 10 months, 2 weeks ago
correct
upvoted 6 times

codingdown Most Recent 6 months ago
i don't see why the write cost should increase, the read cost does
upvoted 1 times

susejzepol 6 months ago
i think could be because the operation need to do and upsert instead of an insert.
upvoted 3 times

muks1003 11 months, 1 week ago
Correct Answer Yes, Yes, No
upvoted 2 times

shachar_ash 11 months, 2 weeks ago
Correct
upvoted 1 times

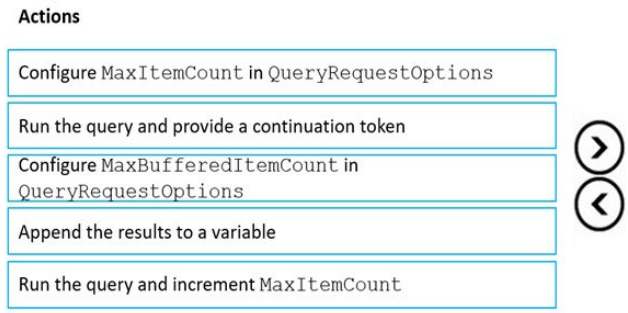
DRAG DROP -

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

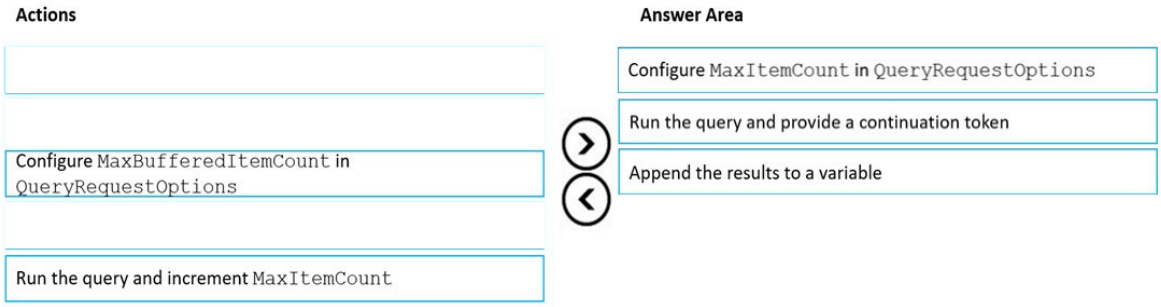
You need to return a complete result set to the app by using pagination. Each page of results must return 80 items.

Which three actions should you perform in sequence? To answer, move the appropriate actions from the list of actions to the answer area and arrange them in the correct order.

Select and Place:



Correct Answer:



Step 1: Configure the MaxItemCount in QueryRequestOptions

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

Box 2: Run the query and provide a continuation token

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

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

Step 3: Append the results to a variable

Reference:

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

remz Highly Voted 9 months, 1 week ago

Correct

upvoted 5 times

purplefish Most Recent 3 weeks, 4 days ago

Correct. Yes, you should configure the MaxItemCount property to limit the number of items returned per page to 80. Additionally, you should use the continuation token returned in the response to retrieve the next page of results.

upvoted 1 times

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have an Azure Cosmos DB Core (SQL) API account named account 1 that uses autoscale throughput.
You need to run an Azure function when the normalized request units per second for a container in account1 exceeds a specific value.
Solution: You configure an Azure Monitor alert to trigger the function.
Does this meet the goal?

A. Yes

B. No

Correct Answer: *A*
You can set up alerts from the Azure Cosmos DB pane or the Azure Monitor service in the Azure portal.
Note: Alerts are used to set up recurring tests to monitor the availability and responsiveness of your Azure Cosmos DB resources. Alerts can send you a notification in the form of an email, or execute an Azure Function when one of your metrics reaches the threshold or if a specific event is logged in the activity log.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/create-alerts

Community vote distribution

A (100%)

purplefish 3 weeks, 4 days ago

Selected Answer: A

An Azure Monitor alert can be configured to trigger a function when certain conditions are met, such as the normalized request units per seconds.

upvoted 1 times

Amokrane 4 months, 2 weeks ago

Selected Answer: A

upvoted 1 times

ognamala 9 months, 1 week ago

Selected Answer: A

Agreed

upvoted 4 times

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have an Azure Cosmos DB Core (SQL) API account named account 1 that uses autoscale throughput.
You need to run an Azure function when the normalized request units per second for a container in account1 exceeds a specific value.
Solution: You configure the function to have an Azure CosmosDB trigger.
Does this meet the goal?

A. Yes

B. No

Correct Answer: *B*
Instead configure an Azure Monitor alert to trigger the function.
You can set up alerts from the Azure Cosmos DB pane or the Azure Monitor service in the Azure portal.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/create-alerts

Community vote distribution

B (100%)

imando 1 month, 3 weeks ago

Selected Answer: B

No Azure Monitor is correct option. B is the correct answer.

upvoted 2 times

nitohime 5 months, 2 weeks ago

B!!!!!!!!!!!!!!!

upvoted 1 times

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have an Azure Cosmos DB Core (SQL) API account named account 1 that uses autoscale throughput.
You need to run an Azure function when the normalized request units per second for a container in account1 exceeds a specific value.
Solution: You configure an application to use the change feed processor to read the change feed and you configure the application to trigger the function.
Does this meet the goal?

A. Yes

B. No

Correct Answer: *B*
Instead configure an Azure Monitor alert to trigger the function.
You can set up alerts from the Azure Cosmos DB pane or the Azure Monitor service in the Azure portal.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/create-alerts

Community vote distribution

B (100%)

imando 1 month, 3 weeks ago

Selected Answer: B

B is correct answer

upvoted 2 times

You have a database named db1 in an Azure Cosmos DB Core (SQL) API account.
You are designing an application that will use db1.
In db1, you are creating a new container named coll1 that will store online orders.
The following is a sample of a document that will be stored in coll1.

```
{
    "customerId" : "bba6fe24-6d97-4935-8d58-36baa4b8a0e1",
    "orderId" : "9d7816e6-f401-42ba-ad05-0e03de35c0b8",
    "orderDate" : "2021-09-29",
    "orderDetails" : []
}
```

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. orderId

B. customerId

C. orderDate

D. id

Correct Answer: *B*
If most of your workload's requests are queries and most of your queries have an equality filter on the same property, this property can be a good partition key choice. For example, if you frequently run a query that filters on UserID, then selecting UserID as the partition key would reduce the number of cross-partition queries.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/partitioning-overview

TRUESON 1 month ago
all orders from one customer will be stored on the same partition which will be beneficial for reads
upvoted 1 times

You have a container in an Azure Cosmos DB Core (SQL) API account that stores data about orders.
The following is a sample of an order document.

```
{
    "orderId" : "d4a9179b-5ead-43a3-b851-add9a71ac4b6",
    "customerId" : "f6e39103-bdc7-4346-9cfb-45daa4b2becf",
    "orderDate" : "2021-09-29",
    "orderItems" : [...],
    "total" : 12345
}
```

Documents are up to 2 KB.
You plan to receive one million orders daily.
Customers will frequently view their past order history.
You are the evaluating whether to use orderDate as the partition key.
What are two effects of using orderDate as the partition key? Each correct answer presents a complete solution.
NOTE: Each correct selection is worth one point.

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

B. You will exceed the maximum storage per partition

C. There will always be a hot partition

D. Queries will run cross-partition

Correct Answer: *CD*
Not C: But the problem is that λ€" when the application writes new data, the writes will always be directed to the same partition, based on whatever day it is. This results in what's called a hot partition, where we have a bottleneck that's going to quickly consume a great deal more of the reserved throughput you've provisioned for the container. Specifically, Cosmos DB evenly distributes your provisioned throughput across all the physical partitions in the container.
D: Customers will frequently view their past order history. OrderID will be used. Queries will run cross-partition
Incorrect:
Not B: 1 million a day x 2KB -> 2 GB of data/day while maximum storage across all items per (logical) partition: 20 GB
Reference:
https://www.tallan.com/blog/2019/04/30/horizontal-partitioning-in-azure-cosmos-db https://docs.microsoft.com/en-us/azure/cosmos-db/concepts-limits

Community vote distribution

CD (100%)

purplefish 3 weeks, 4 days ago

Selected Answer: CD
Correct

upvoted 1 times

Ranzzzan 2 months, 3 weeks ago

correct.
Not A: There is no limit to the total number of physical partitions in your container. As your provisioned throughput or data size grows, Azure Cosmos DB will automatically create new physical partitions by splitting existing ones
Not B: as max is 20 GB.
So C and D correct answer

upvoted 1 times

Question #18 *Topic 1*

You have a container in an Azure Cosmos DB Core (SQL) API account. The container stores data about families. Data about parents, children, and pets are stored as separate documents.
Each document contains the address of each family. Members of the same family share the same partition key named familyId.
You need to update the address for each member of the same family that share the same address. The solution must meet the following requirements:
⇨ Be atomic, consistent, isolated, and durable (ACID).
⇨ Provide the lowest latency.
What should you do?

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

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

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

Correct Answer: *C*
Each transaction provides ACID (Atomicity, Consistency, Isolation, Durability) property guarantees.
Transactional batch operations offer reduced latency on equivalent operations.
Note: Transactional batch describes a group of point operations that need to either succeed or fail together with the same partition key in a container. In the .NET
SDK, the TransactionalBatch class is used to define this batch of operations. If all operations succeed in the order they are described within the transactional batch operation, the transaction will be committed. However, if any operation fails, the entire transaction is rolled back.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/sql/transactional-batch

Community vote distribution

C (100%)

imando 1 month, 3 weeks ago

Selected Answer: C

Keyword is transactional

upvoted 2 times

Question #19 *Topic 1*

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

The following is a sample of the data.

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

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

- Minimizes the partition skew
- Avoids capacity limits
- Avoids hot partitions

What should you do?

A. Create a new synthetic key that contains deviceId and timestamp

B. Use timestamp as the partition key

C. Use deviceManufacturer as the partition key

D. Use sensor1Value as the partition key

Correct Answer: *A*

Concatenate multiple properties of an item.

You can form a partition key by concatenating multiple property values into a single artificial partitionKey property. These keys are referred to as synthetic keys.

For example, consider the following example document:

{

"deviceId": "abc-123",

"date": 2018

}

For the previous document, one option is to set /deviceId or /date as the partition key. Use this option, if you want to partition your container based on either device

ID or date. Another option is to concatenate these two values into a synthetic partitionKey property that's used as the partition key.

{

"deviceId": "abc-123",

"date": 2018,

"partitionKey": "abc-123-2018"

}

Incorrect:

Not B: But the problem is that λ€" when the application writes new data, the writes will always be directed to the same partition, based on whatever day it is. This results in what's called a hot partition, where we have a bottleneck that's going to quickly consume a great deal more of the reserved throughput you've provisioned for the container. Specifically, Cosmos DB evenly distributes your provisioned throughput across all the physical partitions in the container.

Reference:

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

Community vote distribution

A (100%)

imando 1 month, 3 weeks ago

Selected Answer: A

Correct answer

upvoted 2 times

Question #20 *Topic 1*

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 the company ID as the partition key.

B. 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 document.

C. 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.

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

Correct Answer: *D*

All employees within a company would nicely fit within a single document (document size 2 MB).

Note: An Azure Cosmos container is the unit of scalability both for provisioned throughput and storage. A container is horizontally partitioned and then replicated across multiple regions. The items that you add to the container are automatically grouped into logical partitions, which are distributed across physical partitions, based on the partition key.

Reference:

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

Community vote distribution

D (67%) B (33%)

TRUESON 1 month ago

A is correct … We want users and companies on the same partition. This is done by using company id as partition key for both. As a compayid is a widely used filter thes data that are related can be covered from the same partition. we add a type pro to distinguish the two types

upvoted 2 times

imando 1 month, 3 weeks ago

Selected Answer: D

D is correct answer. Googled other sites and they are showing D as answer.

upvoted 2 times

arnabdt 1 month, 3 weeks ago

Selected Answer: B

B. 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 document.

This design separates the users and companies into two different containers, with each container optimized for querying their respective entities. The user container uses the user ID as the partition key, which allows for efficient retrieval of individual users. The company container uses the company ID as the partition key, which enables efficient retrieval of all users associated with a particular company.

Adding the company ID to each user document in the user container allows for easy and fast querying of users by company. Whenever a company or user profile is selected, the details page for the company and all related users can be retrieved by joining the data from the two containers using the company ID.

This design meets all the requirements specified in the prompt, including efficient reading of data, browsing by company, and browsing users by company.

upvoted 1 times

You are building an application that will store data in an Azure Cosmos DB Core (SQL) API account. The account uses the session default consistency level. The account is used by five other applications. The account has a single read-write region and 10 additional read region.
Approximately 20 percent of the items stored in the account are updated hourly.
Several users will access the new application from multiple devices.
You need to ensure that the users see the same item values consistently when they browse from the different devices. The solution must NOT affect the other applications.
Which two actions should you perform? Each correct answer presents part of the solution.
NOTE: Each correct selection is worth one point.

A. Set the default consistency level to eventual

B. Associate a session token to the device

C. Use implicit session management when performing read requests

D. Provide a stored session token when performing read requests

E. Associate a session token to the user account

Correct Answer: *DE*
Use Session consistency and use Stateful Entities in Durable Functions or something similar that will allow you to implement a distributed mutex to store and update the Session token across multiple Cosmos client instances.
Note: The session consistency is the default consistency that you get while configuring the cosmos DB account. This level of consistency honors the client session. It ensures a strong consistency for an application session with the same session token. What that means is that whatever is written by a session will return the latest version for reads as well, from that same session.
Incorrect:
Not A: Eventual consistency is the weakest consistency level of all. The first thing to consider in this model is that there is no guarantee on the order of the data and also no guarantee of how long the data can take to replicate. As the name suggests, the reads are consistent, but eventually.
This model offers high availability and low latency along with the highest throughput of all. This model suits the application that does not require any ordering guarantee. The best usage of this type of model would be the count of retweets, likes, non-threaded comments where the count is more important than any other information.
Not B: The reads should be the same from different devices.
Reference:
https://stackoverflow.com/questions/64084499/can-i-use-a-client-constructed-session-token-for-cosmosdb

Community vote distribution
BD (50%) BC (50%)

purplefish 3 weeks, 4 days ago

Selected Answer: BD
By associating a session token to the device and providing a stored session token when performing read requests, the application ensures that the user sees the same item values consistently when browsing from different devices.
upvoted 1 times

arnabdt 1 month, 3 weeks ago

Selected Answer: BC
B. Associate a session token to the device: When a user logs in to the application from a device, a session token should be generated and associated with the device. This will ensure that subsequent requests from the same device are routed to the same backend server that served the first request, and the user will see consistent item values.

C. Use implicit session management when performing read requests: When performing read requests, the application should use implicit session management. This means that the SDK will automatically manage the session token, and the user will continue to see consistent item values regardless of which backend server serves the request.
upvoted 1 times

arnabdt 1 month, 3 weeks ago

It talk about different devices so I think I will go with D & E.
upvoted 4 times

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,
        "pets":[

        ],
        "firstname":"Dana",
        "age":15,
        "gender":"female"
      },
      {
        "grade":7,
        "pets":[
          {
            "name":"Bob",
            "type":"guinea pig"
          },
          {
```

You run the following query against the container.

```
SELECT
    ch.name ?? ch.firstName AS childName,
    f.parents
    ARRAY_LENGTH(ch.pets) ?? 0 AS numberOfPets,
    ch.pets
FROM c AS f
JOIN ch IN f.children
```

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
Children who do not have parents defined will appear on the list	◯	◯
Children who have more than one pet will appear on the list multiple times	◯	◯
Children who do not have pets defined will appear on the list	◯	◯

Correct Answer:

Answer Area

Statements	Yes	No
Children who do not have parents defined will appear on the list	○	◉
Children who have more than one pet will appear on the list multiple times	○	◉
Children who do not have pets defined will appear on the list	◉	○

Box 1: No -
The result is empty, since the cross product of each item from source and an empty set is empty:
Note: Joins result in a complete cross product of the sets participating in the join. The result of an N-way join is a set of N-element tuples, where each value in the tuple is associated with the aliased set participating in the join and can be accessed by referencing that alias in other clauses.

Box 2: No -
The cross join is not on the pets.

Box 3: Yes -
The cross join is not on the pets.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/sql/sql-query-join

YJC Highly Voted 6 months, 2 weeks ago
After I tested with JSON, the answer is Yes, No, Yes
upvoted 11 times

ferpin 2 months ago
You are right!
upvoted 1 times

susejzepol 5 months, 3 weeks ago
You are right! In the first question the query return a value because "c" is represent the entire document and not only the parent's array.
upvoted 3 times

basiltomato Most Recent 2 months, 3 weeks ago
Also tested this.

There's missing comma and should be ch.firstname instead of firstName:
SELECT
ch.name ?? ch.firstname AS childName, f.lastName, f.parents, ARRAY_LENGTH(ch.pets) ?? 0 AS numberOfPets,
ch.pets
FROM mktest1 AS f
JOIN ch IN f.children

Answer is YES, NO, YES
upvoted 1 times

Question #23 *Topic 1*

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have an Azure Cosmos DB Core (SQL) API account named account1 that uses autoscale throughput.
You need to run an Azure function when the normalized request units per second for a container in account1 exceeds a specific value.
Solution: You configure Azure Event Grid to send events to the function by using an Event Grid trigger in the function.
Does this meet the goal?

A. Yes

B. No

Correct Answer: *B*
Instead configure an Azure Monitor alert to trigger the function.
You can set up alerts from the Azure Cosmos DB pane or the Azure Monitor service in the Azure portal.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/create-alerts

Ranzzzan 2 months, 3 weeks ago
No, Use Azure monitor alert to trigger function using action group
upvoted 2 times

Question #24 *Topic 1*

You have an application named App1 that reads the data in an Azure Cosmos DB Core (SQL) API account. App1 runs the same read queries every minute. The default consistency level for the account is set to eventual.
You discover that every query consumes request units (RUs) instead of using the cache.
You verify the IntegratedCacheiteItemHitRate metric and the IntegratedCacheQueryHitRate metric. Both metrics have values of 0.
You verify that the dedicated gateway cluster is provisioned and used in the connection string.
You need to ensure that App1 uses the Azure Cosmos DB integrated cache.
What should you configure?

A. the indexing policy of the Azure Cosmos DB container

B. the consistency level of the requests from App1

C. the connectivity mode of the App1 CosmosClient

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

Correct Answer: *C*
Because the integrated cache is specific to your Azure Cosmos DB account and requires significant CPU and memory, it requires a dedicated gateway node.
Connect to Azure Cosmos DB using gateway mode.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/integrated-cache-faq

Community vote distribution
C (100%)

lakime Highly Voted 1 year ago

Selected Answer: C
C is correct
https://docs.microsoft.com/en-us/azure/cosmos-db/how-to-configure-integrated-cache
If you're using the .NET or Java SDK, set the connection mode to gateway mode. This step isn't necessary for the Python and Node.js SDKs since they don't have additional options of connecting besides gateway mo

upvoted 8 times

HOTSPOT -

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

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

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

NOTE: Each correct selection is worth one point.

Hot Area:

Answer Area

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

Correct Answer:

Answer Area

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

Box 1: No -

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

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

Box 2: No -

Only an SMS action will be taken.

Emailreceivers is empty so no email action is taken.

Box 3: Yes -

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

Reference:

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

Ritesh073 Highly Voted 1 year ago

Answer should be No,No,No. Answer description is correct

upvoted 17 times

ferpin Most Recent 2 months ago

No, No, No. The third one is when a new access key is generated not when an item with a new partition key value is created.

upvoted 1 times

ognamala 9 months, 1 week ago

The last one should also be no - the trigger is fired when the database account has a new key regenerated not when an item with a new partition key value is created

upvoted 4 times

Question #26 *Topic 1*

You plan to store order data in an Azure Cosmos DB Core (SQL) API 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.
What should you do?

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

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

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

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

Correct Answer: *D*
By denormalizing data, your application may need to issue fewer queries and updates to complete common operations.
Typically denormalized data models provide better read performance.
Note: In general, use embedded data models when:
There are contained relationships between entities.
There are one-to-few relationships between entities.
There's embedded data that changes infrequently.
There's embedded data that will not grow without bound.
There's embedded data that is queried frequently together.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/sql/modeling-data

Community vote distribution
B (100%)

arnabdt 1 month, 3 weeks ago

Selected Answer: B
B. Create a single database that contains a container for orders and a container for order items.
Creating a single database with two containers, one for orders and another for order items, is the best approach to support order read operations while minimizing the number of requests. This approach provides a clear separation of concerns between orders and order items, making it easier to manage the data and perform read operations.
With this approach, queries that require only order data can be directed to the orders container, while queries that require order item data can be directed to the order items container. This reduces the number of requests required to retrieve the data needed for each query.
upvoted 1 times

Question #27 *Topic 1*

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 uses a partition key of deviceId.

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

Correct Answer: *C*
The analytics application would frequently read from the same partition of Device ID is used as partition key.
Note: The partition key determines how Azure Cosmos DB routes data in partitions. The IoT Device ID is the usual partition key for IoT applications.
Reference:
https://docs.microsoft.com/en-us/azure/architecture/solution-ideas/articles/iot-using-cosmos-db

Zaytsev 1 month ago
Correct
upvoted 1 times

Question #1 *Topic 2*

DRAG DROP -

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

The following is a sample of a document in container1:

```
{
"customerId": 1234,
"firstName": "John",
"lastName": "Smith",
"policyYear": 2021
}
```

The following is a sample of a document in container2:

```
{
"gpsId": 1234,
"latitude": 38.8951,
"longitude": -77.0364
}
```

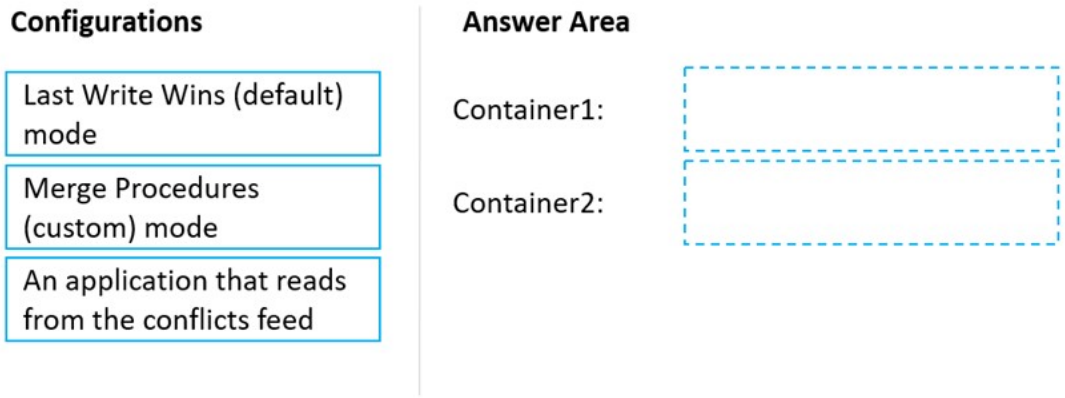
You need to configure conflict resolution to meet the following requirements:

- For container1 you must resolve conflicts by using the highest value for policyYear.
- For container2 you must resolve conflicts by accepting the distance closest to latitude: 40.730610 and longitude: -73.935242.
- Administrative effort must be minimized to implement the solution.

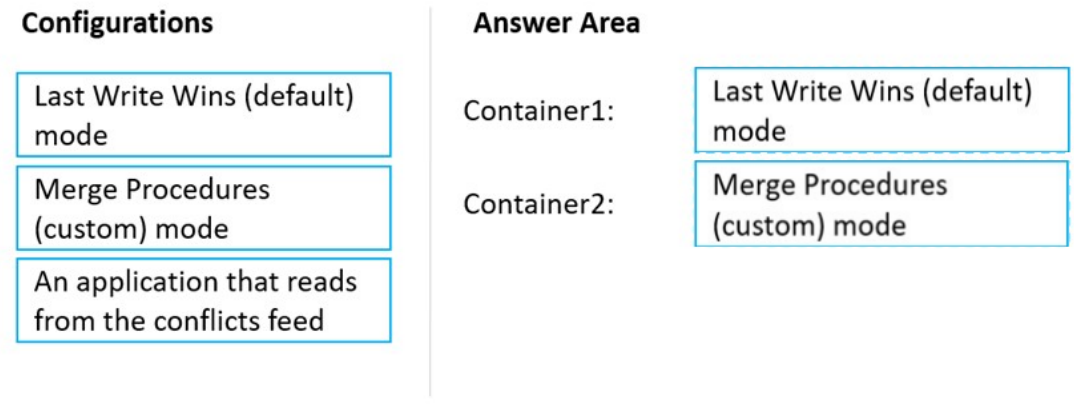
What should you configure for each container? To answer, drag the appropriate configurations to the correct containers. Each configuration may be used once, more than once, or not at all. You may need to drag the split bar between panes or scroll to view content.

NOTE: Each correct selection is worth one point.

Select and Place:



Correct Answer:



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

Last Write Wins (LWW): This resolution policy, by default, uses a system-defined timestamp property. It's based on the time-synchronization clock protocol.
Box 2: Merge Procedures (custom) mode
Custom: This resolution policy is designed for application-defined semantics for reconciliation of conflicts. When you set this policy on your Azure Cosmos container, you also need to register a merge stored procedure. This procedure is automatically invoked when conflicts are detected under a database transaction on the server. The system provides exactly once guarantee for the execution of a merge procedure as part of the commitment protocol.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/conflict-resolution-policies https://docs.microsoft.com/en-us/azure/cosmos-db/sql/how-to-manage-conflicts

YJC 6 months, 2 weeks ago
For the highest value wins, there is one description relate to LWW.
For API for NoSQL, this may also be set to a user-defined path with a numeric type. In a conflict, the highest value wins.
https://learn.microsoft.com/en-us/azure/cosmos-db/nosql/how-to-manage-conflicts?tabs=dotnetv2%2Capi-async%2Casync
upvoted 4 times

Shaunp 7 months ago
what is the correct order Procedure and applicant?
upvoted 1 times

TimSss 7 months, 3 weeks ago
it is explicitly stated that LWW used the DEFAULT (timestamp) mode, so increasing policyYear will not be used by LWW
upvoted 3 times

DrC 9 months, 2 weeks ago
I don't see how the default (last one to write wins) would handle "the highest value for policyYear" if there was a conflict. Both should be procedures.
upvoted 2 times

ognamala 9 months, 1 week ago
Both answers are correct, remember that a LWW policy can be configured to use any numeric path, not just the timestamp, so it can be configured with the /policyYear path where the highest number will win and it would satisfy the requirement
upvoted 2 times

Torent2005 8 months ago
Not really, default mode for LWW is based on timestamp.
"Last Write Wins (LWW): This resolution policy, by default, uses a system-defined timestamp property. It's based on the time-synchronization clock protocol. If you use the SQL API, you can specify any other custom numerical property (e.g., your own notion of a timestamp) to be used for conflict resolution. A custom numerical property is also referred to as the conflict resolution path."
https://learn.microsoft.com/en-us/azure/cosmos-db/conflict-resolution-policies
For Merge Procedures (custom) mode it's also not possible because you should set this policy during container creation. You can't do this for existing container.
"Custom conflict resolution policy is available only for SQL API accounts and can be set only at creation time. It is not possible to set a custom resolution policy on an existing container."

So the correct options for both containers is the last one - "an application that reads from the conflict feed.
upvoted 3 times

Torent2005 8 months ago
In general, you can't modify conflict policy after container creation.
upvoted 1 times

HOTSPOT -

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

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

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

The databases contain the containers shown in the following table.

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

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

NOTE: Each correct selection is worth one point.

Hot Area:

Answer Area

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

Correct Answer:

Answer Area

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

Box 1: No -

Four containers with 1000 RU/s each.

Box 2: No -

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

Box 3: Yes -

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

Reference:

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

ognamala Highly Voted 9 months, 1 week ago

Answers are correct but the reasoning for the first answer makes no sense - at minimum you will be charged 10% x 5000 = 500 (db autoscale) + 10% x 10,000 (container 1 autoscale) = 1500 in total

upvoted 8 times

Zaytsev 1 month ago

You are only counting one container. If you had the four left on db1, you get 4000+1500= 5500 minimum.

upvoted 1 times

HOTSPOT -
You have a database named telemetry in an Azure Cosmos DB Core (SQL) API account that stores IoT data. The database contains two containers named readings and devices.
Documents in readings have the following structure.
⇨ id
⇨ deviceid
⇨ timestamp
⇨ ownerid
⇨ measures (array)
- type
- value
- metricid
Documents in devices have the following structure.
⇨ id
⇨ deviceid
⇨ owner
- ownerid
- emailaddress
- name
⇨ brand
⇨ model
For each of the following statements, select Yes if the statement is true. Otherwise, select No.
NOTE: Each correct selection is worth one point.
Hot Area:

Answer Area

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

Correct Answer:

Answer Area

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

Box 1: Yes -
Need to join readings and devices.

Box 2: No -
Only readings is required. All required fields are in readings.

Box 3: No -
Only devices is required. All required fields are in devices.

grada Highly Voted 10 months, 1 week ago
Who ever wrote the suggested responses has no idea of what a JOIN does in a document database... it has nothing to do with joining documents, but is an inner-document join.

It's NYN if the first questions means returning only devices, and none of the readings. It's YYN otherwise, because readings are in a separate container, and should be fetched using a second query.

upvoted 6 times

Internal_Koala Highly Voted 7 months, 4 weeks ago
No. Yes. Yes.

==========
First: No

SELECT *
FROM d
WHERE d.emailaddress =

"To return (?) for all devices ...". Sounds like the word "metrics" might be missing. If so, the answer is Yes and a second query must be added.
==========
SELECT d.deviceid
FROM d
WHERE d.emailaddress = ?

SELECT *
FROM m
WHERE m.deviceid = @deviceid

==========
Second: Yes

SELECT m.deviceid
FROM m
WHERE m.metricid = ?

SELECT d.deviceid, o.ownerid, d.timestamp
FROM d
JOIN o IN d.owner
WHERE d.deviceid = @deviceid

==========
Third: Yes

SELECT d.deviceid, o.ownerid, d.emailaddress, d.model
FROM d
JOIN o IN d.owner

upvoted 6 times

nope1234567 5 months ago
I disagree on the second one. All fields to return are in the Readings container, why would you need to join the second container ? FK are sufficients here

upvoted 1 times

susejzepol 6 months ago
i think like you.

upvoted 1 times

avocacao Most Recent 5 months ago
Y - requires 2 requests to get from 2 containers
Y - need join to get 'value' in measures array
Y - need join to get 'ownerid' and 'emailaddress' in owner array
https://learn.microsoft.com/en-us/azure/cosmos-db/nosql/query/join

upvoted 2 times

Alex22022 3 months, 3 weeks ago
Third - No; Devices container: select c.deviceId, c.owner.ownerId, c.owner.emailaddress, c.model from c

upvoted 1 times

kdsingh 1 year ago
The correct answer is YYN
- Yes: you need 2 requests, because the data is in 2 different containers.

- Yes: you need join to get the data from Array
- No: No joins required

upvoted 6 times

ognamala 9 months, 1 week ago

Why would the second one be yes ? We dont need any data from the array.

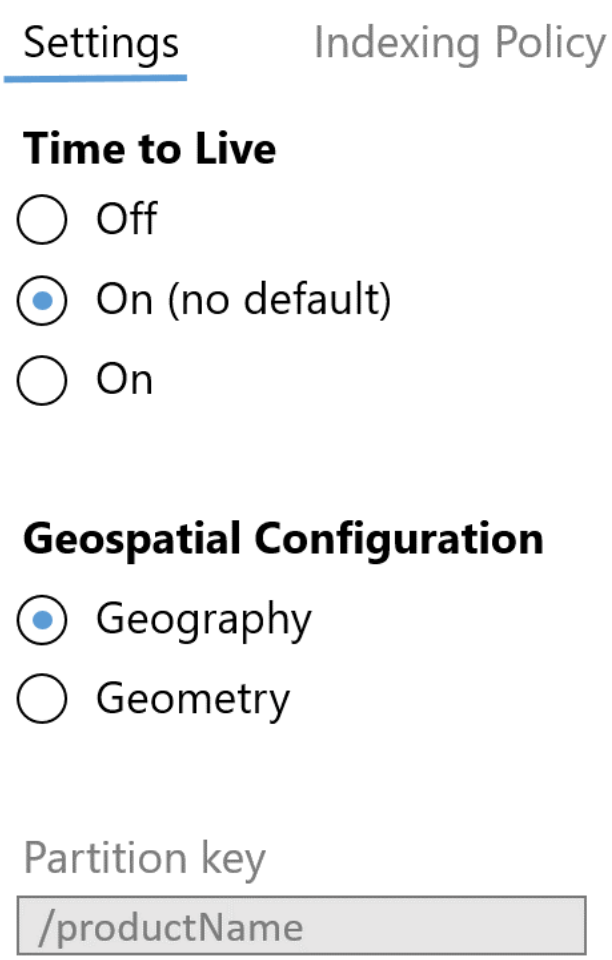
upvoted 1 times

Alex22022 3 months, 3 weeks ago

Because you need value and metricid properties from the measures array.

upvoted 1 times

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



Which statement describes the configuration of the container?

A. All items will be deleted after one year.

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

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

D. All items will be deleted after one hour.

Correct Answer: *C*
When DefaultTimeToLive is -1 then your Time to Live setting is On (No default)
Time to Live on a container, if present and the value is set to "-1", it is equal to infinity, and items don't expire by default.
Time to Live on an item:
This Property is applicable only if DefaultTimeToLive is present and it is not set to null for the parent container.
If present, it overrides the DefaultTimeToLive value of the parent container.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/sql/time-to-live

Question #5 *Topic 2*

You have an Azure Cosmos DB Core (SQL) API account named account1 that is configured for automatic failover. The account1 account has a single read-write region in West US 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. The account will be unavailable to writes during the change

B. The provisioned throughput for account1 will increase

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

D. A manual failover will occur

Correct Answer: *A*

Note: The Update-AzCosmosDBAccountFailoverPriority command updates Failover Region Priority of a CosmosDB Account.

Parameter -FailoverPolicy is an array of strings having region names, ordered by failover priority. E.g eastus, westus

Reference:

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

arnabdt 1 month, 3 weeks ago

Selected Answer: C

C. The account will be configured for multi-region writes.
The Update-AzCosmosDBAccountFailoverPriority command with the -FailoverPolicy parameter updates the failover policy for the Cosmos DB account by specifying the order in which the regions are selected for failover.
In this case, the command updates the failover policy for account1 to include both West US and East US regions, in that order. This means that if the primary (read-write) region in West US becomes unavailable, Cosmos DB will automatically failover to the secondary (read-only) region in East US.
The command does not affect the availability of the account or the provisioned throughput. It only updates the failover policy to allow for multi-region writes in case of a failover.
Option C is correct because the command enables multi-region writes in the failover policy. Options A and B are incorrect because the command does not affect the account's availability or provisioned throughput. Option D is incorrect because the command does not trigger a manual failover.

upvoted 1 times

Ranzzzan 2 months, 3 weeks ago

IMO its A , The Azure Cosmos DB account must be configured for manual failover for manual failover to succeed

upvoted 2 times

virgilpza 2 months, 3 weeks ago

according to https://learn.microsoft.com/en-us/azure/cosmos-db/scripts/powershell/common/failover-priority-update, thats what i believe to. "Any change to a region with failoverPriority=0 triggers a manual failover and can only be done to an account configured for manual failover. Changes to all other regions simply changes the failover priority for an Azure Cosmos DB account."

upvoted 1 times

christosst 5 months, 2 weeks ago

I think is D - West US is the failoverPriority=0 right?

Any change to a region with failoverPriority=0 triggers a manual failover and can only be done to an account configured for manual failover. Changes to all other regions simply changes the failover priority for an Azure Cosmos DB account.

https://learn.microsoft.com/en-us/azure/cosmos-db/scripts/powershell/common/failover-priority-update

upvoted 4 times

Internal_Koala 7 months, 4 weeks ago

As per exact example on the source:
https://learn.microsoft.com/en-us/azure/cosmos-db/sql/manage-with-powershell#trigger-manual-failover

From this website and according to A:
"If you perform a manual failover operation while an asynchronous throughput scaling operation is in progress, the throughput scale-up operation will be paused. It will resume automatically when the failover operation is complete."

1. Has to be an asynchronous throughput scaling operation
2. It will continue after failover completes.

upvoted 1 times

ExamsBertia 8 months ago

They said: "a single read-write region in West US and a read region in East US"
If you flip the FailoverPolicy to "East US" you can't write because it's a read region. So i think A is a correct answer.

upvoted 1 times

IDATA 8 months, 2 weeks ago

I think its worng, the answer is D

Becouse just have 2 failover regions, if u flip this 2 this gona make a Manual Failover

https://docs.microsoft.com/en-us/azure/cosmos-db/sql/manage-with-powershell

upvoted 3 times

codingdown 6 months ago

you are not performing the failover, just deciding the autofailovr policy

upvoted 1 times

Alex22022 3 months, 2 weeks ago

According to the link below, a manual failover is performed if you change one of the regions to priority = 0.

https://learn.microsoft.com/en-us/training/modules/write-scripts-for-azure-cosmos-db-sql-api/7-initiate-failovers

upvoted 1 times

HOTSPOT -

You are developing an application that will connect to an Azure Cosmos DB Core (SQL) API account. The account has a single read-write region and one additional read region. The regions are configured for automatic failover.

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

```
01 {
02   "connectionStrings": [
03     {
04       "connectionString":
05       "AccountEndpoint=https://constosodbaccount.documents.azure.com:443/;
06       AccountKey=MwUgRnGti4vErT2rfPPFdTFFyI9KyI9Kbe1RPGv7OQdHo6VZ2i45TcJzrd4J80zYxrEATzyZh0m1nJaNFA==;",
07       "description": "Primary SQL Connection String"
08     },
09     {
10       "connectionString":
11       "AccountEndpoint=https://constosodbaccount.documents.azure.com:443/;
12       AccountKey=gfThRnGti4vErT2rfPPFdTFFyI43529Kbe1RPGv7OQdHo6VZ2i45TcJzrd4J80zYxrfatzyZh0m1nJaNFA==;",
13       "description": "Secondary SQL Connection String"
14     },
15     {
16       "connectionString":
17       "AccountEndpoint=https://constosodbaccount.documents.azure.com:443/;
18       AccountKey=WGykBc1PHJoos6MdErT2rfPPFx9yI9Kbe1RPGv7OQlIQwQNxq6QdOXjxgyLLebXBp8uJu7FyJy3Uv1vuK2A==;"
19       "description": "Primary Read-Only SQL Connection String"
20     },
21     {
22       "connectionString":
23       "AccountEndpoint=https://constosodbaccount.documents.azure.com:443/;
24       AccountKey=k2DZI0oY4Jc7QeUJqVGH3csda6EyI9Kbe1RPGv7OQErT2rfPPFtbwTPfKAgl9zVxC0MDNn8xPpQrednVYcQ==;",
25       "description": "Secondary Read-Only SQL Connection String"
26     }
27   ]
28 }
```

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
If the primary write region fails, applications that write to the database must use a different connection string to continue to use the service	○	○
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	○	○
Applications can choose from which region they read by setting the `PreferredLocations` property within their connection properties	○	○

Correct Answer:

Answer 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	○	○
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	○	○
Applications can choose from which region they read by setting the `PreferredLocations` property within their connection properties	○	○

Box 1: No -

The same connection string can still be used.

Azure Cosmos DB Core read write region connection strings

Azure Cosmos DB "primary write region" failover "connection string"

Box 2: No -
The AccountEndpoint for both are the same.

Box 3: Yes -
The ConnectionPolicy.PreferredLocations property gets and sets the preferred locations (regions) for geo-replicated database accounts in the Azure Cosmos DB service. For example, "East US" as the preferred location.
When EnableEndpointDiscovery is true and the value of this property is non-empty, the SDK uses the locations in the collection in the order they are specified to perform operations, otherwise if the value of this property is not specified, the SDK uses the write region as the preferred location for all operations.
Reference:
https://docs.microsoft.com/en-us/dotnet/api/microsoft.azure.documents.client.connectionpolicy.preferredlocations

Question #7 *Topic 2*

You have a global ecommerce application that stores data in an Azure Cosmos DB Core (SQL) API account. The account is configured for multi-region writes.
You need to create a stored procedure for a custom conflict resolution policy for a new container. In the event of a conflict caused by a deletion, the deletion must always take priority.
Which parameter should you check in the stored procedure function?

A. isTombstone

B. conflictingItems

C. existingItem

D. incomingItem

Correct Answer: *A*
isTombstone: Boolean indicating if the incomingItem is conflicting with a previously deleted item. When true, existingItem is also null.
Incorrect:
conflictingItems: Array of the committed version of all items in the container that are conflicting with incomingItem on ID or any other unique index properties. existingItem: The currently committed item. This value is non-null in an update and null for an insert or deletes. incomingItem: The item being inserted or updated in the commit that is generating the conflicts. Is null for delete operations.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/sql/how-to-manage-conflicts

virgilpza 2 months, 3 weeks ago
A is the correct answer.
See https://learn.microsoft.com/en-us/azure/cosmos-db/nosql/how-to-manage-conflicts?tabs=dotnetv2%2Capi-async%2Casync
upvoted 1 times

Question #8 *Topic 2*

You have an Azure Cosmos DB Core (SQL) API 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 consistency level of requests to session.

B. Change the account endpoint to http://account1.documents.azure.com.

C. Change the account endpoint to http://account1.sqlx.cosmos.azure.com.

D. Change the connection mode to direct.

E. Change the consistency level of requests to strong.

Correct Answer: *AC*
C: Modify your application's connection string to use the new dedicated gateway endpoint.
All dedicated gateway connection strings follow the same pattern. Remove documents.azure.com from your original connection string and replace it with sqlx.cosmos.azure.com. A dedicated gateway will always have the same connection string, even if you remove and reprovision it.
A: You must adjust the request consistency to session or eventual. If not, the request will always bypass the integrated cache.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/how-to-configure-integrated-cache

TRUESON 3 weeks, 1 day ago
All dedicated gateway connection strings follow the same pattern. Remove documents.azure.com from your original connection string and replace it with sqlx.cosmos.azure.com

A
C
upvoted 1 times

purplefish 3 weeks, 4 days ago
Selected Answer: AD
To use the integrated cache with an Azure Cosmos DB Core (SQL) API account, you need to use the "Gateway" connection mode and the "Session" or "Consistent Prefix" consistency levels. Therefore, you should perform the following actions for App1:

A. Change the consistency level of requests to session.
D. Change the connection mode.
upvoted 1 times

arnabdt 1 month, 3 weeks ago
Selected Answer: AD
To ensure that App1 can use the integrated cache, you need to configure the Cosmos DB client SDK to use the Direct connection mode and set the consistency level to session. This is because the integrated cache is available only in the Direct connection mode and the session consistency level is recommended to maximize cache hit rates.
upvoted 1 times

Question #9 *Topic 2*

You have an Azure Cosmos DB Core (SQL) API 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 eventual consistency level.

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

C. Writes will use the strong consistency level.

D. The write order is not guaranteed during replication.

Correct Answer: *C*
Overriding the default consistency level only applies to reads within the SDK client. An account configured for strong consistency by default will still write and replicate data synchronously to every region in the account. When the SDK client instance or request overrides this with Session or weaker consistency, reads will be performed using a single replica.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/consistency-levels

TRUESON 3 weeks, 1 day ago
C

Overriding the default consistency level only applies to reads within the SDK client. An account configured for strong consistency by default will still write and replicate data synchronously to every region in the account. https://learn.microsoft.com/en-us/azure/cosmos-db/nosql/how-to-manage-consistency?tabs=portal%2Cdotnetv2%2Capi-async#override-the-default-consistency-level

upvoted 1 times

Topic 3 - Question Set 3

Question #1 *Topic 3*

You have an Azure Cosmos DB Core (SQL) API account that uses a custom conflict resolution policy. The account has a registered merge procedure that throws a runtime exception.
The runtime exception prevents conflicts from being resolved.
You need to use an Azure function to resolve the conflicts.
What should you use?

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

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

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

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

Correct Answer: *D*
The Azure Cosmos DB Trigger uses the Azure Cosmos DB Change Feed to listen for inserts and updates across partitions. The change feed publishes inserts and updates, not deletions.
Reference:
https://docs.microsoft.com/en-us/azure/azure-functions/functions-bindings-cosmosdb

ognamala Highly Voted 9 months, 1 week ago
Selected Answer: A
Correct answer should be "A" - we need to resolve conflicts, hence we definitely need to read the conflicts feed, hence answers B and C are immediately eliminated as they are pull from the changes feed, answer D mentions an Azure Cosmos DB trigger, but this is only for the change feed, not for the conflicts feed, hence A is the correct answer since there is no trigger mechanism for the conflict feed in Cosmos DB
upvoted 5 times

virgilpza Most Recent 2 months, 3 weeks ago
Correct answer is A
upvoted 1 times

klepper 7 months, 4 weeks ago
Selected Answer: A
Correct answer is A
upvoted 1 times

Shiggi 8 months, 2 weeks ago
Selected Answer: A
Correct answer is A: all conflicts related data is stored in the conflicts feed, you have to orchestrate the function using a timer
upvoted 1 times

remz 9 months ago
Selected Answer: A
Answer A
upvoted 1 times

ognamala 9 months, 1 week ago
Selected Answer: C
As grada explained very well, the answer should be C
upvoted 1 times

ognamala 9 months, 1 week ago
Actually, ignore this answer - C is retrieving from the change feed not the conflicts feed
upvoted 1 times

ognamala 9 months, 1 week ago
Correct answer is A
upvoted 1 times

mybiai 10 months ago
Selected Answer: B

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

upvoted 2 times

avocacao 10 months ago

Answer should be A. Need to read from conflict feed but there is no trigger mechanism for the conflict feed in Cosmos DB.

upvoted 4 times

grada 10 months, 1 week ago

The correct answer is C, only change feed triggers are available, and conflict feed needs to be manually queried from a timer-triggered function.

Source: https://docs.microsoft.com/en-us/azure/azure-functions/functions-bindings-cosmosdb-v2-trigger?tabs=in-process%2Cextensionv4&pivots=programming-language-csharp#attributes

Nothing conflict-feed-related there, only change-feed-related. Conflict feed needs to be manually triggered like this: https://docs.microsoft.com/en-us/azure/cosmos-db/sql/how-to-manage-conflicts?tabs=dotnetv3%2Capi-async%2Casync#read-from-conflict-feed

upvoted 2 times

The following is a sample of a document in orders.

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

The orders container uses customerId as the partition key.

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

- Ensure that the report can run as quickly as possible.
- Minimize the consumption of request units (RUs).

What should you do?

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

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

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

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

Correct Answer: *B*

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

Reference:

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

purplefish 3 weeks, 4 days ago

Selected Answer: B

Option D suggests populating the aggregates by using SQL queries that run daily. While this may reduce the RU consumption during querying, it may not necessarily minimize RU consumption overall. Additionally, this approach may result in stale data since the aggregates are only updated once a day.

The best approach to minimize RU consumption and ensure the report runs as quickly as possible is to use the change feed to populate the aggregates in real-time, as suggested in option B. This way, the aggregates are always up-to-date, and the report can be generated quickly and with minimal RU consumption.

upvoted 1 times

BOT_123 3 months, 3 weeks ago

Selected Answer: B

B Is Correct

upvoted 1 times

TimSss 7 months, 3 weeks ago

Selected Answer: D

I would go with D, we need to minimize RU, so we run the job daily which should be fine for these types of reports (using monthly data). Using the change feed costs much more RU.

upvoted 4 times

remz 9 months ago

Selected Answer: B

B Is Correct

upvoted 1 times

Gall 9 months, 2 weeks ago

Selected Answer: D

I would go with D, as we need to reduce RU. An additional container writes costs RU and the change-feed will be updated every time a new item appears.

upvoted 2 times

grada 10 months, 1 week ago

Selected Answer: B

After processing items in the change feed, you can build a materialized view and persist aggregated values back in Azure Cosmos DB. If you're using Azure Cosmos DB to build a game, you can, for example, use change feed to implement real-time leaderboards based on scores from completed games.

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

upvoted 3 times

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:

- A function named Fn1 that reads the change feed of cn1
- A function named Fn2 that reads the change feed of cn2
- A function named Fn3 that reads the change feed of cn3

You perform the following actions:

- Delete an item named item1 from cn1.
- Update an item named item2 in cn2.
- For an item named item3 in cn3, update the item time to live to 3,600 seconds.

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

NOTE: Each correct selection is worth one point.

Hot Area:

Answer Area

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

Correct Answer:

Answer Area

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

Box 1: No -

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

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

Box 2: No -

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

Box 3: Yes -

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

Reference:

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

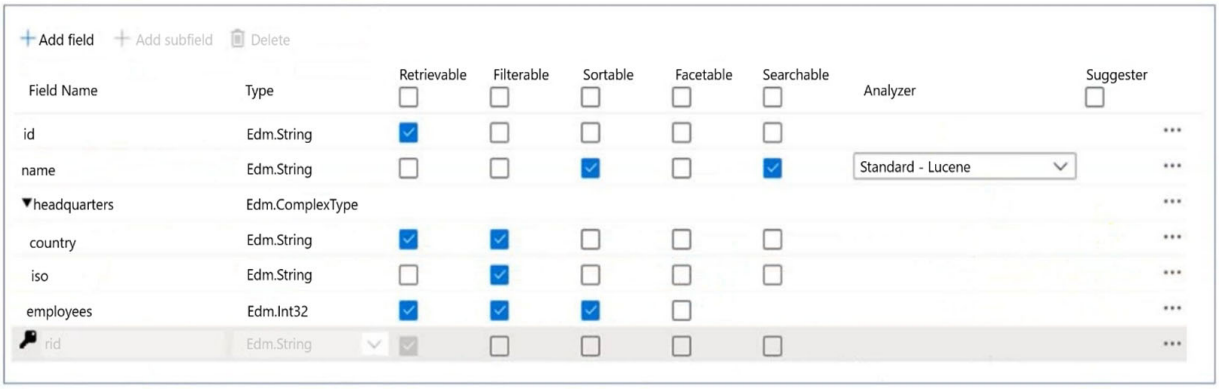
basiltomato 2 months, 3 weeks ago

Correct -
N - https://learn.microsoft.com/en-us/azure/cosmos-db/change-feed#features-of-change-feed
N - https://stackoverflow.com/questions/68409298/how-to-tell-the-difference-between-insert-and-update-in-cosmos-db-change-feed
Y - 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

upvoted 2 times

HOTSPOT -

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



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

NOTE: Each correct selection is worth one point.

Hot Area:

Answer Area

The **[answer choice]** field is limited to exact match comparisons

- country
- id
- name

The **[answer choice]** field is hidden form the search results

- country
- id
- name

Correct Answer:

Answer Area

The **[answer choice]** field is limited to exact match comparisons

- country
- id
- name

The **[answer choice]** field is hidden form the search results

- country
- id
- name

Box 1: country -

The country field is filterable.

Note: filterable: Indicates whether to enable the field to be referenced in $filter queries. Filterable differs from searchable in how strings are handled. Fields of type

Edm.String or Collection(Edm.String) that are filterable do not undergo lexical analysis, so comparisons are for exact matches only.

Box 2: name -

The name field is not Retrievable.
Retrievable: Indicates whether the field can be returned in a search result. Set this attribute to false if you want to use a field (for example, margin) as a filter, sorting, or scoring mechanism but do not want the field to be visible to the end user.
Note: searchable: Indicates whether the field is full-text searchable and can be referenced in search queries.
Reference:
https://docs.microsoft.com/en-us/rest/api/searchservice/create-index

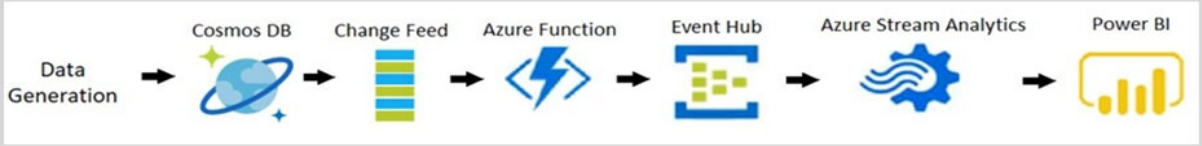
Question #5 *Topic 3*

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have a container named container1 in an Azure Cosmos DB Core (SQL) API account.
You need to make the contents of container1 available as reference data for an Azure Stream Analytics job.
Solution: You create an Azure Synapse pipeline that uses Azure Cosmos DB Core (SQL) API as the input and Azure Blob Storage as the output.
Does this meet the goal?

A. Yes

B. No

Correct Answer: *B*
Instead create an Azure function that uses Azure Cosmos DB Core (SQL) API change feed as a trigger and Azure event hub as the output.
The Azure Cosmos DB change feed is a mechanism to get a continuous and incremental feed of records from an Azure Cosmos container as those records are being created or modified. Change feed support works by listening to container for any changes. It then outputs the sorted list of documents that were changed in the order in which they were modified.
The following diagram represents the data flow and components involved in the solution:



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

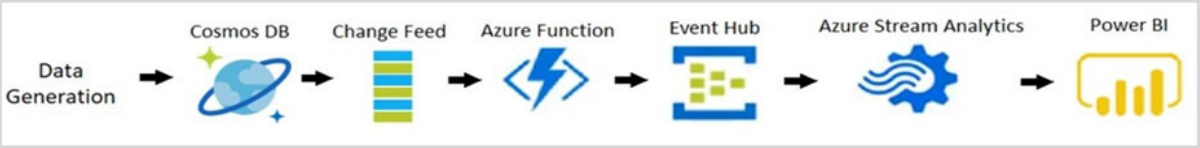
Question #6 *Topic 3*

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have a container named container1 in an Azure Cosmos DB Core (SQL) API account.
You need to make the contents of container1 available as reference data for an Azure Stream Analytics job.
Solution: You create an Azure Data Factory pipeline that uses Azure Cosmos DB Core (SQL) API as the input and Azure Blob Storage as the output.
Does this meet the goal?

A. Yes

B. No

Correct Answer: *B*
Instead create an Azure function that uses Azure Cosmos DB Core (SQL) API change feed as a trigger and Azure event hub as the output.
The Azure Cosmos DB change feed is a mechanism to get a continuous and incremental feed of records from an Azure Cosmos container as those records are being created or modified. Change feed support works by listening to container for any changes. It then outputs the sorted list of documents that were changed in the order in which they were modified.
The following diagram represents the data flow and components involved in the solution:



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

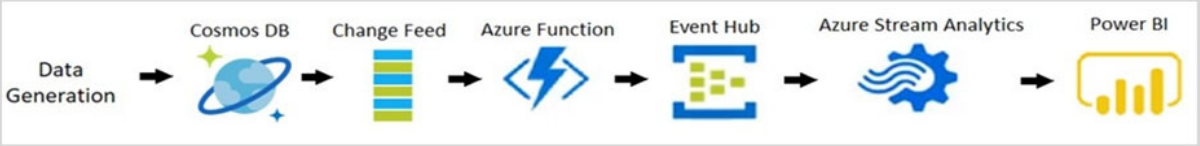
Question #7 *Topic 3*

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have a container named container1 in an Azure Cosmos DB Core (SQL) API account.
You need to make the contents of container1 available as reference data for an Azure Stream Analytics job.
Solution: You create an Azure function that uses Azure Cosmos DB Core (SQL) API change feed as a trigger and Azure event hub as the output.
Does this meet the goal?

A. Yes

B. No

Correct Answer: *A*
The Azure Cosmos DB change feed is a mechanism to get a continuous and incremental feed of records from an Azure Cosmos container as those records are being created or modified. Change feed support works by listening to container for any changes. It then outputs the sorted list of documents that were changed in the order in which they were modified.
The following diagram represents the data flow and components involved in the solution:



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

Flammkuchen 7 months, 3 weeks ago
I don't thing that Y is correct. While event hub would be a great input source for Stream Analytics *Streaming* Data, it is not supported as input for Stream Analytics *Reference* Data. https://learn.microsoft.com/en-us/azure/stream-analytics/stream-analytics-add-inputs.

Using Data Factory is the recommended solution for reference data in the Microsoft docs: https://learn.microsoft.com/en-us/azure/stream-analytics/stream-analytics-use-reference-data
upvoted 3 times

Question #8 *Topic 3*

You have an Azure Cosmos DB Core (SQL) API 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. only the changed properties and the system-defined 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. all the properties of the updated items

Correct Answer: *B*
Change feed is available for each logical partition key within the container.
The change feed is sorted by the order of modification within each logical partition key value.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/change-feed

arnabdt 1 month, 3 weeks ago

Selected Answer: D

D is correct

upvoted 1 times

roxsauromech 5 months ago

Selected Answer: D

D is the correct answer

upvoted 3 times

khushbu123 7 months, 3 weeks ago

Selected Answer: D

It should be D

upvoted 4 times

TimSss 7 months, 3 weeks ago

Selected Answer: D

Wrong. D is correct, entire doc is passed

upvoted 4 times

DudeWheresMyCar 8 months ago

Answer is D as all properties are sent to the change feed. This is testable by creating an Azure Function with CosmosDB input trigger.

upvoted 4 times

DRAG DROP -

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

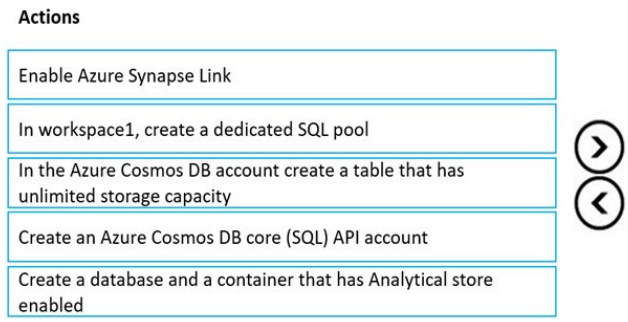
You have an Azure Table Storage account that stores operational data.

You need to replace the Table storage account with Azure Cosmos DB Core (SQL) API. The solution must meet the following requirements:

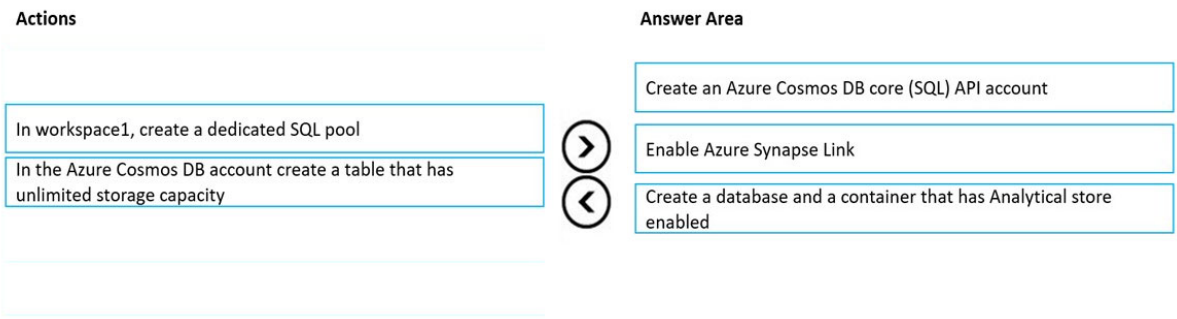
- Support queries from the serverless SQL pool.
- Only pay for analytical compute when running queries.
- Ensure that analytical processes do NOT affect operational processes.

Which three actions should you perform in sequence? To answer, move the appropriate actions from the list of actions to the answer area and arrange them in the correct order.

Select and Place:



Correct Answer:



Step 1: Create an Azure Cosmos DB core (SQL) API account

Step 2: Enable Azure Synapse Link

Synapse Link creates a tight seamless integration between Azure Cosmos DB and Azure Synapse Analytics.

Serverless SQL pool allows you to query and analyze data in your Azure Cosmos DB containers that are enabled with Azure Synapse Link. You can analyze data in near real-time without impacting the performance of your transactional workloads.

Step 3: Create a database and a container that has Analytical store enabled

Create an analytical store enabled container

Reference:

https://docs.microsoft.com/en-us/azure/cosmos-db/configure-synapse-link

Question #10 *Topic 3*

You have a database named db1 in an Azure Cosmos DB Core (SQL) API account named account1.
You need to write JSON data to db1 by using Azure Stream Analytics. The solution must minimize costs.
Which should you do before you can use db1 as an output of Stream Analytics?

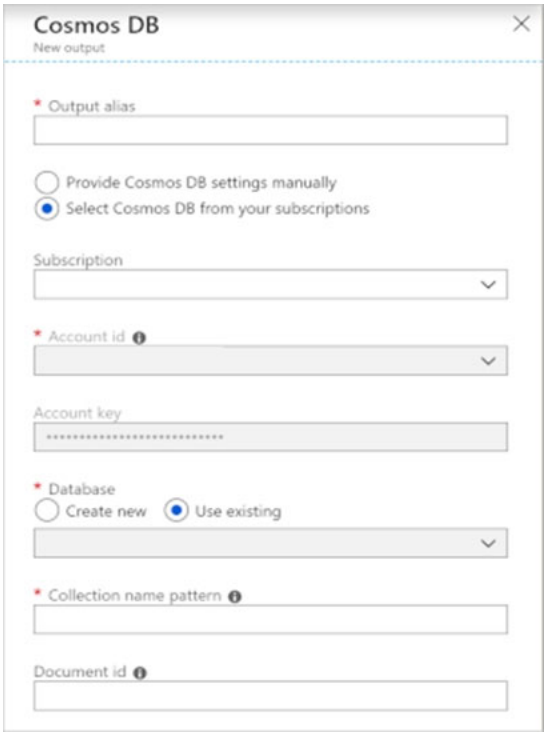
A. In account1, add a private endpoint

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

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

D. In account1, enable a dedicated gateway

Correct Answer: *A*
Azure Cosmos DB settings for JSON output.
Using Azure Cosmos DB as an output in Stream Analytics generates the following prompt for information.



Field: Description -
Output alias: An alias to refer to this output in your Stream Analytics query.
Subscription:The Azure subscription.
Account ID: The name or endpoint URI of the Azure Cosmos DB account.
Etc.
Note: A private endpoint could be used for a VPN connection.
Reference:
https://docs.microsoft.com/en-us/azure/stream-analytics/stream-analytics-documentdb-output

basiltomato 2 months, 2 weeks ago

Selected Answer: B
"Stream Analytics doesn't create containers in your database. Instead, it requires you to create them beforehand. You can then control the billing costs of Azure Cosmos DB containers. You can also tune the performance, consistency, and capacity of your containers directly by using the Azure Cosmos DB APIs."
https://learn.microsoft.com/en-us/azure/stream-analytics/stream-analytics-documentdb-output#basics-of-azure-cosmos-db-as-an-output-target
upvoted 1 times

susejzepol 7 months, 1 week ago

Selected Answer: B
I Think that B is the correct answer because you don't need the Analytical Store.

upvoted 1 times

TimSss 7 months, 3 weeks ago

Selected Answer: B

sure it's not A, but usure about B

upvoted 2 times

essdeecee 8 months, 1 week ago

Selected Answer: B

Definitely not A

upvoted 2 times

IDATA 8 months, 2 weeks ago

Answer is B

Stream Analytics doesn't create containers in your database. Instead, it requires you to create them up front. You can then control the billing costs of Azure Cosmos DB containers, and u dont need to use Analytical Store becosue we dont have ColumnStore

upvoted 3 times

Question #11 *Topic 3*

You have a database named db1 in an Azure Cosmos DB Core (SQL) API account.

You have a third-party application that is exposed through a REST API.

You need to migrate data from the application to a container in db1 on a weekly basis.

What should you use?

A. Azure Migrate

B. Azure Data Factory

C. Database Migration Assistant

Correct Answer: *B*

You can use Copy Activity in Azure Data Factory to copy data from and to Azure Cosmos DB (SQL API).

The Azure Cosmos DB (SQL API) connector is supported for the following activities:

Copy activity with supported source/sink matrix

Mapping data flow -

Lookup activity -

Incorrect:

Not A: Azure Migrate provides a centralized hub to assess and migrate on-premises servers, infrastructure, applications, and data to Azure. It assesses on- premises databases and migrate them to Azure SQL Database or to SQL Managed Instance.

Not C: Data Migration Assistant (DMA) enables you to upgrade to a modern data platform by detecting compatibility issues that can impact database functionality on your new version of SQL Server. It recommends performance and reliability improvements for your target environment.

Reference:

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

purplefish 3 weeks, 4 days ago

Selected Answer: B

Correct Answer is B.

Azure Migrate is a service designed for migrating on-premises virtual machines and database Migration Assistant is a tool that helps you assess and migrate databases to Azure, but it is not designed for migrating data from a third-party application exposed through a REST API

upvoted 1 times

HOTSPOT -

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

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

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

streamQuery.awaitTermination()
```

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

NOTE: Each correct selection is worth one point.

Hot Area:

Answer Area

Statements	Yes	No
New and updated orders will be added to contoso-erp.orders.	○	○
The code performs bulk data ingestion from contoso-app.	○	○
Both contoso-app and contoso-erp have Analytical store enabled.	○	○

Correct Answer:

Answer Area

Statements	Yes	No
New and updated orders will be added to contoso-erp.orders.	○	○
The code performs bulk data ingestion from contoso-app.	○	○
Both contoso-app and contoso-erp have Analytical store enabled.	○	○

Box 1: No -

Streaming λ€" Append Output Mode is an outputMode in which only the new rows in the streaming DataFrame/Dataset will be written to the sink.

This is the default mode. Use append as output mode outputMode("append") when you want to output only new rows to the output sink.

Note:

Streaming λ€" Complete Output Mode is an OutputMode in which all the rows in the streaming DataFrame/Dataset will be written to the sink every time there are some updates.

Streaming λ€" Update Output Mode is an outputMode in which only the rows that were updated in the streaming DataFrame/Dataset will be written to the sink every time there are some updates.

Box 2: No -
Structured Streaming is a scalable and fault-tolerant stream processing engine built on the Spark SQL engine. You can express your streaming computation the same way you would express a batch computation on static data. The Spark SQL engine will take care of running it incrementally and continuously and updating the final result as streaming data continues to arrive.

Box 3: Yes -
Synapse Apache Spark also allows you to ingest data into Azure Cosmos DB. It is important to note that data is always ingested into Azure Cosmos DB containers through the transactional store. When Synapse Link is enabled, any new inserts, updates, and deletes are then automatically synced to the analytical store.
Reference:
https://sparkbyexamples.com/spark/spark-streaming-outputmode/ https://spark.apache.org/docs/latest/structured-streaming-programming-guide.html https://docs.microsoft.com/en-us/azure/synapse-analytics/synapse-link/how-to-query-analytical-store-spark

Nath2 6 months, 3 weeks ago
And Question 1 is "Yes" - as this is reading from the change feed and so gets all the inserts and updates.
upvoted 2 times

Bharat 4 months ago
Since the append mode is enabled, updates will not be recorded, only new records are added so your answer is partially right/wrong. I would go with 'No' for Question 1.
upvoted 3 times

Nath2 6 months, 3 weeks ago
This is linked to this documentation:
https://learn.microsoft.com/en-us/azure/synapse-analytics/synapse-link/how-to-query-analytical-store-spark-3#load-streaming-dataframe-from-azure-cosmos-db-container

And so doesn't need the analytical store to be enabled, so "No" for the third question.
upvoted 2 times

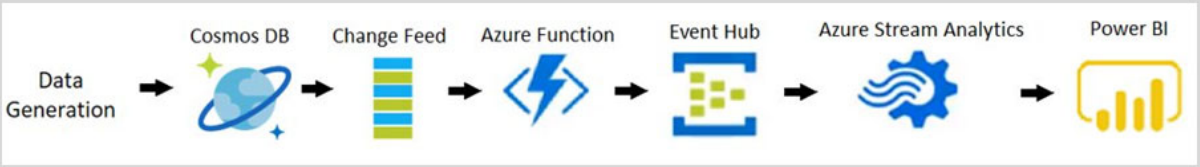
Bharat 4 months ago
I agree. For the third question, it should be a "No"
upvoted 1 times

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have a container named container1 in an Azure Cosmos DB Core (SQL) API account.
You need to make the contents of container1 available as reference data for an Azure Stream Analytics job.
Solution: You create an Azure function to copy data to another Azure Cosmos DB Core (SQL) API container.
Does this meet the goal?

A. Yes

B. No

Correct Answer: *B*
Instead: Create an Azure function that uses Azure Cosmos DB Core (SQL) API change feed as a trigger and Azure event hub as the output.
Note: The Azure Cosmos DB change feed is a mechanism to get a continuous and incremental feed of records from an Azure Cosmos container as those records are being created or modified. Change feed support works by listening to container for any changes. It then outputs the sorted list of documents that were changed in the order in which they were modified.
The following diagram represents the data flow and components involved in the solution:

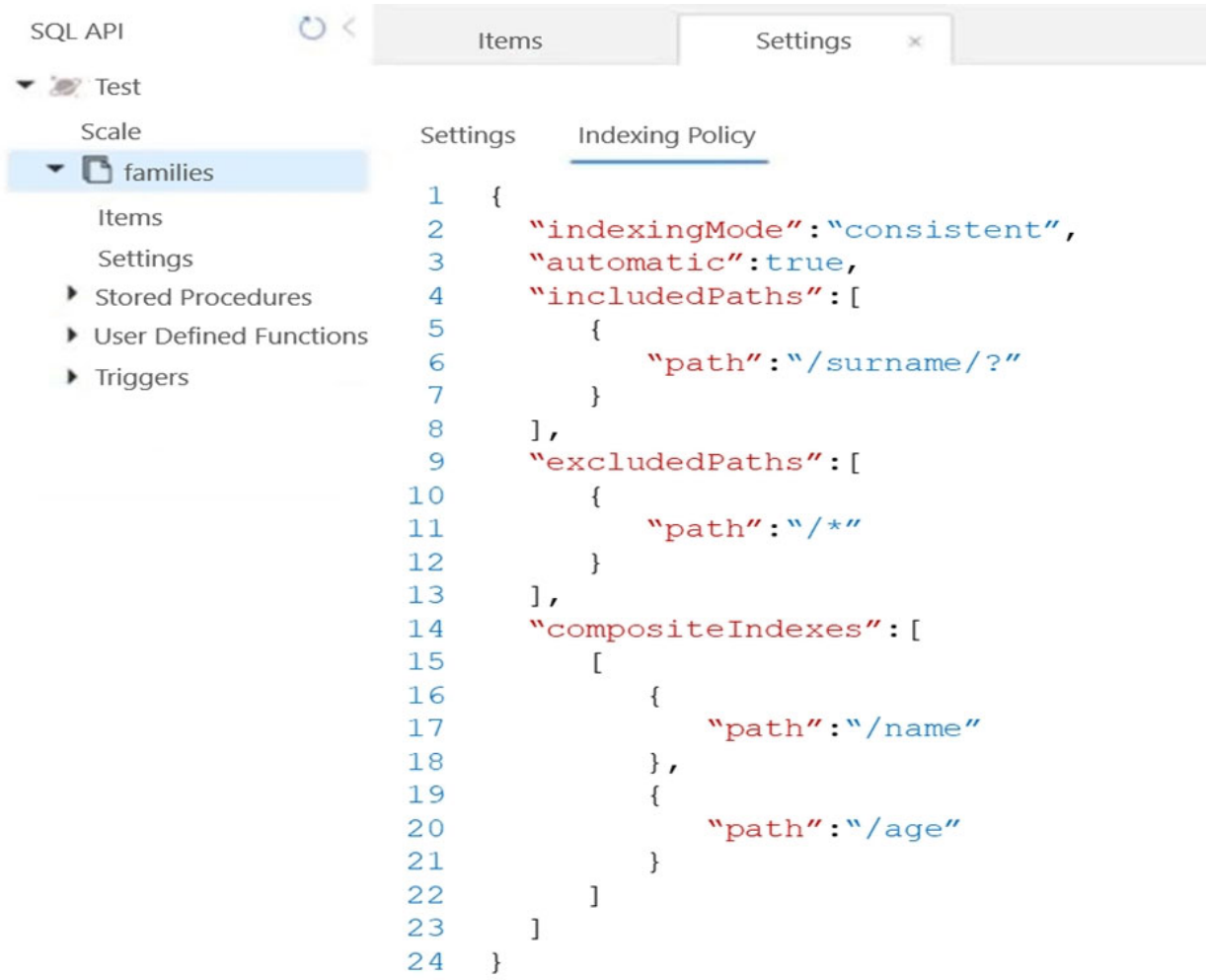


Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/sql/changefeed-ecommerce-solution

Question #1 *Topic 4*

HOTSPOT -

You have the indexing policy shown in the following exhibit.



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

NOTE: Each correct selection is worth one point.

Hot Area:

Answer Area

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

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

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

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

Correct Answer:

Answer Area

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

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

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

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

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

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

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

- If the composite index paths do not match the sequence of the properties in the ORDER BY clause, then the composite index can't support the query.
- The order of composite index paths (ascending or descending) should also match the order in the ORDER BY clause.
- The composite index also supports an ORDER BY clause with the opposite order on all paths.

Box 2: At the same time as the item creation

Azure Cosmos DB supports two indexing modes:

- Consistent: The index is updated synchronously as you create, update or delete items. This means that the consistency of your read queries will be the consistency configured for the account.
- None: Indexing is disabled on the container.

Reference:

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

Question #2 *Topic 4*

You have a container named container1 in an Azure Cosmos DB Core (SQL) API account. Upserts of items in container1 occur every three seconds.
You have an Azure Functions app named function1 that is supposed to run whenever items are inserted or replaced in container1.
You discover that function1 runs, but not on every upsert.
You need to ensure that function1 processes each upsert within one second of the upsert.
Which property should you change in the Function.json file of function1?

A. checkpointInterval

B. leaseCollectionsThroughput

C. maxItemsPerInvocation

D. feedPollDelay

Correct Answer: *D*
With an upsert operation we can either insert or update an existing record at the same time.
FeedPollDelay: The time (in milliseconds) for the delay between polling a partition for new changes on the feed, after all current changes are drained. Default is
5,000 milliseconds, or 5 seconds.
Incorrect Answers:
A: checkpointInterval: When set, it defines, in milliseconds, the interval between lease checkpoints. Default is always after each Function call.
C: maxItemsPerInvocation: When set, this property sets the maximum number of items received per Function call. If operations in the monitored collection are performed through stored procedures, transaction scope is preserved when reading items from the change feed. As a result, the number of items received could be higher than the specified value so that the items changed by the same transaction are returned as part of one atomic batch.
Reference:
https://docs.microsoft.com/en-us/azure/azure-functions/functions-bindings-cosmosdb-v2-trigger

HOTSPOT -

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

The following is a sample of a document in container1.

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

The container1 container has the following indexing policy.

```
{
    "indexingMode": "consistent",
    "includedPaths": [
        {
            "path": "/*"
        },
        {
            "path": "/address/city/?"
        }
    ],
    "excludedPaths": [
        {
            "path": "/address/*"
        },
        {
            "path": "/firstname/?"
        }
    ]
}
```

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

NOTE: Each correct selection is worth one point.

Hot Area:

Answer Area

Statements	Yes	No
The `/isActivelyEnrolled` property is included in the index	◯	◯
The `/fisrtname` property is included in the index	◯	◯
The `/address/city` property is included in the index	◯	◯

Correct Answer:

Answer Area

Statements	Yes	No
The `/isActivelyEnrolled` property is included in the index	◯	○
The `/fisrtname` property is included in the index	○	◯
The `/address/city` property is included in the index	◯	○

Box 1: Yes -
"path": "/*" is in includePaths.
Include the root path to selectively exclude paths that don't need to be indexed. This is the recommended approach as it lets Azure Cosmos DB proactively index any new property that may be added to your model.

Box 2: No -
"path": "/firstName/?" is in excludePaths.

Box 3: Yes -
"path": "/address/city/?" is in includePaths
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/index-policy

You have the following query.

```
SELECT * FROM c
WHERE c.sensor = "TEMP1"
AND c.value < 22
AND c.timestamp >= 1619146031231
```

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

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

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

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

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

Correct Answer: *A*
If a query has a filter with two or more properties, adding a composite index will improve performance.
Consider the following query:
SELECT * FROM c WHERE c.name = ג€Timג€ and c.age > 18
In the absence of a composite index on (name ASC, and age ASC), we will utilize a range index for this query. We can improve the efficiency of this query by creating a composite index for name and age.
Queries with multiple equality filters and a maximum of one range filter (such as >,<, <=, >=, !=) will utilize the composite index.
Reference:
https://azure.microsoft.com/en-us/blog/three-ways-to-leverage-composite-indexes-in-azure-cosmos-db/

TimSss 7 months, 3 weeks ago

Selected Answer: A
Open the given link, the msft docs show explain this example
upvoted 2 times

ExamsBertia 7 months, 4 weeks ago

Selected Answer: A
Answer: Queries with multiple equality filters and a MAXIMUM of one range filter (such as >,<, <=, >=, !=) will utilize the composite index
upvoted 1 times

Internal_Koala 7 months, 4 weeks ago

Selected Answer: A
Solution is correct as there are two range filters:
"Two separate composite indexes are required instead of a single composite index on (name ASC, age ASC, _ts ASC) since each composite index can only optimize a single range filter."

Source:
https://learn.microsoft.com/en-us/azure/cosmos-db/index-policy#queries-with-filters-on-multiple-properties
upvoted 3 times

josch123 6 months, 3 weeks ago
This is a great source.
upvoted 2 times

essdeecee 8 months, 1 week ago

Selected Answer: B
Consider this where clause

WHERE c.categoryName = 'foo'
AND c.name = 'bar'
AND c.price < 909

Returns this result when checking index utilization:
Index Utilization Information
Utilized Single Indexes
Index Spec: /name/?
Index Impact Score: High

Index Spec: /categoryName/?
Index Impact Score: High

Index Spec: /price/?
Index Impact Score: High

Potential Single Indexes
Utilized Composite Indexes
Index Spec: /categoryName ASC, /price ASC
Index Impact Score: High

Potential Composite Indexes
Index Spec: /categoryName ASC, /name ASC, /price ASC
Index Impact Score: High

I believe the best index contains every clause.
upvoted 1 times

Question #5 *Topic 4*

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

A. Modify the index attributes in Cognitive Search to Searchable

B. Modify the index attributes in Cognitive Search to Retrievable

C. Modify the indexing policy of container1 to exclude the /* path

D. Change the indexing mode of container1 to consistent

Correct Answer: *D*
Azure Cognitive Search needs an index.
Azure Cosmos DB supports two indexing modes:
Consistent: The index is updated synchronously as you create, update or delete items.
None: Indexing is disabled on the container. This is commonly used when a container is used as a pure key-value store without the need for secondary indexes. It can also be used to improve the performance of bulk operations.
Note: In Azure Cognitive Search, a search index is your searchable content, available to the search engine for indexing, full text search, and filtered queries.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/index-policy https://docs.microsoft.com/en-us/azure/search/search-what-is-an-index

purplefish 3 weeks, 4 days ago

Selected Answer: D
To make data searchable in Azure Cognitive Search, the indexing mode of the container in Azure Cosmos DB Core (SQL API) account must be set to consistent or lazy.
upvoted 1 times

HOTSPOT -

You have an Azure Cosmos DB Core (SQL) API 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')
```

[▼] Composite / Range / Spatial

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

[▼] Composite / Range / Spatial

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

[▼] Composite / Range / Spatial

Correct Answer:

Answer Area

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

[▼] Composite / **Range** / Spatial

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

[▼] **Composite** / Range / Spatial

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

[▼] **Composite** / Range / Spatial

Box 1: Range -

Range index is based on an ordered tree-like structure. The range index type is used for:

Equality queries:

SELECT * FROM container c WHERE c.property = 'value'

SELECT * FROM c WHERE c.property IN ("value1", "value2", "value3")

Box 2: Composite -
Composite indexes increase the efficiency when you are performing operations on multiple fields. The composite index type is used for:
Queries with a filter on two or more properties where at least one property is an equality filter
SELECT * FROM container c WHERE c.property1 = 'value' AND c.property2 > 'value'

Box 3: Composite -
Incorrect:
Spatial indices enable efficient queries on geospatial objects such as - points, lines, polygons, and multipolygon. These queries use
ST_DISTANCE, ST_WITHIN,
ST_INTERSECTS keywords.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/index-overview

essdeecee 8 months, 1 week ago
B should be range from the link supplied : https://docs.microsoft.com/en-us/azure/cosmos-db/index-overview#:~:text=Range%20queries%3A,of%20a%20property%3A
upvoted 1 times

Internal_Koala 7 months, 3 weeks ago
B is correct as composite: "Queries with a filter on two or more properties where at least one property is an equality filter"
There is an equality filter on the city name.
upvoted 4 times

Question #7 *Topic 4*

You have a container named container1 in an Azure Cosmos DB Core (SQL) API account named account1 that is set to the session default consistency level. The average size of an item in contained 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 App1, modify the connection policy settings

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 account1, change the default consistency level to bounded staleness

Correct Answer: *C*
The main goal of the integrated cache is to reduce costs for read-heavy workloads. Low latency, while helpful, is not the main benefit of the integrated cache because Azure Cosmos DB is already fast without caching.
Point reads and queries that hit the integrated cache won't use any RUs. In other words, any cache hits will have an RU charge of 0. Cache hits will have a much lower per-operation cost than reads from the backend database.
Workloads that fit the following characteristics should evaluate if the integrated cache will help lower costs:

Read-heavy workloads -
Many repeated point reads on large items

Many repeated high RU queries -

Hot partition key for reads -
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/integrated-cache

purplefish 3 weeks, 4 days ago

Selected Answer: B
Since the application performs a point read on the same set of items in the container every minute, using "consistent prefix" as the consistency level will allow the application to consume fewer RUs.
upvoted 1 times

You have an application that queries an Azure Cosmos DB Core (SQL) API account.
You discover that the following two queries run frequently.

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

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

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

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

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

C. a composite index for (name ASC, timestamp ASC)

D. a composite index for (name ASC, timestamp DESC)

Correct Answer: *C*
You should customize your indexing policy so you can serve all necessary ORDER BY queries.
ORDER BY queries on multiple properties:
The following considerations are used when using composite indexes for queries with an ORDER BY clause with two or more properties:
* The composite index also supports an ORDER BY clause with the opposite order on all paths.
* If the composite index paths do not match the sequence of the properties in the ORDER BY clause, then the composite index can't support the query.
* The order of composite index paths (ascending or descending) should also match the order in the ORDER BY clause.
Consider the following example where a composite index is defined on properties name, age, and _ts:

Composite Index	Sample `ORDER BY` Query	Supported by Composite Index?
`(name ASC, age ASC)`	`SELECT * FROM c ORDER BY c.name ASC, c.age asc`	`Yes`
`(name ASC, age ASC)`	`SELECT * FROM c ORDER BY c.age ASC, c.name asc`	`No`
`(name ASC, age ASC)`	`SELECT * FROM c ORDER BY c.name DESC, c.age DESC`	`Yes`
`(name ASC, age ASC)`	`SELECT * FROM c ORDER BY c.name ASC, c.age DESC`	`No`
`(name ASC, age ASC, timestamp ASC)`	`SELECT * FROM c ORDER BY c.name ASC, c.age ASC, timestamp ASC`	`Yes`
`(name ASC, age ASC, timestamp ASC)`	`SELECT * FROM c ORDER BY c.name ASC, c.age ASC`	`No`

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

Question #9 *Topic 4*

You have a container in an Azure Cosmos DB Core (SQL) API account.
Data update volumes are unpredictable.
You need to process the change feed 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 instance name for each instance

D. Configure a different lease container configuration for each instance

E. Configure the same instance name for all the instances

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

Correct Answer: *ACF*
A: Implementing the change feed processor.
The point of entry is always the monitored container, from a Container instance you call GetChangeFeedProcessorBuilder.
C: You define the compute instance name or unique identifier with WithInstanceName, this should be unique and different in each compute instance you are deploying.
F: The lease container: The lease container acts as a state storage and coordinates processing the change feed across multiple workers. The lease container can be stored in the same account as the monitored container or in a separate account.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/sql/change-feed-processor

purplefish 3 weeks, 4 days ago

Selected Answer: BCF
Configuring the same processor name for all instances would not allow multiple instances to share the workload, as each instance would try to grab and process the same leases. Instead, a different processor name should be configured for each instance, while using the same instance name and lease container configuration.
upvoted 1 times

Nath2 6 months, 3 weeks ago

Selected Answer: ACF
Answer is correct:
"
To take advantage of the compute distribution within the deployment unit, the only key requirements are:

All instances should have the same lease container configuration.
All instances should have the same processorName.
Each instance needs to have a different instance name (WithInstanceName).
"
From: https://learn.microsoft.com/en-us/azure/cosmos-db/nosql/change-feed-processor?tabs=dotnet#dynamic-scaling
upvoted 1 times

andicoro 7 months ago

Selected Answer: BEF
I think it is B,E,F
Sharing the lease container
You can share the lease container across multiple deployment units, each deployment unit would be listening to a different monitored container or have a different processorName. With this configuration, each deployment unit would maintain an independent state on the lease container.
Review the request unit consumption on the lease container to make sure the provisioned throughput is enough for all the deployment units.
upvoted 1 times

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 database in an Azure Cosmos DB Core (SQL) API 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 conflicts are sent to the conflicts feed.
Solution: You set ConflictResolutionMode to Custom and you use the default settings for the policy.
Does this meet the goal?

A. Yes

B. No

Correct Answer: *B*
You can create a custom conflict resolution policy but you need a stored procedure as well.
The stored procedure should use the conflictingItems parameter. conflictingItems: Array of the committed version of all items in the container that are conflicting with incomingItem on ID or any other unique index properties.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/sql/how-to-manage-conflicts

DudeWheresMyCar Highly Voted 8 months, 1 week ago

Selected Answer: A
Answer should be A. The question explicitly states that:
1. Use the Azure Cosmos DB SDK to implement the conflict resolution policy for a container
2. Conflicts are sent to the conflicts feed

There are three ways for conflict resolution with the SDK:
1. Last-Writer-Wins: Conflicts do NOT show up in conflict feed
2. Custom Conflict Resolution Policy with Stored Proc: Conflicts do NOT show up in conflict feed unlesss there's an error in your stored procedure
3. Custom Conflict Resolution Policy: Conflicts DO show up in conflict feed.

https://learn.microsoft.com/en-us/azure/cosmos-db/sql/how-to-manage-conflicts?tabs=dotnetv2%2Capi-async%2Casync#create-a-custom-conflict-resolution-policy

upvoted 5 times

Question #11 *Topic 4*

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 database in an Azure Cosmos DB Core (SQL) API 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 conflicts are sent to the conflicts feed.
Solution: You set ConflictResolutionMode to Custom. You set ResolutionProcedure to a custom stored procedure. You configure the custom stored procedure to use the conflictingItems parameter to resolve conflicts.
Does this meet the goal?

A. Yes

B. No

Correct Answer: *A*
You can create a custom conflict resolution policy using a stored procedure.
The stored procedure should use the conflictingItems parameter. conflictingItems: Array of the committed version of all items in the container that are conflicting with incomingItem on ID or any other unique index properties.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/sql/how-to-manage-conflicts

DudeWheresMyCar 8 months, 1 week ago

Selected Answer: B

Answer should be B. The question explicitly states that:
1. Use the Azure Cosmos DB SDK to implement the conflict resolution policy for a container
2. Conflicts are sent to the conflicts feed

There are three ways for conflict resolution with the SDK:
1. Last-Writer-Wins: Conflicts do NOT show up in conflict feed
2. Custom Conflict Resolution Policy with Stored Proc: Conflicts do NOT show up in conflict feed unless there's an error in your stored procedure
3. Custom Conflict Resolution Policy: Conflicts DO show up in conflict feed.

This question is asking about item #2, which does NOT show up in the conflict feed by default. Now, if there is an error in the stored procedure it would be, but I'm assuming the question isn't considering this, as it isn't stated.

https://learn.microsoft.com/en-us/azure/cosmos-db/sql/how-to-manage-conflicts?tabs=dotnetv2%2Capi-async%2Casync#create-a-custom-conflict-resolution-policy

upvoted 2 times

Question #12 *Topic 4*

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 database in an Azure Cosmos DB Core (SQL) API 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 conflicts are sent to the conflicts feed.
Solution: You set ConflictResolutionMode to LastWriterWins and you use the default settings for the policy.
Does this meet the goal?

A. Yes

B. No

Correct Answer: *A*
You can create a last-writer-wins conflict resolution policy.
These samples show how to set up a container with a last-writer-wins conflict resolution policy. The default path for last-writer-wins is the timestamp field or the _ts property. For SQL API, this may also be set to a user-defined path with a numeric type. In a conflict, the highest value wins. If the path isn't set or it's invalid, it defaults to _ts. Conflicts resolved with this policy do not show up in the conflict feed. This policy can be used by all APIs.
.NET SDK

```
DocumentCollection lwwCollection = await createClient.CreateDocumentCollectionIfNotExistsAsync(
UriFactory.CreateDatabaseUri(this.databaseName), new DocumentCollection
{
Id = this.lwwCollectionName,
ConflictResolutionPolicy = new ConflictResolutionPolicy
{
Mode = ConflictResolutionMode.LastWriterWins,
ConflictResolutionPath = "/myCustomId",
},
});
```

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

essdeecee Highly Voted 8 months, 1 week ago

Selected Answer: B

The question states "The solution must ensure that any conflicts are sent to the conflicts feed." Last-writer-wins explicitly states - "Conflicts resolved with this policy do not show up in the conflict feed. "

upvoted 8 times

TimSss Most Recent 7 months, 3 weeks ago

Selected Answer: B

What essdeecee said. LWW doesn't send anything to the conflict feed

upvoted 2 times

Question #1 *Topic 5*

HOTSPOT -

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

```
{
    "type": "Microsoft.DocumentDB/databaseAccounts/sqlDatabases/containers",
    "apiVersion": "2021-01-15",
    "name": "acct01/mydb/mycontainer",
    "properties": {
      "resource": {
        "id": "mycontainer",
        "partitionKey": {
          "paths": [
            "/companyid"
          ],
          "kind": "Hash"
        },
        "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.

NOTE: Each correct selection is worth one point.

Hot Area:

Answer Area

Statements	Yes	No
When the template is deployed, an Azure Cosmos DB account named `acct01` will be created if the account does **NOT** exist	○	○
When the template is deployed, an container named `mycontainer` in `mydb` will be created if the container does **NOT** exist	○	○
When the template is deployed, if `mycontainer` exists and has a partition key on `name`, the partition key will change to `companyid`	○	○

Correct Answer:

Answer Area

Statements	Yes	No
When the template is deployed, an Azure Cosmos DB account named `acct01` will be created if the account does **NOT** exist	○	●
When the template is deployed, an container named `mycontainer` in `mydb` will be created if the container does **NOT** exist	●	○
When the template is deployed, if `mycontainer` exists and has a partition key on `name`, the partition key will change to `companyid`	●	○

Box 1: No -

An account is not a resource in this context.

Note: In incremental mode, Resource Manager leaves unchanged resources that exist in the resource group but aren't specified in the template. Resources in the template are added to the resource group.

The Microsoft.DocumentDB databaseAccounts/sqlDatabases/containers resource type can be deployed to: Resource groups.

Box 2: Yes -

A container is a resource.

Box 3: Yes -

When redeploying an existing resource in incremental mode, all properties are reapplied. The properties aren't incrementally added. A common misunderstanding is to think properties that aren't specified in the template are left unchanged. If you don't specify certain properties, Resource Manager interprets the deployment as overwriting those values. Properties that aren't included in the template are reset to the default values. Specify all non-default values for the resource, not just the ones you're updating. The resource definition in the template always contains the final state of the resource. It can't represent a partial update to an existing resource.

Note: In incremental mode, Resource Manager leaves unchanged resources that exist in the resource group but aren't specified in the template. Resources in the template are added to the resource group.

Reference:

https://docs.microsoft.com/en-us/azure/templates/microsoft.documentdb/databaseaccounts/sqldatabases/containers

https://docs.microsoft.com/en-us/azure/azure-resource-manager/templates/deployment-modes#incremental-mode

Internal_Koala Highly Voted 7 months, 3 weeks ago

NYN

Confirming what DudeWheresMyCar said. Here is a source for this:

"Note that you can't change the partition key of a container. If you do need to change it, you need to migrate the container data to a new container with the correct key."

https://microsoft.github.io/AzureTipsAndTricks/blog/tip335.html

upvoted 6 times

DudeWheresMyCar Most Recent 8 months, 1 week ago

NYN

A partition key cannot be changed on a container. Instead a live migration must take place to a new container. Even if the ARM template tries to change it, it will fail.

upvoted 1 times

Question #2 *Topic 5*

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

B. Provisioned Throughput

C. Total Request Units

D. Document Count

Correct Answer: *B*
Use an alert which is triggered when the container or a database has exceeded the provisioned throughput limit.
Note: Provisioned throughput is the maximum amount of capacity that an application can consume from a table or index. If your application exceeds your provisioned throughput capacity on a table or index, it is subject to request throttling. Throttling prevents your application from consuming too many capacity units.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/monitor-cosmos-db

basiltomato 2 months, 2 weeks ago

Selected Answer: C
Correct answer: Total Request Units

https://learn.microsoft.com/en-us/azure/cosmos-db/create-alerts#create-an-alert-rule
"Now, you can define the logic for triggering an alert and use the chart to view trends of your Azure Cosmos DB account. The Total Request Units metric supports dimensions. These dimensions allow you to filter on the metric. For example, you can use dimensions to filter to a specific database or container you want to monitor. If you don't select any dimension, this value is ignored.

Choose StatusCode as the Dimension name. Select Add custom value and set the status code to 429."
upvoted 2 times

mkahmann 4 months, 1 week ago

Selected Answer: B
I think B is the correct answer. The link provided by avocacao is about checking whether a certain amount of requests exceeded the rate limit.

Reading the text on this page, I think the answer should be B.
https://docs.microsoft.com/en-us/azure/cosmos-db/monitor-cosmos-db
upvoted 3 times

purplefish 3 weeks, 4 days ago
I think you are right, the Provisioned Throughput metric measures the number of provisioned Request Units per second (RU/s) for a Cosmos DB account, and can be used to trigger an alert when the actual usage exceeds a threshold.
upvoted 1 times

mkahmann 4 months, 1 week ago
To be more specific:
https://learn.microsoft.com/en-us/azure/cosmos-db/monitor?tabs=azure-diagnostics#alerts
upvoted 1 times

avocacao 5 months ago

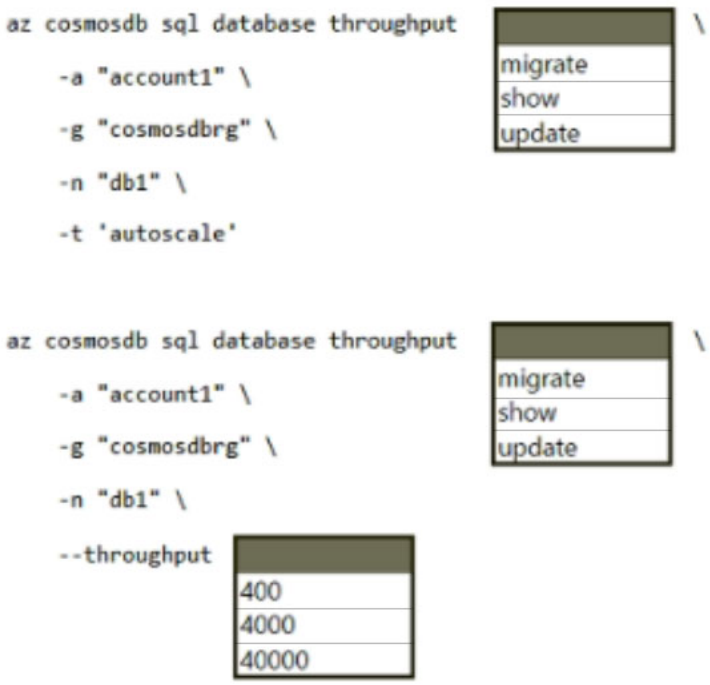
Selected Answer: C
https://learn.microsoft.com/en-us/azure/cosmos-db/create-alerts
upvoted 2 times

andicoro 7 months ago

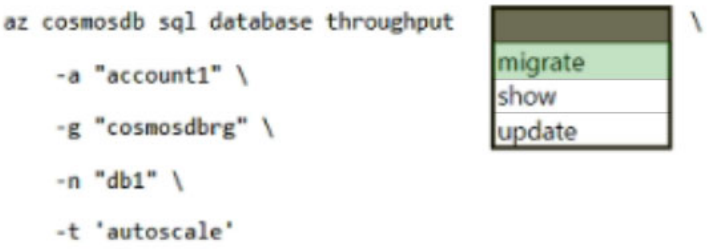
Selected Answer: C
Answer should be C for sure
upvoted 3 times

HOTSPOT -
You have a database named db1 in an Azure Cosmos DR Core (SQL) API 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:

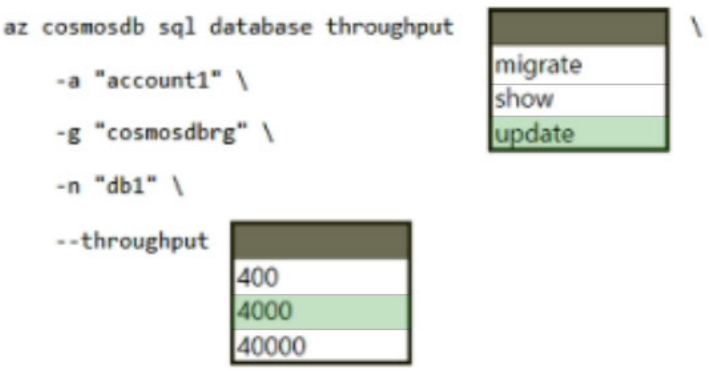
Answer Area



Answer Area



Correct Answer:



Box 1: migrate -
The az cosmosdb sql database throughput migrate command migrates the throughput of the SQL database between autoscale and manually

provisioned.
Syntax: az cosmosdb sql database throughput migrate
--account-name
--name
--resource-group
--throughput-type {autoscale, manual}

Box 2: update -
The az cosmosdb sql database throughput update command updates the throughput of the SQL database under an Azure Cosmos DB account.
Syntax: az cosmosdb sql database throughput update
--account-name
--name
--resource-group
[--max-throughput]
[--throughput]

Box 3: 4000 -
Specify the throughput.

Parameter --throughput -
The throughput of SQL database (RU/s).
Note: Example migration from standard (manual) provisioned throughput to autoscale: Suppose you have a container with 10,000 RU/s manual provisioned throughput, and 25 GB of storage. When you enable autoscale, the initial autoscale max RU/s will be: 10,000 RU/s, which will scale between 1000 - 10,000 RU/s.
Note 2: Parameter --max-throughput
The maximum throughput resource can scale to (RU/s). Provided when the resource is autoscale enabled. The minimum value can be 4000 (RU/s).
Reference:
https://docs.microsoft.com/en-us/cli/azure/cosmosdb/sql/database/throughput

TRUESON 3 weeks ago
For the second don‘t confuse throughput & max-throughput
upvoted 1 times

Juba1711 5 months, 3 weeks ago
correct
https://learn.microsoft.com/en-us/azure/cosmos-db/nosql/manage-with-cli?source=recommendations#migrate-a-database-to-autoscale-throughput
upvoted 2 times

You need to create a database in an Azure Cosmos DB Core (SQL) API 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 serverless account.

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

Correct Answer: *B*
Manual is best suited for workloads with steady or predictable traffic.
Autoscale workloads is best suited with variable or unpredictable traffic.
Note: Azure Cosmos DB allows you to set provisioned throughput on your databases and containers. There are two types of provisioned throughput, standard
(manual) or autoscale.
Autoscale provisioned throughput in Azure Cosmos DB allows you to scale the throughput (RU/s) of your database or container automatically and instantly. The throughput is scaled based on the usage, without impacting the availability, latency, throughput, or performance of the workload.
The use cases of autoscale include:
* Variable or unpredictable workloads
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/how-to-choose-offer

HOTSPOT -

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

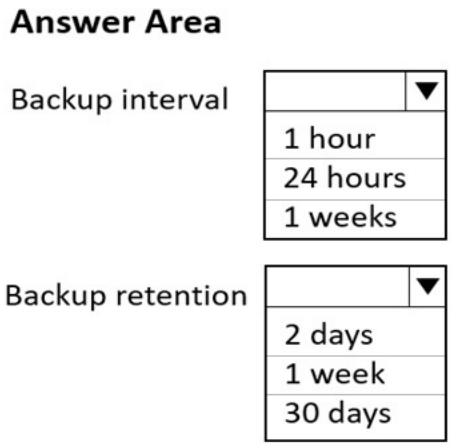
The database is modified once per day in a batch process.

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

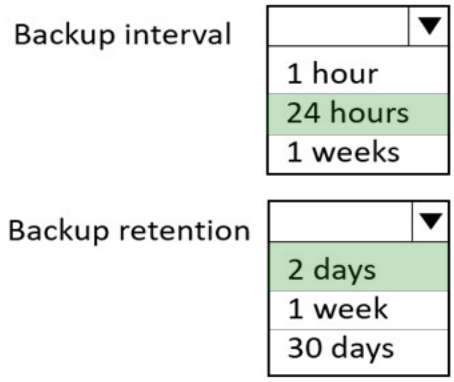
How should you configure the backup settings? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Hot Area:



Correct Answer:



HOTSPOT -

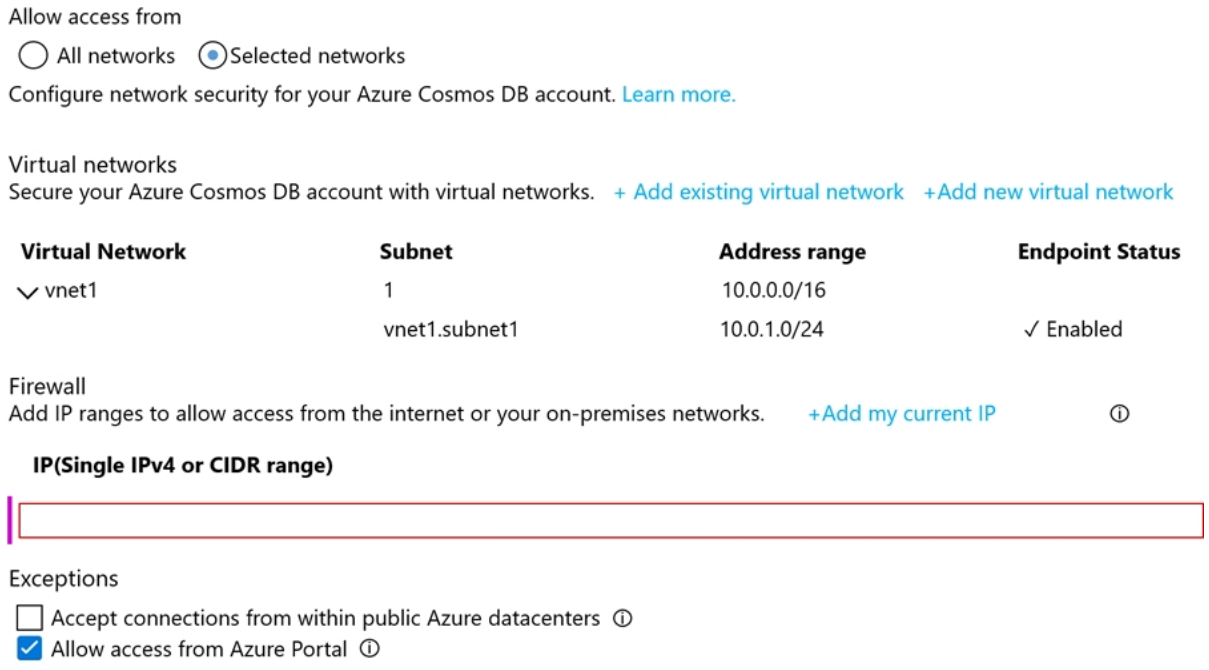
You have an Azure Cosmos DB Core (SQL) API account named account1.

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

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

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

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



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

NOTE: Each correct selection is worth one point.

Hot Area:

Answer Area

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

Answer Area

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

Correct Answer:

Box 1: Yes -
VM1 is on vnet1.subnet1 which has the Endpoint Status enabled.

Box 2: No -
Only virtual network and their subnets added to Azure Cosmos account have access. Their peered VNets cannot access the account until the subnets within peered virtual networks are added to the account.

Box 3: No -
Only virtual network and their subnets added to Azure Cosmos account have access.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/how-to-configure-vnet-service-endpoint

lakime Highly Voted 1 year ago
the only allowed network is 10.0.1.0/24 - NYN
upvoted 7 times

Gall Most Recent 8 months, 3 weeks ago
The table is not in sync with FW regards the subnets. Based on the table YNN, but based on the screenshot NYN.
upvoted 3 times

ivangh 9 months, 1 week ago
Correct! Y N N. /24 is inside /16. And only allowed subnet 1. (VM1)
upvoted 1 times

nqthien041292 1 year ago
Vote YYN
upvoted 1 times

Question #7 *Topic 5*

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

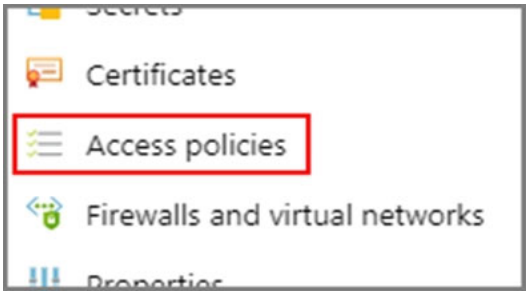
D. Update

E. Sign

F. Verify

G. Unwrap Key

Correct Answer: *ABG*
To Configure customer-managed keys for your Azure Cosmos account with Azure Key Vault:
Add an access policy to your Azure Key Vault instance:
1. From the Azure portal, go to the Azure Key Vault instance that you plan to use to host your encryption keys. Select Access Policies from the left menu:



2. Select + Add Access Policy.
3. Under the Key permissions drop-down menu, select Get, Unwrap Key, and Wrap Key permissions:

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

Juba1711 5 months, 3 weeks ago
Correct answer
https://learn.microsoft.com/en-us/azure/storage/common/customer-managed-keys-overview
upvoted 2 times

Question #8 *Topic 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: *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://<cosmosinstance-name>.documents.azure.com:443/",
"connect.cosmos.master.key": "<cosmosdbprimarykey>",
"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/

TimSss Highly Voted 7 months, 3 weeks ago

Selected Answer: ACD

We want to have data from cosmos to kafka so source, not sink

upvoted 6 times

TRUESON 3 weeks ago

source is to get cosmosdb data to kafka, sink is to write kafka data to cosmosdb
upvoted 1 times

TRUESON 3 weeks ago
for better understanding watch this video from 22:30 https://www.youtube.com/live/b9L_CTUaz5Y
upvoted 1 times

AscentAcademy Highly Voted 9 months, 1 week ago
Not sure, but shouldn't we have answer A and not F ?

Here we intent to have the AzureCosmosDB as a source to export to Kafka as a sink - meaning that we should import the "com.azure.cosmos.kafka.connect.source.CosmosDBSourceConnector" as stated here https://docs.microsoft.com/en-us/azure/cosmos-db/sql/kafka-connector-source
"Kafka Connect for Azure Cosmos DB is a connector to read from and write data to Azure Cosmos DB. The Azure Cosmos DB source connector provides the capability to read data from the Azure Cosmos DB change feed and publish this data to a Kafka topic."
upvoted 5 times

Question #9 *Topic 5*

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*
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 #10 *Topic 5*

You have a container named container1 in an Azure Cosmos DB Core (SQL) API account.
You need to provide a user named User1 with the ability to insert items into container1 by using role-based access control (RBAC). The solution must use the principle of least privilege.
Which roles should you assign to User1?

A. CosmosDB Operator only

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

C. DocumentDB Account Contributor only

D. Cosmos DB Built-in Data Contributor only

Correct Answer: *A*
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

Billy_inat Highly Voted 9 months ago
Selected Answer: D
Answer is D
upvoted 5 times

khushbu123 Most Recent 7 months, 3 weeks ago
Selected Answer: D
It should be D
upvoted 1 times

TimSss 7 months, 3 weeks ago
Like the answer text says, db operator can do much more than just insert items
upvoted 1 times

essdeecee 8 months, 1 week ago
Selected Answer: D
Supplied answer Cosmos DB Operator : Can provision Azure Cosmos accounts, databases, and containers. Cannot access any data or use Data Explorer.
Cosmos DB Built-in Data Contributor Microsoft.DocumentDB/databaseAccounts/readMetadata
Microsoft.DocumentDB/databaseAccounts/sqlDatabases/containers/*
Cosmos DB Built-in Data Contributor contains Microsoft.DocumentDB/databaseAccounts/sqlDatabases/containers/items/* for read/replace/upsert/delete
upvoted 2 times

janisk 9 months ago
Selected Answer: D
Cosmos DB Built-in Data Contributor
https://docs.microsoft.com/en-us/azure/cosmos-db/how-to-setup-rbac
upvoted 4 times

lakime 1 year ago
Selected Answer: C
DocumentDB Account Contributor as
Cosmos DB Operator Can provision Azure Cosmos accounts, databases, and containers. Cannot access any data or use Data Explorer.
upvoted 2 times

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*
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

Internal_Koala Highly Voted 6 months, 4 weeks ago

Selected Answer: D
Vote for D as per the linked website's example:

```
AzureDiagnostics
| where Category == "ControlPlaneRequests"
| where OperationName startswith "SqlContainersThroughputUpdate"
```

upvoted 6 times

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?

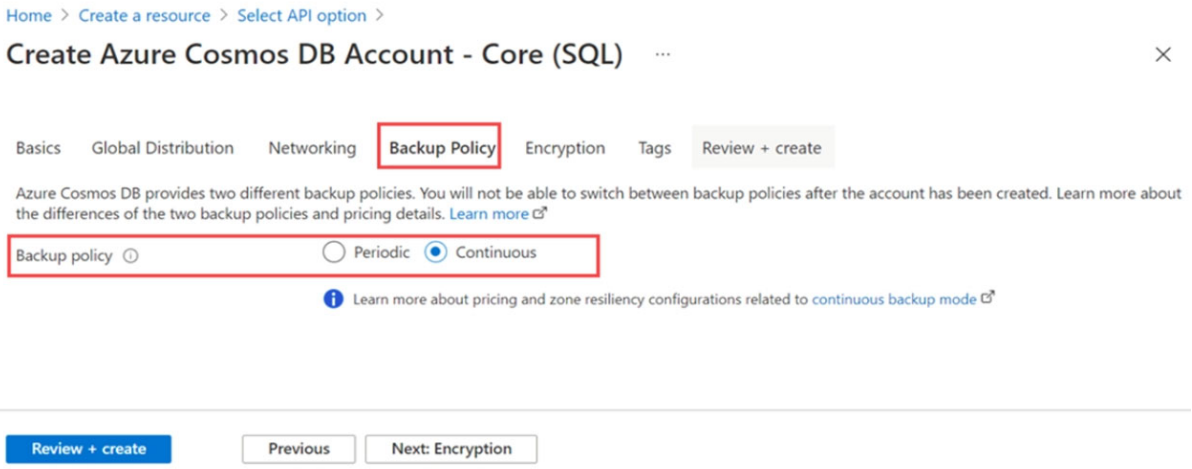
A. Enable Continuous Backup for the account.

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*
When creating a new Azure Cosmos DB account, in the Backup policy tab, choose continuous mode to enable the point in time restore functionality for the new account. With the point-in-time restore, data is restored to a new account, currently you can't restore to an existing account.



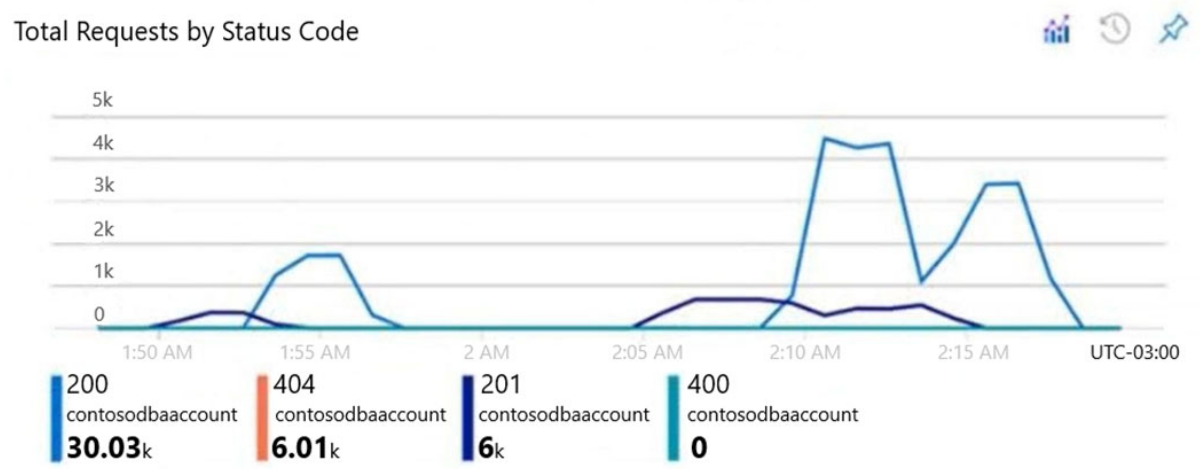
Provision an Azure Cosmos DB account with continuous backup configuration.

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

HOTSPOT -

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

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



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

NOTE: Each correct selection is worth one point.

Hot Area:

Answer Area

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

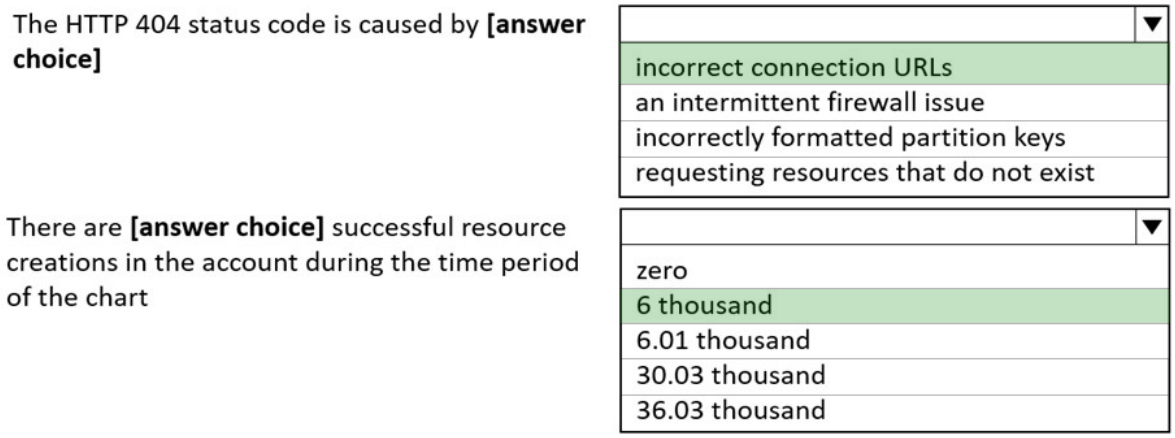
- incorrect connection URLs
- an intermittent firewall issue
- incorrectly formatted partition keys
- requesting resources that do not exist

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

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

Correct Answer:

Answer Area



Box 1: incorrect connection URLs

400 Bad Request: Returned when there is an error in the request URI, headers, or body. The response body will contain an error message explaining what the specific problem is.

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

Box 2: 6 thousand -

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

Note:

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

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

Reference:

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

lakime Highly Voted 1 year ago

404 Not Found: Returned when a resource does not exist on the server. If you are managing or querying an index, check the syntax and verify the index name is specified correctly.
- so first answer should be D
201 Created: Success on PUT or POST. Object created or updated successfully.
So second is correct

upvoted 11 times

Ranzzzan 2 months, 3 weeks ago

100% correct

upvoted 1 times

essdeecee 8 months, 1 week ago

404 | Not Found | The document is no longer a resource, that is, the document was deleted.
https://docs.microsoft.com/en-us/training/modules/monitor-responses-events-azure-cosmos-db-sql-api/2-review-common-response-codes#:~:text=Strong%20or%20Bounded.-,404,is%20no%20longer%20a%20resource%2C%20that%20is%2C%20the%20document%20was%20deleted.,-Replace%20a%20Document

upvoted 2 times

TimSss Most Recent 7 months, 3 weeks ago

Explanation is talking about 400 but question is about 404

upvoted 2 times

essdeecee 8 months, 1 week ago

2nd answer is 30.3: looking for 200 codes. Answer selection is for 404 codes which is not correct.

upvoted 1 times

Internal_Koala 6 months, 4 weeks ago

200 is for succesful GET while 201 is a successful creation. Second choice (201) is correct.

upvoted 2 times

Question #14 *Topic 5*

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*
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

HOTSPOT -

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

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

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

NOTE: Each correct selection is worth one point.

Hot Area:

Answer Area

development: [ ]
- Serverless capacity mode
- Provisioned throughput capacity mode and manual throughput
- Provisioned throughput capacity mode and autoscale throughput

shipments: [ ]
- Serverless capacity mode
- Provisioned throughput capacity mode and manual throughput
- Provisioned throughput capacity mode and autoscale throughput

Correct Answer:

Answer Area

development: [Serverless capacity mode]
- Serverless capacity mode
- Provisioned throughput capacity mode and manual throughput
- Provisioned throughput capacity mode and autoscale throughput

shipments: [Provisioned throughput capacity mode and autoscale throughput]
- Serverless capacity mode
- Provisioned throughput capacity mode and manual throughput
- Provisioned throughput capacity mode and autoscale throughput

Box 1: Serverless capacity mode -

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

- Getting started with Azure Cosmos DB
- Running applications with bursty, intermittent traffic that is hard to forecast, or low (<10%) average-to-peak traffic ratio
- Developing, testing, prototyping and running in production new applications where the traffic pattern is unknown
- Integrating with serverless compute services like Azure Functions

Box 2: Provisioned throughput capacity mode and autoscale throughput

The use cases of autoscale include:

- Variable or unpredictable workloads: When your workloads have variable or unpredictable spikes in usage, autoscale helps by automatically

scaling up and down based on usage. Examples include retail websites that have different traffic patterns depending on seasonality; IOT workloads that have spikes at various times during the day; line of business applications that see peak usage a few times a month or year, and more. With autoscale, you no longer need to manually provision for peak or average capacity.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/serverless
https://docs.microsoft.com/en-us/azure/cosmos-db/provision-throughput-autoscale#use-cases-of-autoscale

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	○	○
The query uses an index	○	○
Recreating the container with the partition key set to /categoryId will improve the performance of the query	○	○

Correct Answer:

Answer Area

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

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

TRUESON 2 weeks, 5 days ago

To correctly answer the question one would have to know the partionkey https://learn.microsoft.com/en-us/azure/cosmos-db/nosql/how-to-query-container

upvoted 1 times

Question #17 *Topic 5*

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.

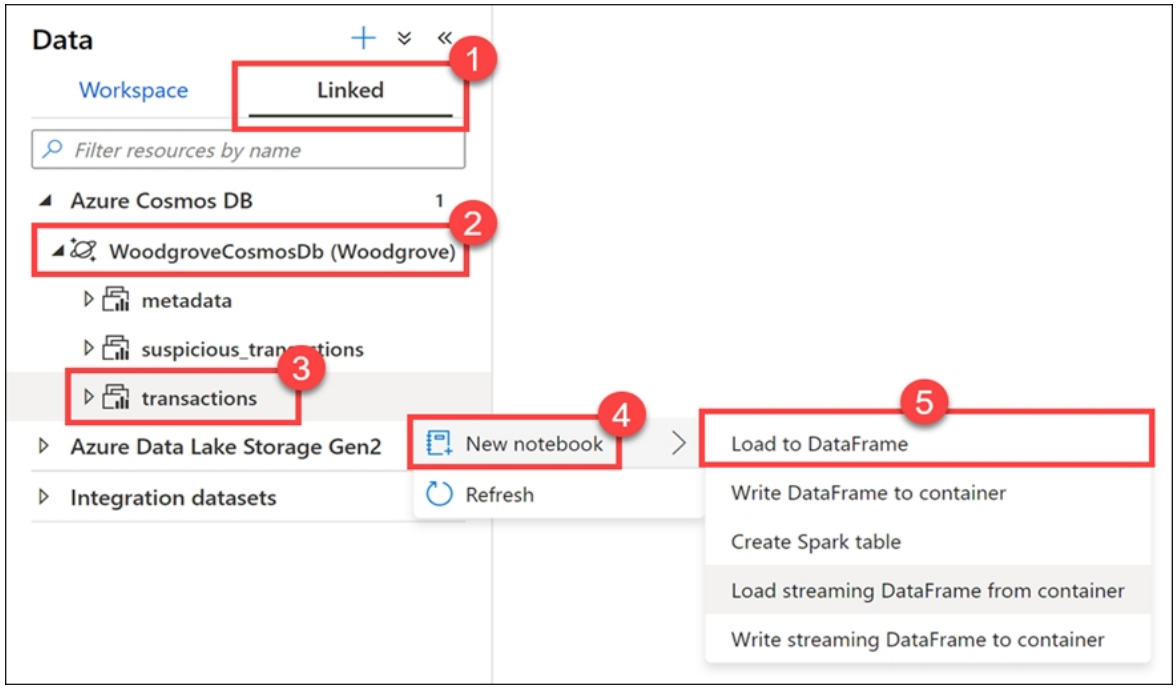
B. Create a private endpoint connection to the account.

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

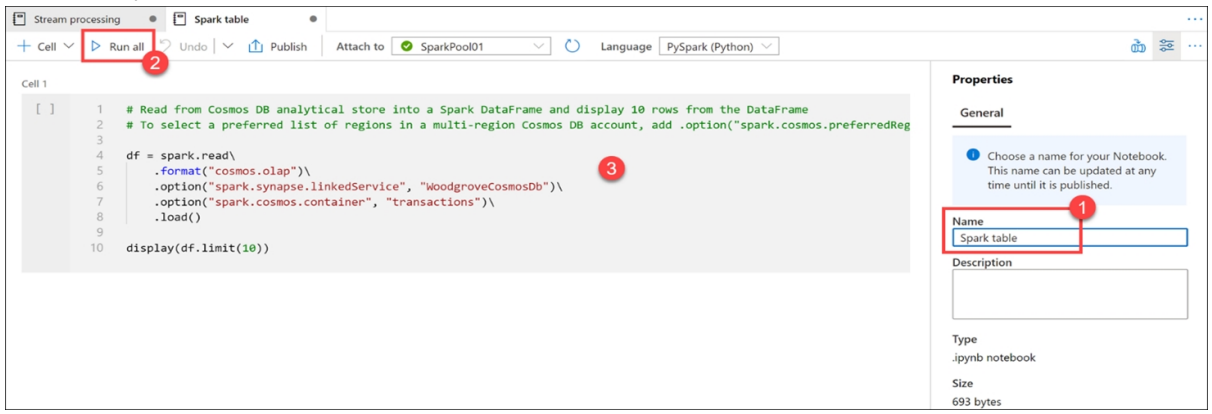
D. Enable Azure Synapse Link for the account and Analytical store on the container.

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

Correct Answer: *AD*
Explore analytical store with Apache Spark
1. Navigate to the Data hub.
2. Select the Linked tab (1), expand the Azure Cosmos DB group (if you don't see this, select the Refresh button above), then expand the WoodgroveCosmosDb account (2). Right-click on the transactions container (3), select New notebook (4), then select Load to DataFrame (5).



3. In the generated code within Cell 1 (3), notice that the spark.read format is set to cosmos.olap. This instructs Synapse Link to use the container's analytical store. If we wanted to connect to the transactional store, like to read from the change feed or write to the container, we'd use cosmos.oltp instead.



Reference:

https://github.com/microsoft/MCW-Cosmos-DB-Real-Time-Advanced-Analytics/blob/main/Hands-on%20lab/HOL%20step-by%20step%20-%20Cosmos%20DB%20real-time%20advanced%20analytics.md

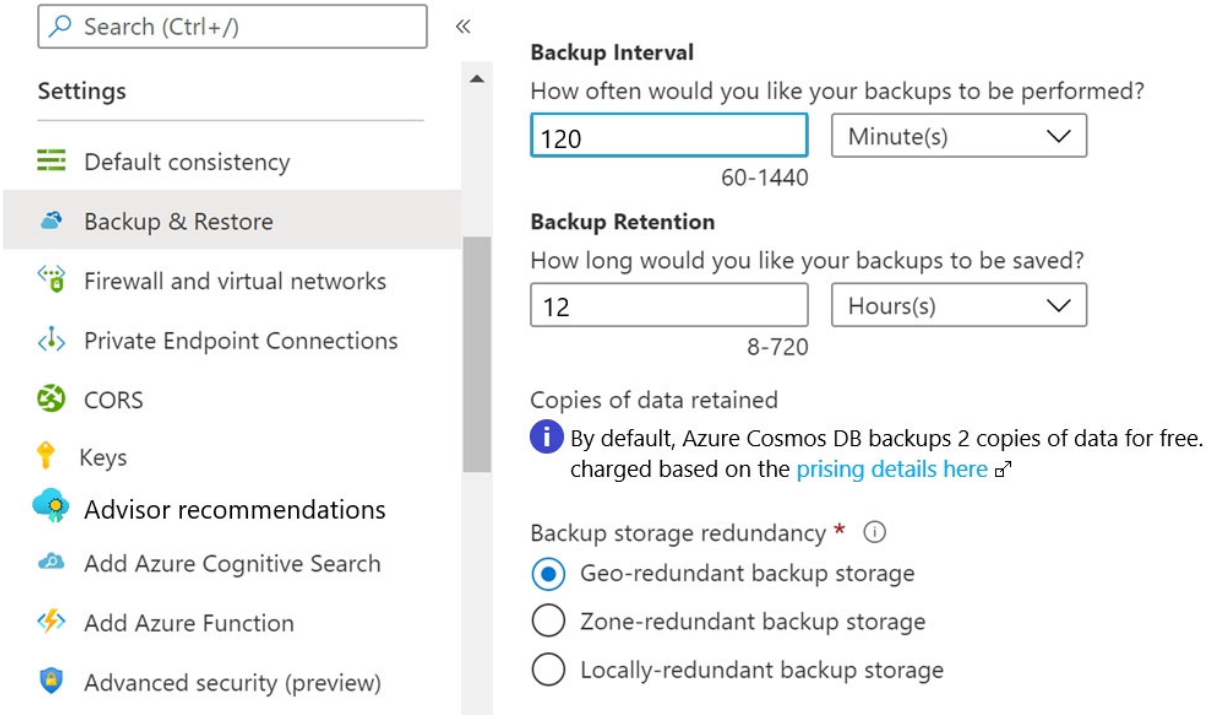
TRUESON 3 weeks ago

Might be correct https://learn.microsoft.com/en-us/azure/synapse-analytics/synapse-link/how-to-query-analytical-store-spark

upvoted 1 times

HOTSPOT -

You configure a backup for an Azure Cosmos DB Core (SQL) API account as shown in the following exhibit.



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

NOTE: Each correct selection is worth one point.

Hot Area:

Answer Area

The current backup interval and retention policy provide protection for **[answer choice]**

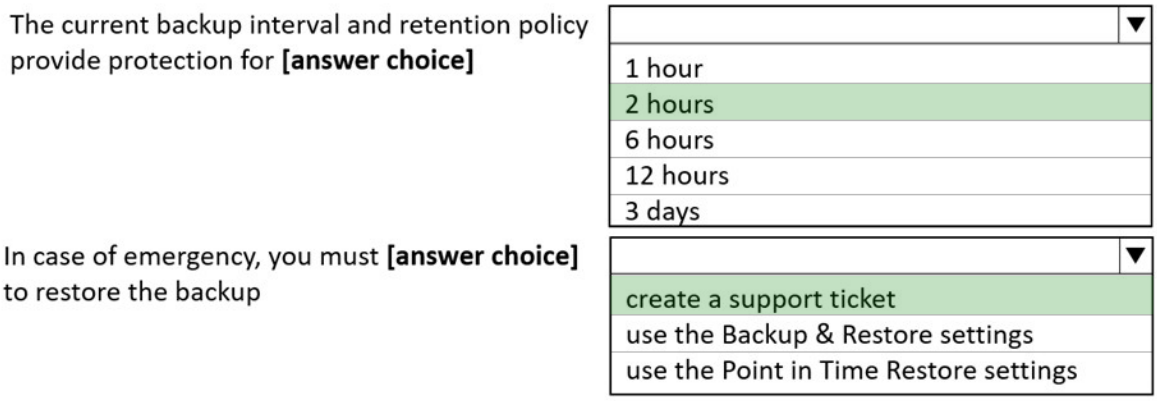
- 1 hour
- 2 hours
- 6 hours
- 12 hours
- 3 days

In case of emergency, you must **[answer choice]** to restore the backup

- create a support ticket
- use the Backup & Restore settings
- use the Point in Time Restore settings

Correct Answer:

Answer Area



Box 1: 2 hours -
RPO is 2 hours.
Note: Recovery Point Objective (RPO) is a measure of how frequently you take backups. If a disaster occurs between backups, can you afford to lose five minutes' worth of data updates? Or five hours? Or a full day? RPO represents how fresh recovered data will be. In practice, the RPO indicates the amount of data
(updated or created) that will be lost or need to be reentered after an outage.

Box 2: create a support ticket -
Request data restore from a backup
If you accidentally delete your database or a container, you can file a support ticket or call the Azure support to restore the data from automatic online backups.
Note: With Azure Cosmos DB SQL API accounts, you can also maintain your own backups by using one of the following approaches:
Use Azure Data Factory to move data periodically to a storage of your choice.
Use Azure Cosmos DB change feed to read data periodically for full backups or for incremental changes, and store it in your own storage.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/configure-periodic-backup-restore

Internal_Koala Highly Voted 7 months, 3 weeks ago

RPO is 2 hours, correct, but this is not what is asked for, right!? If I set the interval to 12 hrs, then I have a RPO of 4 hours, but this is obviously a lower level of protection. Correct answer should be 12 hrs. My last backup will be deleted after 12 hours as per retention policy.

upvoted 6 times

Lyarra Most Recent 7 months, 2 weeks ago

Wouldn't it provide protection for 6 hours? Backups retention is set for 12 hours, but Azure only keeps 2 backup copies by default.

upvoted 1 times

Alex22022 3 months, 3 weeks ago

The number of copies stored depends on the interval and the retention period.
The default retention backup intervall is 240 minutes (4h) and the retention period is 8h, thats why 2 copies are stored by default. The first 2 copies are free as well.

In this example, every 2h a copy is created and is kept for 12h. This means there can be up to 6 copies. 4 of the 6 copies are billed according to Azure blob storage pricing.

Thus, the correct answer is 12h.

This article explains that well: https://learn.microsoft.com/en-us/training/modules/implement-backup-restore-for-azure-cosmos-db-sql-api/2-evaluate-periodic-backup

upvoted 1 times

TimSss 7 months, 3 weeks ago

I have 12 hours of backup, which is updated every 2 hours.

upvoted 1 times

TimSss 7 months, 3 weeks ago

so protection is 12h

upvoted 2 times

Question #19 *Topic 5*

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

B. Metadata Requests

C. Region Removed

D. Region Added

Correct Answer: *B*
Metric Metadata Requests: Count of metadata requests. Azure Cosmos DB maintains system metadata container for each account, that allows you to enumerate collections, databases, etc., and their configurations, free of charge.
Used to monitor throttles due to metadata requests.
Incorrect:
Not A: Metric Data Usage: Total data usage reported at 5-minutes granularity per region. Used to monitor total data usage at container and region, minimum granularity should be 5 minutes.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/monitor-cosmos-db-reference#metrics

Question #20 *Topic 5*

You have an Azure Cosmos DB Core (SQL) API account.
You need to create an Azure Monitor query that lists recent modifications to the regional failover policy.
Which Azure Monitor table should you query?

A. CDBControlPlaneRequests

B. CDBQueryRunTimeStatistics

C. CDBPartitionKeyStatistics

D. CDBDataPlaneRequests

Correct Answer: *A*
The CDBControlPlaneRequests table details all control plane operations executed on the account, which include modifications to the regional failover policy, indexing policy, IAM role assignments, backup/restore policies, VNet and firewall rules, private links as well as updates and deletes of the account.
Reference:
https://docs.microsoft.com/en-us/azure/azure-monitor/reference/tables/cdbcontrolplanerequests

IngeTahiti 5 months, 3 weeks ago

Selected Answer: A
CDBControlPlaneRequests -> This table details all control plane operations executed on the account, which include modifications to the regional failover policy, indexing policy, IAM role assignments, backup/restore policies, VNet and firewall rules, private links as well as updates and deletes of the account.
upvoted 3 times

Juba1711 5 months, 3 weeks ago

answer is D
https://learn.microsoft.com/en-us/azure/azure-monitor/reference/tables/cdbdataplanerequests
upvoted 1 times

TRUESON 3 weeks ago

The question is not about operations executed to create, update, delete or retrieve data within the account
upvoted 1 times

HOTSPOT -

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

```
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();
```

Click the Java tab to see the encryption policy in Java.)

```
ClientEncryptionIncludedPath path1 = new ClientEncryptionIncludedPath();
path1.path = "/creditcard";
path1.clientEncryptionKeyId = "encryptionkey";
path1.encryptionType = CosmosEncryptionType.RANDOMIZED;
path1.encryptionAlgorithm = CosmosEncryptionAlgorithm.AEAD_AES_256_CBC_HMAC_SHA256;

ClientEncryptionIncludedPath path2 = new ClientEncryptionIncludedPath();
path1.path = "/SSN";
path2.clientEncryptionKeyId = "encryptionkey";
path2.encryptionType = EncryptionType.DETERMINISTIC;
path2.encryptionAlgorithm = CosmosEncryptionAlgorithm.AEAD_AES_256_CBC_HMAC_SHA256;

List<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:

Answer Area

Statements	Yes	No
You can perform that filters on the `creditcard` property	◯	◯
You can perform a query that filters on the `SSN` property	◯	◯
An application can be allowed to read the `creditcard` property while being restricted from reading the `SSN` property	◯	◯

Correct Answer:

Answer Area

Statements	Yes	No
You can perform that filters on the `creditcard` property	○	◉
You can perform a query that filters on the `SSN` property	◉	○
An application can be allowed to read the `creditcard` property while being restricted from reading the `SSN` property	◉	○

Box 1: No -
The creditcard property uses randomized encryption.
Randomized encryption is more secure, but prevents queries from filtering on encrypted properties.

Box 2: Yes -
The SSN property uses deterministic encryption.
Using deterministic encryption allows queries to perform equality filters on encrypted properties.

Box 3: Yes -
Reading documents when only a subset of properties can be decrypted.
In situations where the client does not have access to all the CMK used to encrypt properties, only a subset of properties can be decrypted when data is read back. For example, if property1 was encrypted with key1 and property2 was encrypted with key2, a client application that only has access to key1 can still read data, but not property2. In such a case, you must read your data through SQL queries and project away the properties that the client can't decrypt: SELECT c.property1, c.property3 FROM c.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/how-to-always-encrypted

Question #1 *Topic 6*

Introductory Info

Case Study -

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

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

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

To start the case study -

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

Overview -

Litware, Inc. is a United States-based grocery retailer. Litware has a main office and a primary datacenter in Seattle. The company has 50 retail stores across the

United States and an emerging -

online presence. Each store connects directly to the internet.

Existing environment. Cloud and Data Service Environments.

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

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

Each container in productdb is configured for manual throughput.

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

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

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

SELECT * FROM con-productVendor pv

ORDER BY pv.creditRating, pv.yearFounded

Existing environment. Current Problems.

Litware identifies the following issues:

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

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

Requirements. Planned Changes -
Litware plans to implement a new Azure Cosmos DB Core (SQL) API account named account2 that will contain a database named iotdb. The iotdb database will contain two containers named con-iot1 and con-iot2.
Litware plans to make the following changes:
Store the telemetry data in account2.
Configure account1 to support multiple read-write regions.
Implement referential integrity for the con-product container.
Use Azure Functions to send notifications about product updates to different recipients.
Develop an app named App1 that will run from all locations and query the data in account1.
Develop an app named App2 that will run from the retail stores and query the data in account2. App2 must be limited to a single DNS endpoint when accessing account2.
Requirements. Business Requirements
Litware identifies the following business requirements:
Whenever there are multiple solutions for a requirement, select the solution that provides the best performance, as long as there are no additional costs associated.
Ensure that Azure Cosmos DB costs for IoT-related processing are predictable.
Minimize the number of firewall changes in the retail stores.
Requirements. IoT Telemetry Requirements
Litware identifies the following IoT telemetry requirements:
Write the telemetry data from streamanalytics1 to the con-iot1 container as the data arrives. Use streamanalytics1 to create hopping window aggregated telemetry readings once per hour and write the data to con-iot2.
Automatically delete items in con-iot1 after one hour unless a per-item time to live is set.
Optimize the partitioning of con-iot1 for queries that support historical data of a specific device.
Aggregate the telemetry data by device by using streamanalytics1 and write the aggregate data to con-iot2.
Ensure that the items in con-iot2 persist regardless of the per-item time to live setting.
Requirements. Product Catalog Requirements
Litware identifies the following requirements for the product catalog:
Implement a custom conflict resolution policy for the product catalog data.
Minimize the frequency of errors during updates of the con-product container.
Once multi-region writes are configured, maximize the performance of App1 queries against the data in account1.
Trigger the execution of two Azure functions following every update to any document in the con-product container.

Question

You are troubleshooting the current issues caused by the application updates.
Which action can address the application updates issue without affecting the functionality of the application?

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

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

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

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

Correct Answer: *C*
Bounded staleness is frequently chosen by globally distributed applications that expect low write latencies but require total global order guarantee. Bounded staleness is great for applications featuring group collaboration and sharing, stock ticker, publish-subscribe/queueing etc.
Scenario: Application updates in con-product frequently cause HTTP status code 429 "Too many requests". You discover that the 429 status code relates to excessive request unit (RU) consumption during the updates.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/consistency-levels

DudeWheresMyCar Highly Voted 8 months ago

Selected Answer: D

For 429's on updating documents it is recommended to create a custom index policy, and only contain the properties that are needed.

https://learn.microsoft.com/en-us/azure/cosmos-db/sql/troubleshoot-request-rate-too-large?tabs=resource-specific#429s-on-create-replace-or-upsert-document-requests

upvoted 6 times

Introductory Info

Case Study -

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

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

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

To start the case study -

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

Overview -

Litware, Inc. is a United States-based grocery retailer. Litware has a main office and a primary datacenter in Seattle. The company has 50 retail stores across the

United States and an emerging -

online presence. Each store connects directly to the internet.

Existing environment. Cloud and Data Service Environments.

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

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

Each container in productdb is configured for manual throughput.

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

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

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

SELECT * FROM con-productVendor pv

ORDER BY pv.creditRating, pv.yearFounded

Existing environment. Current Problems.

Litware identifies the following issues:

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

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

Requirements. Planned Changes -

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

Litware plans to make the following changes:
Store the telemetry data in account2.
Configure account1 to support multiple read-write regions.
Implement referential integrity for the con-product container.
Use Azure Functions to send notifications about product updates to different recipients.
Develop an app named App1 that will run from all locations and query the data in account1.
Develop an app named App2 that will run from the retail stores and query the data in account2. App2 must be limited to a single DNS endpoint when accessing account2.
Requirements. Business Requirements
Litware identifies the following business requirements:
Whenever there are multiple solutions for a requirement, select the solution that provides the best performance, as long as there are no additional costs associated.
Ensure that Azure Cosmos DB costs for IoT-related processing are predictable.
Minimize the number of firewall changes in the retail stores.
Requirements. IoT Telemetry Requirements
Litware identifies the following IoT telemetry requirements:
Write the telemetry data from streamanalytics1 to the con-iot1 container as the data arrives. Use streamanalytics1 to create hopping window aggregated telemetry readings once per hour and write the data to con-iot2.
Automatically delete items in con-iot1 after one hour unless a per-item time to live is set.
Optimize the partitioning of con-iot1 for queries that support historical data of a specific device.
Aggregate the telemetry data by device by using streamanalytics1 and write the aggregate data to con-iot2.
Ensure that the items in con-iot2 persist regardless of the per-item time to live setting.
Requirements. Product Catalog Requirements
Litware identifies the following requirements for the product catalog:
Implement a custom conflict resolution policy for the product catalog data.
Minimize the frequency of errors during updates of the con-product container.
Once multi-region writes are configured, maximize the performance of App1 queries against the data in account1.
Trigger the execution of two Azure functions following every update to any document in the con-product container.

Question

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

A. the timestamp

B. the humidity

C. the temperature

D. the device ID

Correct Answer: *D*
The partition key is what will determine how data is routed in the various partitions by Cosmos DB and needs to make sense in the context of your specific scenario. The IoT Device ID is generally the "natural" partition key for IoT applications.
Scenario: The iotdb database will contain two containers named con-iot1 and con-iot2.
Ensure that Azure Cosmos DB costs for IoT-related processing are predictable.
Reference:
https://docs.microsoft.com/en-us/azure/architecture/solution-ideas/articles/iot-using-cosmos-db

TimSss 7 months, 3 weeks ago
correct
upvoted 3 times

Introductory Info

Case Study -

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

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

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

To start the case study -

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

Overview -

Litware, Inc. is a United States-based grocery retailer. Litware has a main office and a primary datacenter in Seattle. The company has 50 retail stores across the

United States and an emerging -

online presence. Each store connects directly to the internet.

Existing environment. Cloud and Data Service Environments.

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

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

Each container in productdb is configured for manual throughput.

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

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

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

SELECT * FROM con-productVendor pv

ORDER BY pv.creditRating, pv.yearFounded

Existing environment. Current Problems.

Litware identifies the following issues:

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

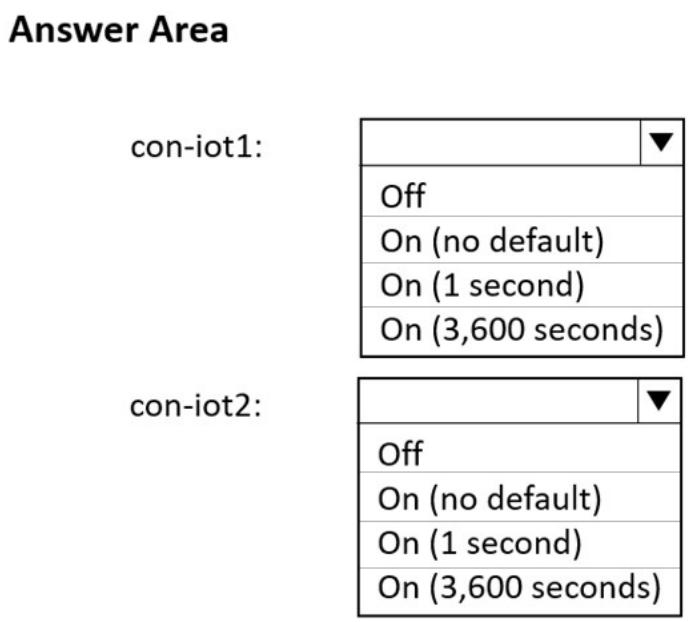
Application updates in con-product frequently cause HTTP status code 429 "Too many requests". You discover that the 429 status code relates to excessive request unit (RU) consumption during the updates.

Requirements. Planned Changes -

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

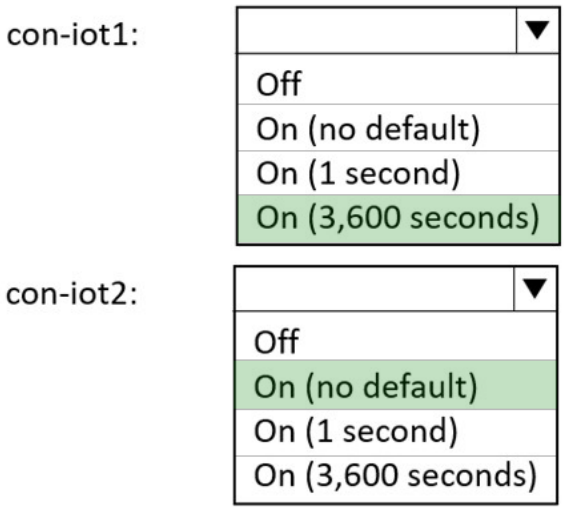
Litware plans to make the following changes:
Store the telemetry data in account2.
Configure account1 to support multiple read-write regions.
Implement referential integrity for the con-product container.
Use Azure Functions to send notifications about product updates to different recipients.
Develop an app named App1 that will run from all locations and query the data in account1.
Develop an app named App2 that will run from the retail stores and query the data in account2. App2 must be limited to a single DNS endpoint when accessing account2.
Requirements. Business Requirements
Litware identifies the following business requirements:
Whenever there are multiple solutions for a requirement, select the solution that provides the best performance, as long as there are no additional costs associated.
Ensure that Azure Cosmos DB costs for IoT-related processing are predictable.
Minimize the number of firewall changes in the retail stores.
Requirements. IoT Telemetry Requirements
Litware identifies the following IoT telemetry requirements:
Write the telemetry data from streamanalytics1 to the con-iot1 container as the data arrives. Use streamanalytics1 to create hopping window aggregated telemetry readings once per hour and write the data to con-iot2.
Automatically delete items in con-iot1 after one hour unless a per-item time to live is set.
Optimize the partitioning of con-iot1 for queries that support historical data of a specific device.
Aggregate the telemetry data by device by using streamanalytics1 and write the aggregate data to con-iot2.
Ensure that the items in con-iot2 persist regardless of the per-item time to live setting.
Requirements. Product Catalog Requirements
Litware identifies the following requirements for the product catalog:
Implement a custom conflict resolution policy for the product catalog data.
Minimize the frequency of errors during updates of the con-product container.
Once multi-region writes are configured, maximize the performance of App1 queries against the data in account1.
Trigger the execution of two Azure functions following every update to any document in the con-product container.

Question
HOTSPOT -
You plan to implement con-iot1 and con-iot2.
You need to configure the default Time to Live setting for each container. The solution must meet the IoT telemetry requirements.
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

Correct Answer:



Box 1: On (3,600 seconds)

Note: Litware identifies the following IoT telemetry requirements:

Automatically delete items in con-iot1 after one hour unless a per-item time to live is set.

Box 2: On (no default)

Time to Live on a container (set using DefaultTimeToLive):

* If missing (or set to null), items are not expired automatically.
* If present and the value is set to "-1", it is equal to infinity, and items don't expire by default.
* If present and the value is set to some non-zero number "n" λ€" items will expire "n" seconds after their last modified time.

Note: Litware identifies the following IoT telemetry requirements:

Ensure that the items in con-iot2 persist regardless of the per-item time to live setting.

Reference:

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

DudeWheresMyCar Highly Voted 8 months ago

Answer two should be "Off" due to:
Ensure that the items in con-iot2 persist regardless of the per-item time to live setting.

upvoted 11 times

susejzepol Most Recent 5 months, 3 weeks ago

I think that the answer two must be "off" because the case says: "ensure that the items in con-iot2 persist regardless of per-item time to live setting.

upvoted 2 times

Introductory Info

Case Study -

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

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

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

To start the case study -

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

Overview -

Litware, Inc. is a United States-based grocery retailer. Litware has a main office and a primary datacenter in Seattle. The company has 50 retail stores across the

United States and an emerging -

online presence. Each store connects directly to the internet.

Existing environment. Cloud and Data Service Environments.

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

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

Each container in productdb is configured for manual throughput.

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

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

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

SELECT * FROM con-productVendor pv

ORDER BY pv.creditRating, pv.yearFounded

Existing environment. Current Problems.

Litware identifies the following issues:

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

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

Requirements. Planned Changes -

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

Litware plans to make the following changes:
Store the telemetry data in account2.
Configure account1 to support multiple read-write regions.
Implement referential integrity for the con-product container.
Use Azure Functions to send notifications about product updates to different recipients.
Develop an app named App1 that will run from all locations and query the data in account1.
Develop an app named App2 that will run from the retail stores and query the data in account2. App2 must be limited to a single DNS endpoint when accessing account2.
Requirements. Business Requirements
Litware identifies the following business requirements:
Whenever there are multiple solutions for a requirement, select the solution that provides the best performance, as long as there are no additional costs associated.
Ensure that Azure Cosmos DB costs for IoT-related processing are predictable.
Minimize the number of firewall changes in the retail stores.
Requirements. IoT Telemetry Requirements
Litware identifies the following IoT telemetry requirements:
Write the telemetry data from streamanalytics1 to the con-iot1 container as the data arrives. Use streamanalytics1 to create hopping window aggregated telemetry readings once per hour and write the data to con-iot2.
Automatically delete items in con-iot1 after one hour unless a per-item time to live is set.
Optimize the partitioning of con-iot1 for queries that support historical data of a specific device.
Aggregate the telemetry data by device by using streamanalytics1 and write the aggregate data to con-iot2.
Ensure that the items in con-iot2 persist regardless of the per-item time to live setting.
Requirements. Product Catalog Requirements
Litware identifies the following requirements for the product catalog:
Implement a custom conflict resolution policy for the product catalog data.
Minimize the frequency of errors during updates of the con-product container.
Once multi-region writes are configured, maximize the performance of App1 queries against the data in account1.
Trigger the execution of two Azure functions following every update to any document in the con-product container.

Question
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:

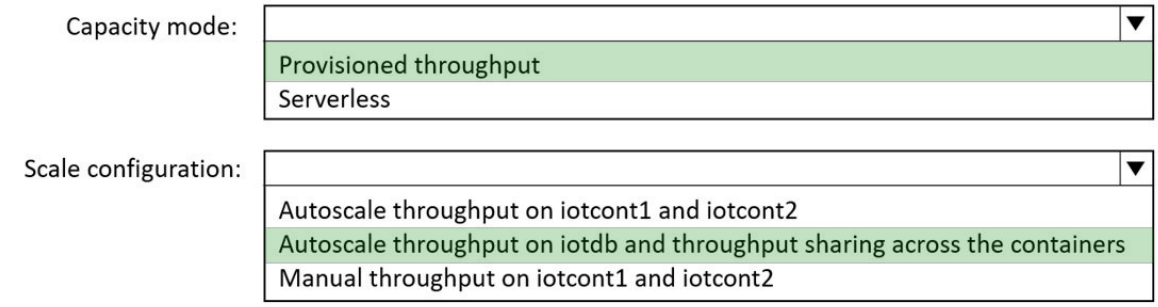
Answer Area

Capacity mode:
- Provisioned throughput
- Serverless

Scale configuration:
- Autoscale throughput on iotcont1 and iotcont2
- Autoscale throughput on iotdb and throughput sharing across the containers
- Manual throughput on iotcont1 and iotcont2

Correct Answer:

Answer Area



Box 1: Provisioned throughput -

Provisioned throughput is best suited for workloads with sustained traffic requiring predictable performance.

Billing model: Billing is done on a per-hour basis for the RU/s provisioned, regardless of how many RUs were consumed.

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

Note: Azure Cosmos DB is available in two different capacity modes: provisioned throughput and serverless. You can perform the exact same database operations in both modes, but the way you get billed for these operations is radically different.

Box 2: Autoscale throughput on iotdb and throughput sharing across the containers

You can enable autoscale on a single container, or provision autoscale throughput on a database and share it among all the containers in the database.

Cost-effective: Autoscale helps optimize your RU/s usage and cost usage by scaling down when not in use. You only pay for the resources that your workloads need on a per-hour basis.

Incorrect: cannot autoscale on two containers.

Note: Autoscale provisioned throughput in Azure Cosmos DB allows you to scale the throughput (RU/s) of your database or container automatically and instantly.

Reference:

https://docs.microsoft.com/en-us/azure/cosmos-db/throughput-serverless https://docs.microsoft.com/en-us/azure/cosmos-db/sql/how-to-provision-autoscale-throughput

essdeecee Highly Voted 8 months, 1 week ago

Manual Scale seems most appropriate:
Workloads with steady or predictable traffic (5k telemetry per second is predictable) https://docs.microsoft.com/en-us/azure/cosmos-db/how-to-choose-offer#:~:text=You%20have%20high%2C%20consistent%20utilization%20of%20provisioned%20RU/s.

upvoted 6 times

Question #1 *Topic 7*

Introductory Info

Case Study -

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

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

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

To start the case study -

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

Overview -

Litware, Inc. is a United States-based grocery retailer. Litware has a main office and a primary datacenter in Seattle. The company has 50 retail stores across the

United States and an emerging -

online presence. Each store connects directly to the internet.

Existing environment. Cloud and Data Service Environments.

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

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

Each container in productdb is configured for manual throughput.

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

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

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

SELECT * FROM con-productVendor pv

ORDER BY pv.creditRating, pv.yearFounded

Existing environment. Current Problems.

Litware identifies the following issues:

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

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

Requirements. Planned Changes -
Litware plans to implement a new Azure Cosmos DB Core (SQL) API account named account2 that will contain a database named iotdb. The iotdb database will contain two containers named con-iot1 and con-iot2.
Litware plans to make the following changes:
Store the telemetry data in account2.
Configure account1 to support multiple read-write regions.
Implement referential integrity for the con-product container.
Use Azure Functions to send notifications about product updates to different recipients.
Develop an app named App1 that will run from all locations and query the data in account1.
Develop an app named App2 that will run from the retail stores and query the data in account2. App2 must be limited to a single DNS endpoint when accessing account2.
Requirements. Business Requirements
Litware identifies the following business requirements:
Whenever there are multiple solutions for a requirement, select the solution that provides the best performance, as long as there are no additional costs associated.
Ensure that Azure Cosmos DB costs for IoT-related processing are predictable.
Minimize the number of firewall changes in the retail stores.
Requirements. IoT Telemetry Requirements
Litware identifies the following IoT telemetry requirements:
Write the telemetry data from streamanalytics1 to the con-iot1 container as the data arrives. Use streamanalytics1 to create hopping window aggregated telemetry readings once per hour and write the data to con-iot2.
Automatically delete items in con-iot1 after one hour unless a per-item time to live is set.
Optimize the partitioning of con-iot1 for queries that support historical data of a specific device.
Aggregate the telemetry data by device by using streamanalytics1 and write the aggregate data to con-iot2.
Ensure that the items in con-iot2 persist regardless of the per-item time to live setting.
Requirements. Product Catalog Requirements
Litware identifies the following requirements for the product catalog:
Implement a custom conflict resolution policy for the product catalog data.
Minimize the frequency of errors during updates of the con-product container.
Once multi-region writes are configured, maximize the performance of App1 queries against the data in account1.
Trigger the execution of two Azure functions following every update to any document in the con-product container.

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

A. Direct mode over HTTPS

B. Gateway mode (using HTTPS)

C. Direct mode over TCP

Correct Answer: *C*
Scenario: Develop an app named App2 that will run from the retail stores and query the data in account2. App2 must be limited to a single DNS endpoint when accessing account2.
By using Azure Private Link, you can connect to an Azure Cosmos account via a private endpoint. The private endpoint is a set of private IP addresses in a subnet within your virtual network.
When you're using Private Link with an Azure Cosmos account through a direct mode connection, you can use only the TCP protocol. The HTTP protocol is not currently supported.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/how-to-configure-private-endpoints

Bharat 4 months ago

Selected Answer: B

Gateway mode meets the requirements. Here is the link: https://learn.microsoft.com/en-us/azure/cosmos-db/nosql/sdk-connection-modes

upvoted 4 times

Introductory Info

Case Study -

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

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

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

To start the case study -

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

Overview -

Litware, Inc. is a United States-based grocery retailer. Litware has a main office and a primary datacenter in Seattle. The company has 50 retail stores across the

United States and an emerging -

online presence. Each store connects directly to the internet.

Existing environment. Cloud and Data Service Environments.

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

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

Each container in productdb is configured for manual throughput.

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

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

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

SELECT * FROM con-productVendor pv

ORDER BY pv.creditRating, pv.yearFounded

Existing environment. Current Problems.

Litware identifies the following issues:

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

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

Requirements. Planned Changes -

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

Litware plans to make the following changes:
Store the telemetry data in account2.
Configure account1 to support multiple read-write regions.
Implement referential integrity for the con-product container.
Use Azure Functions to send notifications about product updates to different recipients.
Develop an app named App1 that will run from all locations and query the data in account1.
Develop an app named App2 that will run from the retail stores and query the data in account2. App2 must be limited to a single DNS endpoint when accessing account2.
Requirements. Business Requirements
Litware identifies the following business requirements:
Whenever there are multiple solutions for a requirement, select the solution that provides the best performance, as long as there are no additional costs associated.
Ensure that Azure Cosmos DB costs for IoT-related processing are predictable.
Minimize the number of firewall changes in the retail stores.
Requirements. IoT Telemetry Requirements
Litware identifies the following IoT telemetry requirements:
Write the telemetry data from streamanalytics1 to the con-iot1 container as the data arrives. Use streamanalytics1 to create hopping window aggregated telemetry readings once per hour and write the data to con-iot2.
Automatically delete items in con-iot1 after one hour unless a per-item time to live is set.
Optimize the partitioning of con-iot1 for queries that support historical data of a specific device.
Aggregate the telemetry data by device by using streamanalytics1 and write the aggregate data to con-iot2.
Ensure that the items in con-iot2 persist regardless of the per-item time to live setting.
Requirements. Product Catalog Requirements
Litware identifies the following requirements for the product catalog:
Implement a custom conflict resolution policy for the product catalog data.
Minimize the frequency of errors during updates of the con-product container.
Once multi-region writes are configured, maximize the performance of App1 queries against the data in account1.
Trigger the execution of two Azure functions following every update to any document in the con-product container.

Question

You need to implement a solution for the planned changes and meet the product catalog requirements.
What should you do to implement the conflict resolution policy?

A. Set the default consistency level for account1 to eventual

B. Remove frequently changed fields from the index policy of the con-product container

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

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

Correct Answer: *A*
Eventual consistency is the weakest form of consistency because a client may read the values that are older than the ones it had read before. Eventual consistency is ideal where the application does not require any ordering guarantees. Examples include count of Retweets, Likes, or non-threaded comments
Note:
The con-product container stores the company's product catalog data
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.
Implement a custom conflict resolution policy for the product catalog data.
Minimize the frequency of errors during updates of the con-product container.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/consistency-levels

josch123 Highly Voted 6 months, 4 weeks ago
Correct solution should be C, I think:
The requirements say: "Implement a custom conflict resolution policy for the product catalog data."
At https://learn.microsoft.com/en-us/azure/cosmos-db/conflict-resolution-policies
there is a note: "Custom conflict resolution policy is available only for API for NoSQL accounts and can be set only at creation time. It is not

possible to set a custom resolution policy on an existing container."
Therefore we need to create a new container if we want to have a custom resolution policy.

upvoted 5 times

Question #1 *Topic 8*

Introductory Info

Case Study -

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

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

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

To start the case study -

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

Overview -

Litware, Inc. is a United States-based grocery retailer. Litware has a main office and a primary datacenter in Seattle. The company has 50 retail stores across the

United States and an emerging -

online presence. Each store connects directly to the internet.

Existing environment. Cloud and Data Service Environments.

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

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

Each container in productdb is configured for manual throughput.

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

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

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

SELECT * FROM con-productVendor pv

ORDER BY pv.creditRating, pv.yearFounded

Existing environment. Current Problems.

Litware identifies the following issues:

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

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

Requirements. Planned Changes -
Litware plans to implement a new Azure Cosmos DB Core (SQL) API account named account2 that will contain a database named iotdb. The iotdb database will contain two containers named con-iot1 and con-iot2.
Litware plans to make the following changes:
Store the telemetry data in account2.
Configure account1 to support multiple read-write regions.
Implement referential integrity for the con-product container.
Use Azure Functions to send notifications about product updates to different recipients.
Develop an app named App1 that will run from all locations and query the data in account1.
Develop an app named App2 that will run from the retail stores and query the data in account2. App2 must be limited to a single DNS endpoint when accessing account2.
Requirements. Business Requirements
Litware identifies the following business requirements:
Whenever there are multiple solutions for a requirement, select the solution that provides the best performance, as long as there are no additional costs associated.
Ensure that Azure Cosmos DB costs for IoT-related processing are predictable.
Minimize the number of firewall changes in the retail stores.
Requirements. IoT Telemetry Requirements
Litware identifies the following IoT telemetry requirements:
Write the telemetry data from streamanalytics1 to the con-iot1 container as the data arrives. Use streamanalytics1 to create hopping window aggregated telemetry readings once per hour and write the data to con-iot2.
Automatically delete items in con-iot1 after one hour unless a per-item time to live is set.
Optimize the partitioning of con-iot1 for queries that support historical data of a specific device.
Aggregate the telemetry data by device by using streamanalytics1 and write the aggregate data to con-iot2.
Ensure that the items in con-iot2 persist regardless of the per-item time to live setting.
Requirements. Product Catalog Requirements
Litware identifies the following requirements for the product catalog:
Implement a custom conflict resolution policy for the product catalog data.
Minimize the frequency of errors during updates of the con-product container.
Once multi-region writes are configured, maximize the performance of App1 queries against the data in account1.
Trigger the execution of two Azure functions following every update to any document in the con-product container.

Question
You configure multi-region writes for account1.
You need to ensure that App1 supports the new configuration for account1. The solution must meet the business requirements and the product catalog requirements.
What should you do?

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

B. Create a private endpoint connection.

C. Modify the connection policy of App1.

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

Correct Answer: *D*
App1 queries the con-product and con-productVendor containers.
Note: Request unit is a performance currency abstracting the system resources such as CPU, IOPS, and memory that are required to perform the database operations supported by Azure Cosmos DB.
Scenario:
Develop an app named App1 that will run from all locations and query the data in account1.
Once multi-region writes are configured, maximize the performance of App1 queries against the data in account1.
Whenever there are multiple solutions for a requirement, select the solution that provides the best performance, as long as there are no additional costs associated.
Incorrect Answers:
A:
Bounded staleness relates to writes. App1 only do reads.
Note: Bounded staleness is frequently chosen by globally distributed applications that expect low write latencies but require total global order

guarantee. Bounded staleness is great for applications featuring group collaboration and sharing, stock ticker, publish-subscribe/queueing etc.
Reference:
https://docs.microsoft.com/en-us/azure/cosmos-db/consistency-levels

Introductory Info

Case Study -

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

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

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

To start the case study -

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

Overview -

Litware, Inc. is a United States-based grocery retailer. Litware has a main office and a primary datacenter in Seattle. The company has 50 retail stores across the

United States and an emerging -

online presence. Each store connects directly to the internet.

Existing environment. Cloud and Data Service Environments.

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

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

Each container in productdb is configured for manual throughput.

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

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

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

SELECT * FROM con-productVendor pv

ORDER BY pv.creditRating, pv.yearFounded

Existing environment. Current Problems.

Litware identifies the following issues:

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

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

Requirements. Planned Changes -

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

Litware plans to make the following changes:
Store the telemetry data in account2.
Configure account1 to support multiple read-write regions.
Implement referential integrity for the con-product container.
Use Azure Functions to send notifications about product updates to different recipients.
Develop an app named App1 that will run from all locations and query the data in account1.
Develop an app named App2 that will run from the retail stores and query the data in account2. App2 must be limited to a single DNS endpoint when accessing account2.
Requirements. Business Requirements
Litware identifies the following business requirements:
Whenever there are multiple solutions for a requirement, select the solution that provides the best performance, as long as there are no additional costs associated.
Ensure that Azure Cosmos DB costs for IoT-related processing are predictable.
Minimize the number of firewall changes in the retail stores.
Requirements. IoT Telemetry Requirements
Litware identifies the following IoT telemetry requirements:
Write the telemetry data from streamanalytics1 to the con-iot1 container as the data arrives. Use streamanalytics1 to create hopping window aggregated telemetry readings once per hour and write the data to con-iot2.
Automatically delete items in con-iot1 after one hour unless a per-item time to live is set.
Optimize the partitioning of con-iot1 for queries that support historical data of a specific device.
Aggregate the telemetry data by device by using streamanalytics1 and write the aggregate data to con-iot2.
Ensure that the items in con-iot2 persist regardless of the per-item time to live setting.
Requirements. Product Catalog Requirements
Litware identifies the following requirements for the product catalog:
Implement a custom conflict resolution policy for the product catalog data.
Minimize the frequency of errors during updates of the con-product container.
Once multi-region writes are configured, maximize the performance of App1 queries against the data in account1.
Trigger the execution of two Azure functions following every update to any document in the con-product container.

Question
You need to provide a solution for the Azure Functions notifications following updates to con-product. The solution must meet the business requirements and the product catalog requirements.
Which two actions should you perform? Each correct answer presents part of the solution.
NOTE: Each correct selection is worth one point.

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

B. Configure the trigger for each function to use the same 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: *AB*
leaseCollectionPrefix: when set, the value is added as a prefix to the leases created in the Lease collection for this Function. Using a prefix allows two separate
Azure Functions to share the same Lease collection by using different prefixes.
Scenario: Use Azure Functions to send notifications about product updates to different recipients.
Trigger the execution of two Azure functions following every update to any document in the con-product container.
Reference:
https://docs.microsoft.com/en-us/azure/azure-functions/functions-bindings-cosmosdb-v2-trigger

Introductory Info

Case Study -

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

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

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

To start the case study -

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

Overview -

Litware, Inc. is a United States-based grocery retailer. Litware has a main office and a primary datacenter in Seattle. The company has 50 retail stores across the

United States and an emerging -

online presence. Each store connects directly to the internet.

Existing environment. Cloud and Data Service Environments.

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

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

Each container in productdb is configured for manual throughput.

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

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

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

SELECT * FROM con-productVendor pv

ORDER BY pv.creditRating, pv.yearFounded

Existing environment. Current Problems.

Litware identifies the following issues:

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

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

Requirements. Planned Changes -

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

Litware plans to make the following changes:
Store the telemetry data in account2.
Configure account1 to support multiple read-write regions.
Implement referential integrity for the con-product container.
Use Azure Functions to send notifications about product updates to different recipients.
Develop an app named App1 that will run from all locations and query the data in account1.
Develop an app named App2 that will run from the retail stores and query the data in account2. App2 must be limited to a single DNS endpoint when accessing account2.
Requirements. Business Requirements
Litware identifies the following business requirements:
Whenever there are multiple solutions for a requirement, select the solution that provides the best performance, as long as there are no additional costs associated.
Ensure that Azure Cosmos DB costs for IoT-related processing are predictable.
Minimize the number of firewall changes in the retail stores.
Requirements. IoT Telemetry Requirements
Litware identifies the following IoT telemetry requirements:
Write the telemetry data from streamanalytics1 to the con-iot1 container as the data arrives. Use streamanalytics1 to create hopping window aggregated telemetry readings once per hour and write the data to con-iot2.
Automatically delete items in con-iot1 after one hour unless a per-item time to live is set.
Optimize the partitioning of con-iot1 for queries that support historical data of a specific device.
Aggregate the telemetry data by device by using streamanalytics1 and write the aggregate data to con-iot2.
Ensure that the items in con-iot2 persist regardless of the per-item time to live setting.
Requirements. Product Catalog Requirements
Litware identifies the following requirements for the product catalog:
Implement a custom conflict resolution policy for the product catalog data.
Minimize the frequency of errors during updates of the con-product container.
Once multi-region writes are configured, maximize the performance of App1 queries against the data in account1.
Trigger the execution of two Azure functions following every update to any document in the con-product container.

Question
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:

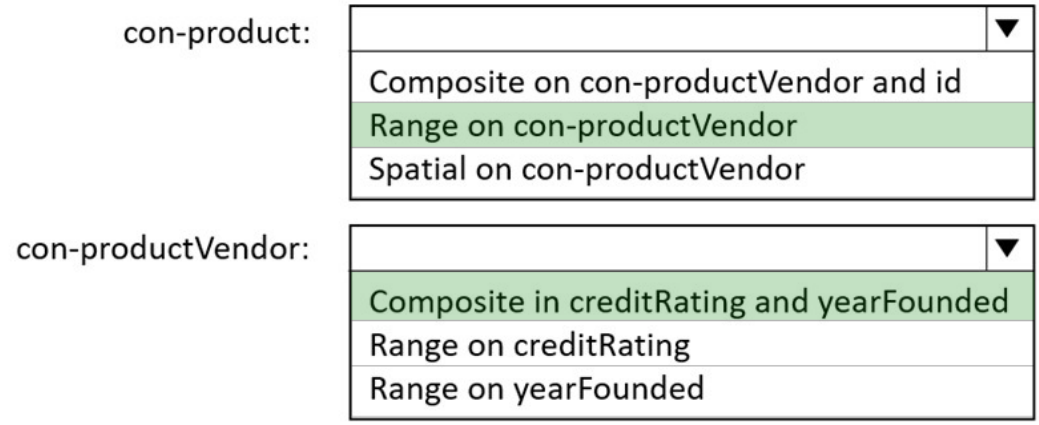
Answer Area

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

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

Correct Answer:

Answer Area



Box 1: Range on con-productVendor

Range index is based on an ordered tree-like structure. The range index type is used for:

Equality queries:

SELECT * FROM container c WHERE c.property = 'value'

SELECT * FROM c WHERE c.property IN ("value1", "value2", "value3")

Note:

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'

Box 2: Composite in creditRating and yearFounded

Composite indexes increase the efficiency when you are performing operations on multiple fields. The composite index type is used for:

ORDER BY queries on multiple properties:

SELECT * FROM container c ORDER BY c.property1, c.property2

Note: 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

Reference:

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