DP-420$^{\text{Q\&As}}$

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

Pass Microsoft DP-420 Exam with 100% Guarantee

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

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

100% Passing Guarantee
100% Money Back Assurance

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

QUESTION 1

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

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

What should you use to group the items?

A. logical partitions

B. physical partitions

C. databases

D. containers

Correct Answer: A

QUESTION 2

HOTSPOT

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

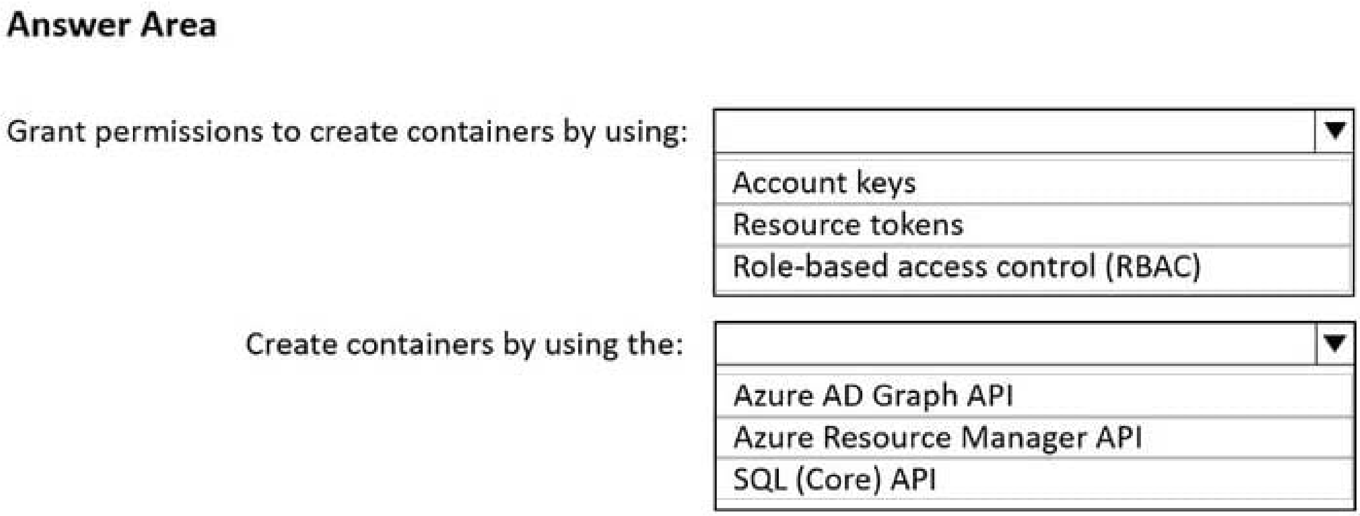
You are developing an app named App1 that will be used by a user named DevUser1 to create containers in account1. DevUser1 has a non-privileged user account in the Azure Active Directory (Azure AD) tenant.

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:



Answer Area

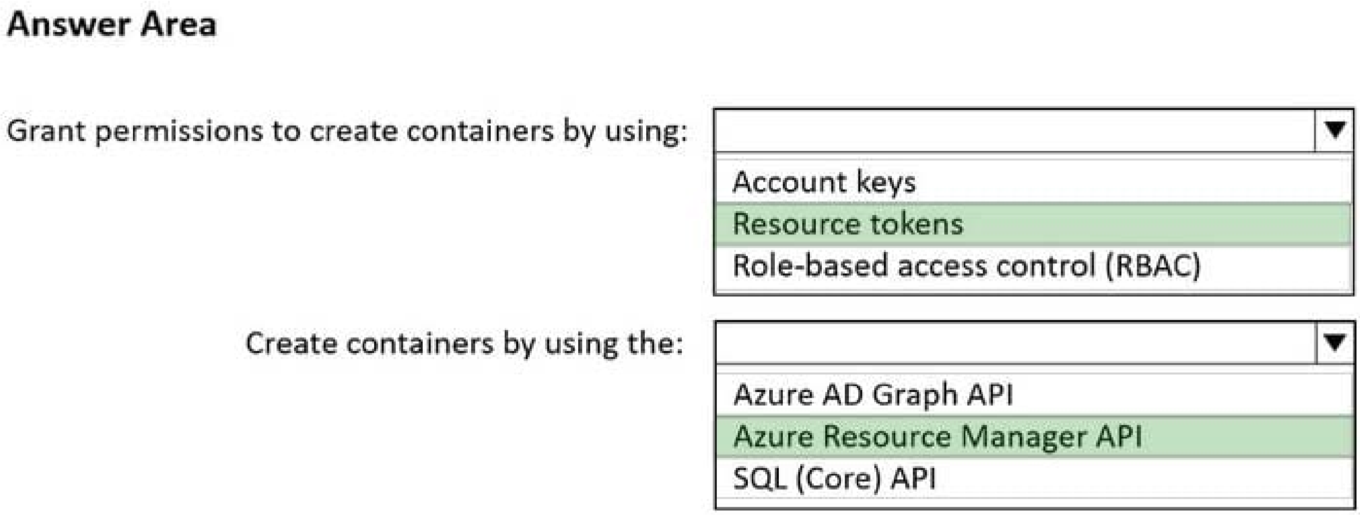
Grant permissions to create containers by using:

- Account keys
- Resource tokens
- Role-based access control (RBAC)

Create containers by using the:

- Azure AD Graph API
- Azure Resource Manager API
- SQL (Core) API

Correct Answer:



Answer Area

Grant permissions to create containers by using:

- Account keys
- [✓ Resource tokens]
- Role-based access control (RBAC)

Create containers by using the:

- Azure AD Graph API
- [✓ Azure Resource Manager API]
- SQL (Core) API

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/

QUESTION 3

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:

1.

RegionAddStart, RegionAddComplete

2.

RegionRemoveStart, RegionRemoveComplete

3.

AccountDeleteStart, AccountDeleteComplete

4.

RegionFailoverStart, RegionFailoverComplete

5.

AccountCreateStart, AccountCreateComplete

6.

AccountUpdateStart, AccountUpdateComplete

7.

VirtualNetworkDeleteStart, VirtualNetworkDeleteComplete

8.

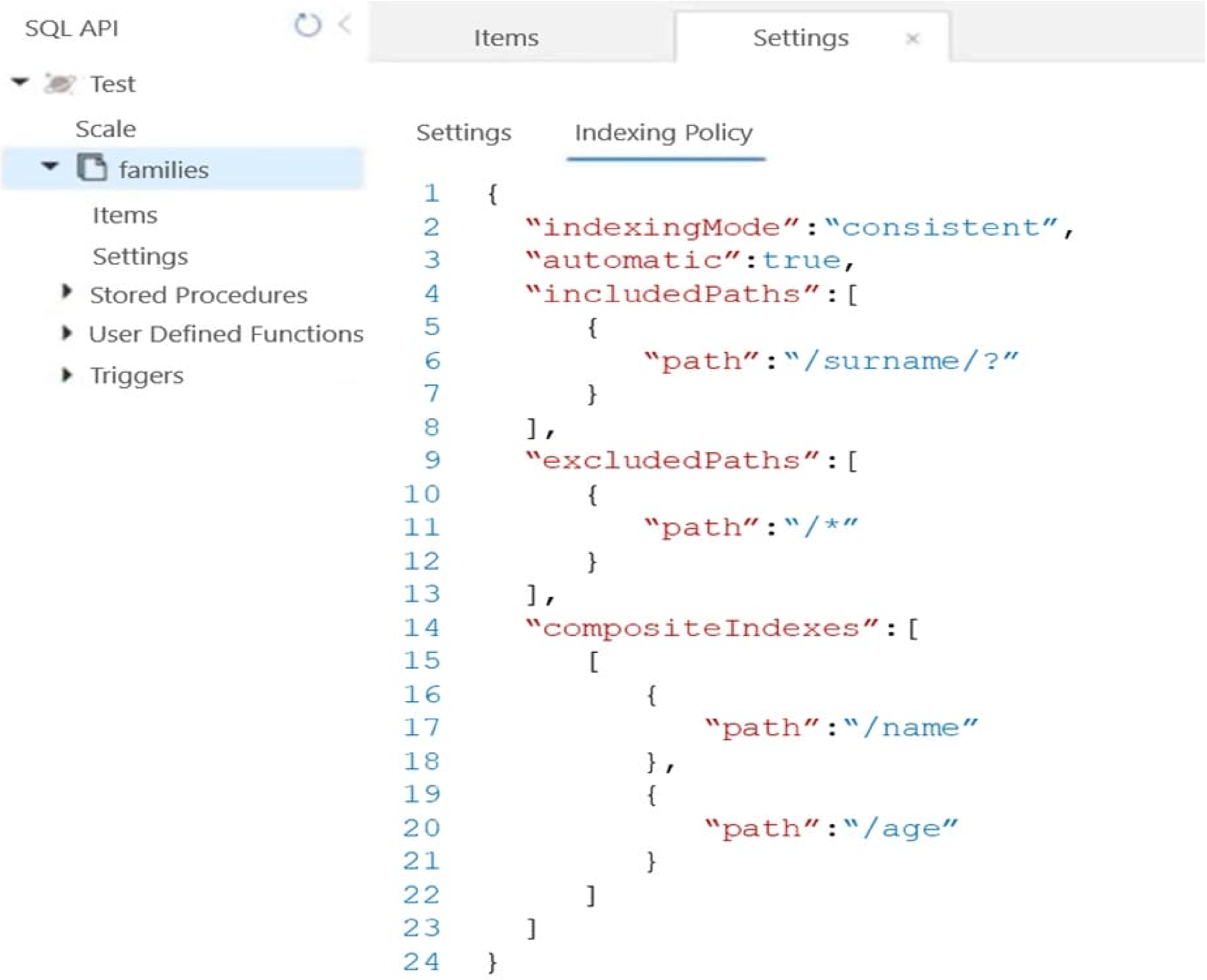
DiagnosticLogUpdateStart, DiagnosticLogUpdateComplete

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

QUESTION 4

HOTSPOT

You have the indexing policy shown in the following exhibit.

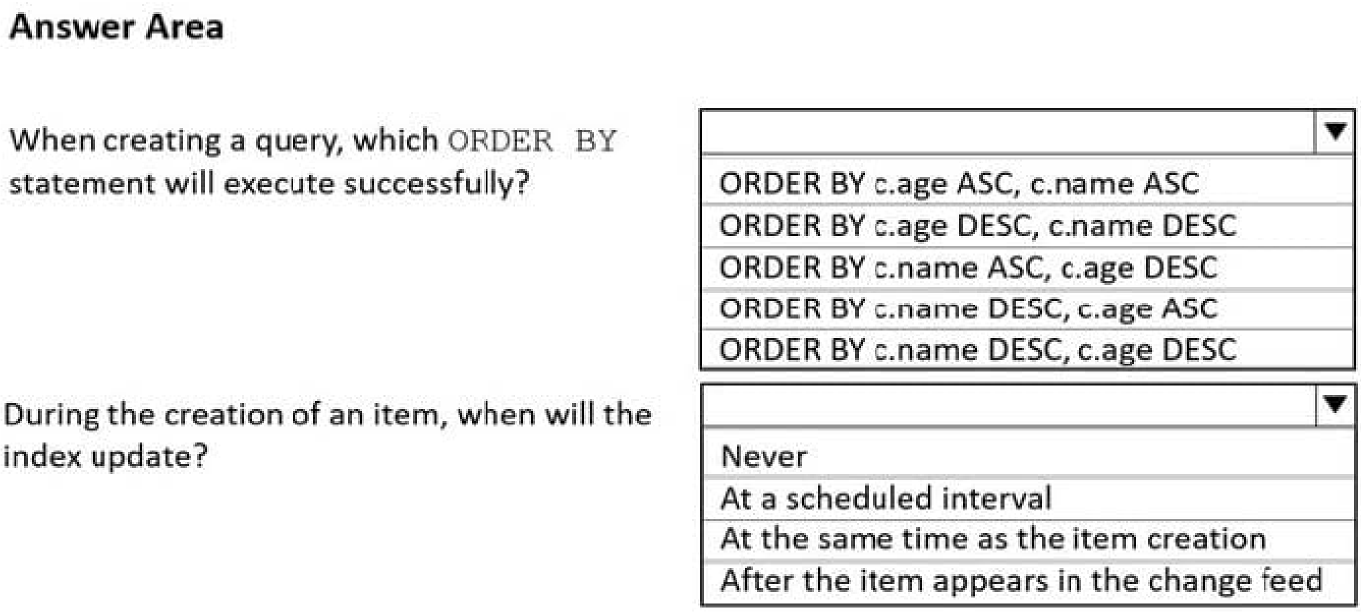


```
{
    “indexingMode”:“consistent”,
    “automatic”:true,
    “includedPaths”:[
        {
            “path”:“/surname/?”
        }
    ],
    “excludedPaths”:[
        {
            “path”:“/*”
        }
    ],
    “compositeIndexes”:[
        [
            {
                “path”:“/name”
            },
            {
                “path”:“/age”
            }
        ]
    ]
}
```

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

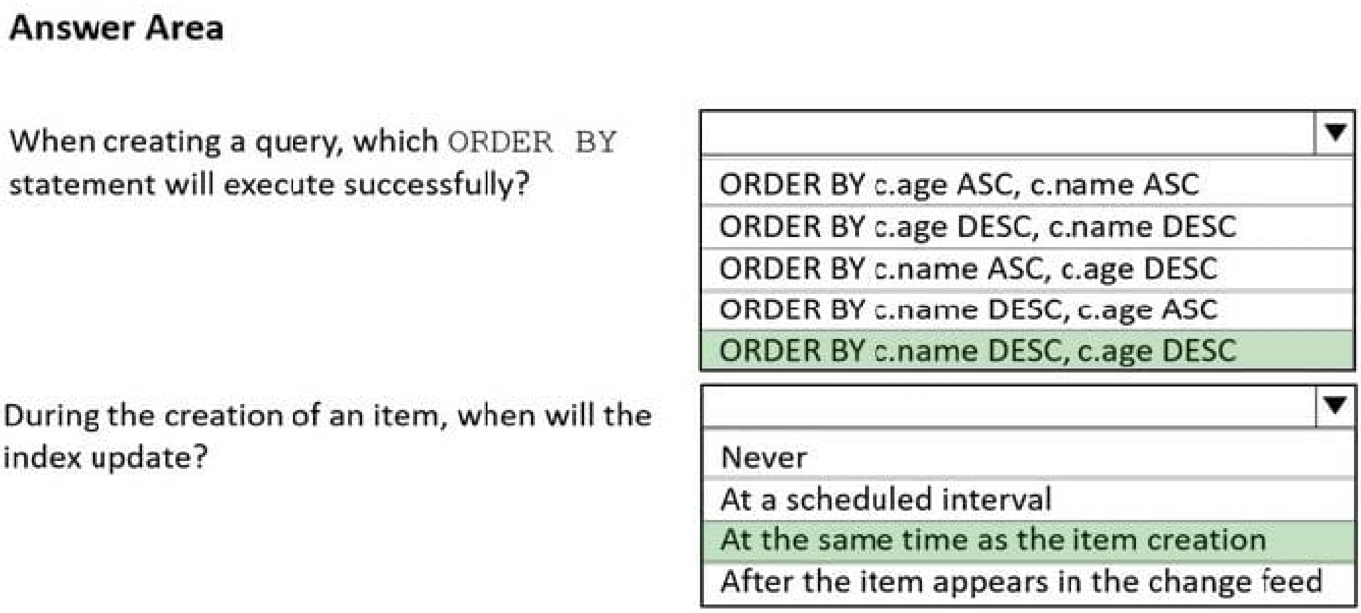
When creating a query, which ORDER BY statement will execute successfully?

- 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:

1.

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.

2.

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

3.

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:

1.

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.

2.

None: Indexing is disabled on the container.

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

---

QUESTION 5

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