

EXAM DP-420

Building AI Applications with Azure Cosmos DB

Practice Question Bank

Every question, answer, and explanation from the exam simulator

112

QUESTIONS

7

QUESTION FORMATS

2

CASE STUDIES

700

PASS SCORE / 1000

Fact-checked. Every question and answer has been checked against current Microsoft Learn documentation, and every explanation links to the pages that support it.

Take the timed, scored, interactive version online: www.mightybs.com/certifications/dp-420/exam/

Study hub: www.mightybs.com/certifications/dp-420/

Updated September 29, 2026. Independent study material, not affiliated with or endorsed by Microsoft. All company names are fictitious.

Part 1: Questions

Answers are not shown here, so you can test yourself first.

- Interactive formats are printed as answer areas. For drop-down items choose one option per row; for drag and drop, match each answer choice to a row (a choice can be used more than once); for build lists, write the actions in order.
- Scenario series share one scenario: decide for each proposed solution whether it meets the goal.
- Case studies come last, each followed by the questions that refer to it.
- Check your answers in [Part 2](#). Each question number links to its answer. On the real exam each correct selection is worth one point, and 700 of 1000 passes.

1 Drop-down selection · Domain 1

You have an Azure Cosmos DB for NoSQL account named account1 that stores data for a distributed application used by Tailspin Toys.

Different containers in account1 have different read-consistency requirements.

You need to choose the weakest consistency level that satisfies each requirement.

Which consistency level should you choose for each requirement?

To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Requirement	Consistency level
A user's reads must include the writes made in the same client session.	Strong · Eventual · Session · Bounded staleness · Consistent prefix
A leaderboard can lag by at most five seconds or 100 updates across regions.	Bounded staleness · Session · Strong · Eventual · Consistent prefix
An activity feed must never show writes out of order, but it does not require the latest value.	Eventual · Strong · Consistent prefix · Session · Bounded staleness

Choose one option in each row.

2 Multiple choice · Domain 1

You have an Azure Cosmos DB for NoSQL container named container1.

Container1 stores 180 GB of data and is provisioned for 65,000 RU/s.

You need to estimate the minimum number of physical partitions implied by the current throughput and storage.

What is the minimum number of physical partitions?

NOTE: Each correct selection is worth one point.

- ☐ A 13
- ☐ B 5
- ☐ C 7
- ☐ D 4
- ☐ E 6

3 Multiple choice · Domain 1

You have an Azure Cosmos DB for NoSQL account named account1 that uses strong consistency.

An application named App1 performs 600 point reads per second. Each read retrieves one item by using the item ID and partition key, and each item is approximately 1 KB.

You need to estimate the RU/s required for the reads only.

Which RU/s estimate should you use?

NOTE: Each correct selection is worth one point.

- ☐ A 1,200 RU/s
- ☐ B 600 RU/s
- ☐ C 2,400 RU/s
- ☐ D 300 RU/s

4 Multiple choice · Domain 1

You have an Azure Cosmos DB for NoSQL account named account1. The default consistency level for account1 is Strong.

An application named App1 performs dashboard reads against container1. The dashboard can use eventual consistency, but other applications must continue to use the account default.

You need to reduce the consistency level for only the dashboard reads.

Which configuration should you use?

NOTE: Each correct selection is worth one point.

- ☐ A Change the default consistency level of account1 to Eventual.
 - ☐ B Enable autoscale throughput on container1 and keep the current consistency level.
 - ☐ C Change the partition key path of container1 to the dashboard identifier.
 - ☐ D Set ItemRequestOptions.ConsistencyLevel to Eventual on the dashboard read requests.
-

5 Multiple choice · Domain 1

You have a .NET deployment task for an Azure Cosmos DB for NoSQL account named account1.

The task might run more than once. It must create a database named db1 and a container named container1 with partition key path /tenantId if they do not already exist.

You need to avoid errors when the database or container already exists.

Which methods should you use?

NOTE: Each correct selection is worth one point.

- ☐ A GetDatabase and GetContainer
- ☐ B CreateDatabaseAsync and CreateContainerAsync
- ☐ C CreateDatabaseIfNotExistsAsync and CreateContainerIfNotExistsAsync
- ☐ D ReadDatabaseAsync and ReadContainerAsync

You have an Azure Cosmos DB for NoSQL container named container1 that stores telemetry documents for Relecloud.

The workload is write-heavy. Applications read most items by item ID and partition key. The payload and debug properties are never used in filters, sort operations, or projections.

You need to reduce RU consumption for write operations without changing the point-read access pattern.

Which two changes should you make?

Each correct answer presents part of the solution. Choose two.

NOTE: Each correct selection is worth one point.

- ☐ A Add another Azure region to the account for the same workload.
- ☐ B Exclude the unused payload and debug paths from indexing.
- ☐ C Remove redundant large properties from the telemetry documents.
- ☐ D Increase the account default consistency level to Strong.
- ☐ E Replace point reads with SQL queries that filter by id.
- ☐ F Add composite indexes that include every property in the documents.

7 Drag and drop · Domain 1

You are designing Azure Cosmos DB for NoSQL resources for Woodgrove Bank.

Each workload has a different traffic pattern or isolation requirement.

You need to choose the most appropriate throughput model for each workload.

Which throughput model should you use for each workload?

To answer, drag the appropriate values to the correct targets. Each value may be used 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.

ANSWER CHOICES

- ① Container-level dedicated throughput
- ② Serverless
- ③ Database-level shared throughput
- ④ Standard (manual) provisioned throughput
- ⑤ Autoscale provisioned throughput

Workload	Throughput model
A prototype has intermittent requests, long idle periods, and no requirement for multiple Azure regions.	
A production API has steady traffic near the configured RU/s every hour, and the team wants to manage a fixed RU/s value.	
A retail events API has unpredictable spikes and requires provisioned-throughput SLAs without custom scaling code.	
Three low-traffic containers can share capacity, and no individual container needs a predictable throughput guarantee.	

8 Yes or No · Domain 1

You have an Azure Cosmos DB for NoSQL container named container1.

Container1 is configured with autoscale maximum throughput of 30,000 RU/s.

You need to evaluate the expected autoscale behavior.

Which statements are correct?

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

NOTE: Each correct selection is worth one point.

Statement	Yes	No
Container1 can scale between 3,000 RU/s and 30,000 RU/s.	☐	☐
For each hour, billing is based on the highest RU/s value that the system scaled to during that hour.	☐	☐
If requests require more than 30,000 RU/s in a second, Azure Cosmos DB automatically scales beyond 30,000 RU/s to avoid rate limiting.	☐	☐

9 Drop-down selection · Domain 1

Tailwind Traders runs a .NET app named App1 in East US against a geo-replicated Azure Cosmos DB for NoSQL account.

App1 must authenticate by using Microsoft Entra ID, prefer East US before West US for regional reads, and use the best-performing connection mode. The CosmosClient instance will be kept for the lifetime of the application.

You need to complete the SDK initialization code.

How should you complete the code?

To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

```
TokenCredential credential = new [A] ();
CosmosClientOptions options = new()
{
    [B] = new List<string> { "East US", "West US" },
    ConnectionMode = [C]
};
CosmosClient client = new(
    accountEndpoint: endpoint,
    tokenCredential: credential,
    clientOptions: options);
```

[A] DefaultAzureCredential · TokenCredential · AzureKeyCredential · CosmosClientOptions

[B] PreferredLocations · ApplicationRegion · ApplicationPreferredRegions · Regions

[C] PortReuseMode.PrivatePortPool · ConnectionMode.Direct · ConsistencyLevel.Session · ConnectionMode.Gateway

10 Multiple response · Domain 1

Proseware is writing a Python application named App1 by using the azure-cosmos package.

App1 runs locally during development and runs in Azure by using a managed identity. The Azure Cosmos DB for NoSQL endpoint is stored in the COSMOS_ENDPOINT environment variable.

You need to initialize the Python SDK client by using Microsoft Entra authentication.

Which two statements should you include?

Each correct answer presents part of the solution. Choose two.

NOTE: Each correct selection is worth one point.

- ☐ A credential = PartitionKey(path='/tenantId')
- ☐ B client = CosmosClient(ENDPOINT)
- ☐ C from azure.identity import DefaultAzureCredential
- ☐ D from azure.cosmos import PartitionKey
- ☐ E client = CosmosClient(ENDPOINT, DefaultAzureCredential())
- ☐ F client = CosmosClient(ENDPOINT, {'masterKey': key})

11 Drop-down selection · Domain 1

You have an Azure Cosmos DB for NoSQL account named account1 that contains a database named db1 and a container named orders.

The orders container stores order documents by using /customerId as the partition key. A C# application named App1 uses the Microsoft.Azure.Cosmos .NET SDK v3.

App1 must update the existing status property of one order and fail if the status path is missing. App1 must also increment a reserved inventory counter and remove a review flag without reading and replacing the whole item.

You need to complete the code that updates the order.

How should you complete the code?

To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

```
List<PatchOperation> operations = new()
{
    PatchOperation. [A] ("/status", "Ready"),
    PatchOperation. [B] ("/inventory/reserved", 1),
    PatchOperation. [C] ("/flags/reviewRequired")
};

ItemResponse<Order> response = await container. [D] <Order> (
    id: orderId,
    partitionKey: new PartitionKey(customerId),
    patchOperations: operations);
```

[A] Add · Move · Replace · Set

[B] Remove · Increment · Set · Replace

[C] Increment · Add · Remove · Set

[D] ReadItemAsync · UpsertItemAsync · ReplaceItemAsync · PatchItemAsync

12 Drop-down selection · Domain 1

Fabrikam has an Azure Cosmos DB for NoSQL container named products that is partitioned by /categoryId. A C# application named App1 reads a product, allows a user to edit the price, and then saves the product.

The save operation must fail if another client updated the product after App1 read it. App1 must reload the product when the save fails because of a stale ETag.

You need to complete the code that implements optimistic concurrency control.

How should you complete the code?

To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

```
ItemResponse<Product> read = await container.ReadItemAsync<Product>(
    id, new PartitionKey(categoryId));
Product updated = read.Resource;
updated.price = newPrice;
ItemRequestOptions options = new() { [A] = read.ETag };

try
{
    await container. [B] <Product>(updated, id, new PartitionKey(categoryId), options);
}
catch ( [C] ex) when (ex.StatusCode == HttpStatusCode. [D] )
{
    await ReloadAndRetryAsync(id, categoryId);
}
```

[A] IfNoneMatchEtag · IfMatchEtag · ConsistencyLevel · SessionToken

[B] CreateItemAsync · ReadItemAsync · ReplaceItemAsync · GetItemQueryIterator

[C] HttpRequestException · TaskCanceledException · CosmosException · InvalidOperationException

[D] NotModified · TooManyRequests · PreconditionFailed · Conflict

13 Drop-down selection · Domain 1

Northwind Traders has an Azure Cosmos DB for NoSQL container named ledger that is partitioned by /accountId.

A C# application named App1 must create a payment document and a matching ledger entry for the same accountId.

Both writes must commit or roll back as one transaction within the same logical partition.

You need to complete the code that executes the transactional batch.

How should you complete the code?

To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

```
PartitionKey partitionKey = new(accountId);

TransactionalBatch batch = container. [A] (partitionKey)
    . [B] <Payment>(payment)
    .CreateItem<LedgerEntry>(entry);

using TransactionalBatchResponse response = await batch. [C] ();
if (response. [D] )
{
    TransactionalBatchOperationResult<Payment> result =
        response.GetOperationResultAtIndex<Payment>(0);
}
```

[A] CreateItemAsync · CreateTransactionalBatch · ReadManyItemsAsync · GetItemQueryIterator

[B] ExecuteAsync · PatchItemAsync · CreateItemAsync · CreateItem

[C] SaveChangesAsync · ExecuteAsync · RunAsync · ReadNextAsync

[D] IsSuccessStatusCode · HasMoreResults · ContinuationToken · Resource

14 Drop-down selection · Domain 1

Tailwind Traders has an Azure Cosmos DB for NoSQL container named tasks that is partitioned by /tenantId. A Python application named App1 uses the azure-cosmos SDK. The application receives tenant_id and status values from a request body.

App1 must query only one tenant partition and must avoid concatenating user input into the query string.

You need to complete the Python code.

How should you complete the code?

To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

```
query = (
    "SELECT * FROM c WHERE c.tenantId = @tenantId "
    "AND c.status = @status"
)
params = [
    {"name": "@tenantId", "value": tenant_id},
    {"name": "@status", "value": status}
]

items = container. [A] (
    query=query,
    [B] =params,
    [C] =tenant_id)

for item in items:
    process_item(item)
```

[A] read_item · read_all_items · execute_item_batch · query_items

[B] parameters · initial_headers · body · partition_key

[C] partition_key · continuation_token_limit · max_item_count · enable_cross_partition_query

15 Yes or No - Domain 1

Proseware has an Azure Cosmos DB for NoSQL container named orders. A web API named App1 returns orders to clients one page at a time.

You review the following C# code that returns one page of open orders.

```
01 QueryRequestOptions options = new() { MaxItemCount = 50 };
02 using FeedIterator<Order> feed = container.GetItemQueryIterator<Order>(
03     queryText: "SELECT * FROM c WHERE c.status = 'Open'",
04     continuationToken: continuationToken,
05     requestOptions: options);
06 if (feed.HasMoreResults)
07 {
08     FeedResponse<Order> page = await feed.ReadNextAsync();
09     continuationToken = page.ContinuationToken;
10     double charge = page.RequestCharge;
11     foreach (Order order in page)
12     {
13         SendToClient(order);
14     }
15 }
```

For each statement about the code, select Yes or No.

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

NOTE: Each correct selection is worth one point.

Statement	Yes	No
Line 01 sets 50 as the maximum number of items that the query engine should return in a single query execution.	☐	☐
Line 09 captures the token that can be supplied on the next request to resume the query when more results remain.	☐	☐
Because MaxItemCount is 50, every page before the final page is guaranteed to contain exactly 50 orders.	☐	☐

16 Drag and drop · Domain 1

Adventure Works is building App1 by using the Azure Cosmos DB for NoSQL .NET SDK v3.

The application works with a container named container1 that is partitioned by /tenantId.

You need to select the most appropriate database operation approach for each requirement.

Which approach should you use for each requirement?

To answer, drag the appropriate values to the correct targets. Each value may be used 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.

ANSWER CHOICES

- 1 TransactionalBatch
- 2 Query with.GetItemQueryIterator
- 3 Bulk execution with AllowBulkExecution = true
- 4 Change feed processor
- 5 Point read with ReadItemAsync

Requirement	Approach
Retrieve one known session document when App1 has both the id and tenantId values.	
Return all delayed orders for a tenant when the number of matching orders is not known.	
Create an order header and a reservation item atomically for the same tenantId value.	
Ingest 200,000 independent telemetry items across many tenantId values as quickly as possible.	

17 Multiple choice · Domain 1

Woodgrove Bank has an Azure Cosmos DB for NoSQL container named profiles that is partitioned by /customerId.

App1 receives the id and customerId for a 1-KB profile item from an authenticated request.

App1 must retrieve exactly one profile with the lowest RU cost.

You need to select the read operation for App1.

Which operation should App1 use?

NOTE: Each correct selection is worth one point.

- A Use GetItemQueryIterator with a query that filters by id and customerId.
- B Use ReadAllItemsAsync and stop after the first item with the matching id.
- C Use ReadItemAsync with the id and a PartitionKey created from customerId.
- D Use GetItemQueryIterator without a partition key and filter the result in App1.

18 Multiple choice · Domain 1

Coho Winery has an Azure Cosmos DB for NoSQL container named carts.

The container DefaultTimeToLive property is null. Several cart items include a ttl property set to 3600.

The development team needs to understand whether the items expire automatically.

You need to identify the TTL behavior.

What happens to the cart items?

NOTE: Each correct selection is worth one point.

- ☐ A The items never expire because container-level TTL is off, so the item ttl value has no effect.
 - ☐ B The items expire only after DefaultTimeToLive is changed to 0.
 - ☐ C The items are deleted immediately because any ttl property enables expiration.
 - ☐ D The items expire 3,600 seconds after their last modified time.
-

19 Multiple choice · Domain 1

Litware has an Azure Cosmos DB for NoSQL container named catalog that stores product documents.

A C# application named App1 receives a complete product document for an existing product.

App1 must overwrite the existing product document. The operation must fail if the product does not already exist, and App1 must not create a new item by accident.

You need to choose the SDK method for App1.

Which method should App1 call?

NOTE: Each correct selection is worth one point.

- ☐ A CreateItemAsync
- ☐ B PatchItemAsync
- ☐ C ReplaceItemAsync
- ☐ D UpsertItemAsync

20 Multiple response · Domain 1

Wide World Importers uses an Azure Cosmos DB for NoSQL container named shipments that is partitioned by /warehouseId.

A developer is designing a TransactionalBatch that creates shipment, invoice, and audit documents.

The batch must comply with documented Azure Cosmos DB transactional batch constraints.

You need to identify the constraints for the batch.

Which three constraints apply to the TransactionalBatch?

Each correct answer presents part of the solution. Choose three.

NOTE: Each correct selection is worth one point.

- ☐ A The batch can include SQL query operations that return multiple items.
- ☐ B The batch can span logical partitions when all items are in the same container.
- ☐ C All operations in the batch must use the same logical partition key value.
- ☐ D The batch can contain at most 100 operations.
- ☐ E When one operation fails, earlier successful operations remain committed.
- ☐ F The request payload must not exceed 2 MB, and execution time is limited to 5 seconds.

21 Multiple choice · Domain 1

Alpine Ski House has an Azure Cosmos DB for NoSQL container named events.

A C# application named Loader1 creates a CosmosClient with AllowBulkExecution set to true. Loader1 creates a list of CreateItemAsync tasks for 300,000 independent event items and then calls Task.WhenAll.

The development team needs to understand how SDK bulk execution changes request handling.

You need to describe the expected SDK behavior.

What should you tell the development team?

NOTE: Each correct selection is worth one point.

- ☐ A The SDK makes the 300,000 creates one ACID transaction that either commits or rolls back as a unit.
- ☐ B The SDK bypasses request unit charges for the create operations after AllowBulkExecution is enabled.
- ☐ C The SDK requires replacing CreateItemAsync with a BulkCreateItemAsync method before any batching occurs.
- ☐ D The SDK serializes the creates one item at a time to keep total RU consumption lower than individual operations.
- ☐ E The SDK converts the concurrent point operations into internal batches, dispatches them across partitions, and maps individual results back to the original tasks.

22 Drop-down selection · Domain 1

You have an Azure Cosmos DB for NoSQL account named account1.

App1 is a B2C SaaS application that stores data for tenants in shared containers.

Use the simplest partitioning design that satisfies each workload.

Avoid full fan-out queries for the primary tenant lookup.

You need to choose a partition key design for each workload.

Which partition key should you use for each workload?

To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Workload	Partition key
ChatHistory: thousands of small tenants, each tenant stores less than 2 GB, and all transactional queries filter by tenantId.	/tenantId · /tenantId then /userId then /sessionId (hierarchical) · /planType · /id
ActivityEvents: some tenants can exceed 20 GB, queries filter by tenantId and often userId, and each tenant must scale beyond one logical partition.	/tenantId · /id · /tenantId then /userId then /id (hierarchical) · /tenantId_userId (synthetic)

Choose one option in each row.

23 Drop-down selection · Domain 1

You have an Azure Cosmos DB for NoSQL database named db1.

The database contains several containers that support different application workloads.

You need to select a partition key for each container or workload.

Which partition key should you use for each workload?

To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Container or workload	Partition key
Orders: writes are heavy at month end for each tenant. Queries usually filter by tenantId and orderMonth. No single existing property has both high cardinality and query alignment.	/status · /tenantId_orderMonth (synthetic concatenation) · /orderMonth · /tenantId
ProductReads: every operation is a point read or write by item id. The app rarely filters on other properties and needs maximum write distribution.	/tenantId · /status · /category · /id
DeviceTelemetry: 50,000 devices write once per second, device data can exceed 20 GB, and real-time queries filter by deviceId and a time slice.	/deviceId then /timestamp then /id (hierarchical) · /districtId · /deviceId · /timestamp

Choose one option in each row.

You are designing item models for an Azure Cosmos DB for NoSQL database used by Northwind Traders. You need to choose how to model each relationship based on growth, update frequency, and query patterns.

Which data model should you use for each relationship?

To answer, drag the appropriate values to the correct targets. Each value 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.

ANSWER CHOICES

- 1 Embed the related data in the parent item
- 2 Create a denormalized read projection
- 3 Reference the related data as separate items

Relationship	Data model
Order to shipping address. The address is captured at checkout, is read with the order, and does not change after checkout.	
Post to comments. A popular post can receive an unbounded number of comments, and comments are paged independently.	
Portfolio to stock quote. The quote is shared by many portfolios and can update many times per day.	
Book list to author summary. Each book list always shows the author's name and thumbnail, while the full author item is maintained separately and changes rarely.	

25 Yes or No · Domain 1

You review a proposed Azure Cosmos DB for NoSQL design for Fabrikam.

The Orders container uses /status as the partition key, and most new items have a status value of Pending.

The team is also considering an HPK of /tenantId, /userId, and /sessionId for a read-heavy activity container.

You need to identify partitioning issues in the design.

For each statement, should you select Yes or No?

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

NOTE: Each correct selection is worth one point.

Statement	Yes	No
Using /status as the partition key can create a hot logical partition and can also run into the 20-GB logical partition limit if most items remain Pending.	☐	☐
In the HPK container, a query that filters by tenantId is routed to the subset of physical partitions that contain that tenant's data.	☐	☐
In the same HPK container, filtering only by userId avoids cross-partition fan-out because userId is a high-cardinality second-level key.	☐	☐

26 Multiple choice · Domain 1

You have an Azure Cosmos DB for NoSQL container named container1.

Container1 is expected to require 85,000 RU/s of provisioned throughput and 320 GB of data storage.

You need to estimate the minimum number of physical partitions required for the workload.

What is the minimum number of physical partitions required?

NOTE: Each correct selection is worth one point.

- ☐ A 8
- ☐ B 7
- ☐ C 17
- ☐ D 9

27 Multiple choice · Domain 1

You are designing a product catalog for Adventure Works in Azure Cosmos DB for NoSQL.

A product details page must show product attributes and the three most recent reviews with one read.

Customers can add an unbounded number of reviews, and the full review list is shown by using pagination.

You need to design the item model for products and reviews.

Which model should you use?

NOTE: Each correct selection is worth one point.

- ☐ A Store only review ids in the product item and query each review by id during page rendering.
- ☐ B Embed recentReviews in the product item and store each review as a separate item with productId.
- ☐ C Embed all reviews in the product item.
- ☐ D Store products and reviews in separate containers and join them in every product query.

28 Multiple response · Domain 1

Relecloud stores high-volume event data in Azure Cosmos DB for NoSQL.

No single existing property provides both high cardinality and alignment with the main query patterns.

You need to implement synthetic partition keys to improve data and workload distribution.

Which three methods can you use?

Each correct answer presents a complete solution. Choose three.

NOTE: Each correct selection is worth one point.

- ☐ A Append a random numeric suffix to a time-based key during writes.
- ☐ B Use the same constant partition key value and increase provisioned RU/s.
- ☐ C Patch the partition key value after the item is stored to rebalance data.
- ☐ D Use /status because it contains a small set of business states.
- ☐ E Append a suffix calculated from a queryable value, such as a hash of VIN.
- ☐ F Concatenate tenantId and eventDate into a client-created partitionKey property.

29 Multiple response · Domain 1

Woodgrove Bank is designing a shared Azure Cosmos DB for NoSQL container for tenant activity data.

The proposed HPK is /tenantId, /userId, and /sessionId.

You need to evaluate the HPK design.

Which two statements describe how HPKs affect the design?

Each correct answer presents a complete solution. Choose two.

NOTE: Each correct selection is worth one point.

- ☐ A Queries that specify tenantId, or tenantId and userId, are routed to relevant physical partitions.
- ☐ B You can configure as many partition key levels as the workload requires.
- ☐ C A low-cardinality first level is preferred for write-heavy ingestion because it collocates data.
- ☐ D A query that filters only on userId is efficiently routed because userId is the second level.
- ☐ E You can add the HPK to the existing container without creating a new container.
- ☐ F A tenantId prefix can exceed 20 GB because lower levels create full logical partition paths.

30 Multiple choice · Domain 1

Tailspin Toys has an Azure Cosmos DB for NoSQL container named Users.

The container uses /email as the partition key, and users can now change their email address.

The application team wants to use /userId as the partition key instead.

You need to recommend the supported way to change the partition key design.

What should you recommend?

NOTE: Each correct selection is worth one point.

- ☐ A Replace each user item and set the email partition key value to the new email address.
- ☐ B Add /userId as a unique key so Azure Cosmos DB treats it as the partition key.
- ☐ C Scale the container to add physical partitions, and then update the partition key path in place.
- ☐ D Create a destination container with /userId, copy the data, and update App1 to use the new container.

31 Multiple response · Domain 1

Northwind Traders stores users, posts, comments, and likes in Azure Cosmos DB for NoSQL.

The home feed must show author names, comment counts, and like counts without issuing extra reads for each post.

User names can change occasionally, and the feed can tolerate eventual consistency for duplicated display fields.

You need to design a read-optimized model and keep duplicated data synchronized.

Which two actions should you include in the design?

Each correct answer presents part of the solution. Choose two.

NOTE: Each correct selection is worth one point.

- ☐ A Use the change feed to process source changes and update denormalized copies or materialized views.
- ☐ B Duplicate author name and aggregate count fields in the post items used by the feed.
- ☐ C Depend on query-time joins across user and post items for each feed request.
- ☐ D Store every user, post, comment, and like for a user in one large user item.
- ☐ E Serve the feed by reading the change feed directly instead of querying stored items.

32 Multiple choice · Domain 1

Proseware is planning an Azure Cosmos DB for NoSQL workload for an AI application.

The workload includes documents of several sizes, point reads, writes, filter queries, vector queries, and change feed consumers.

The solution is not yet deployed.

You need to estimate monthly cost and RU/s before provisioning the account.

Which approach should you use?

NOTE: Each correct selection is worth one point.

- ☐ A Use only the storage price because identical queries always consume deterministic request units.
- ☐ B Use the Azure Cosmos DB Cost Estimator and configure data, traffic, operations, consistency, regions, vector operations, and change feed consumers.
- ☐ C Calculate physical partitions only and multiply the result by 10,000 RU/s.
- ☐ D Use Azure Monitor normalized RU metrics and current alerts from the production account.

33 Multiple choice · Domain 1

You review an Azure Cosmos DB for NoSQL Orders container for Wide World Importers.

The container uses /id as the partition key.

App1 rarely performs point reads by id and partition key. Most requests run queries that filter by customerId and orderDate.

You need to identify the primary issue with the partition key design.

Which issue is the design most likely to cause?

NOTE: Each correct selection is worth one point.

- ☐ A Writes will be limited to one logical partition because all items share the same id.
- ☐ B Point reads by id and partition key will cost more RUs than equivalent cross-partition queries.
- ☐ C Queries that filter by customerId and orderDate will fan out because they do not filter by the partition key.
- ☐ D Each customer logical partition will exceed 20 GB because /id groups all customer orders together.

34 Drop-down selection · Domain 1

Contoso has an Azure Cosmos DB for NoSQL account named account1 that contains a database named db1.

The database contains a source container named container1 and a lease container named leases1.

You are writing a .NET worker named App1 by using the Microsoft.Azure.Cosmos SDK.

You need to start a change feed processor for container1 and store lease state in leases1.

How should you complete the code?

To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

```
Container sourceContainer = client.GetContainer("db1", "container1");
Container leaseContainer = client.GetContainer("db1", "leases1");
ChangeFeedProcessor processor = sourceContainer
    .GetChangeFeedProcessorBuilder<Order>("ordersProcessor", HandleChangesAsync)
    . [A] ("worker-01")
    . [B] (leaseContainer)
    . [C] ();
await processor. [D] ();
```

[A] WithLeaseContainer · WithInstanceName · WithStartTime · GetChangeFeedProcessorBuilder

[B] WithLeaseContainer · GetChangeFeedIterator · WithPollInterval · WithInstanceName

[C] ReadNextAsync · Build · StartAsync · StopAsync

[D] StartAsync · WithLeaseContainer · ReadNextAsync · Build

35 Drag and drop · Domain 1

Tailwind Traders has an Azure Cosmos DB for NoSQL account named account1 that contains a container named container1.

Three development teams need to process changes from container1.

You need to recommend a change feed implementation for each team.

Which implementation should you recommend for each requirement?

To answer, drag the appropriate values to the correct targets. Each value 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.

ANSWER CHOICES

- ① Pull model with a FeedIterator from GetChangeFeedIterator
- ② Change feed processor
- ③ Azure Functions trigger for Azure Cosmos DB

Requirement	Model
Run a serverless handler that starts automatically when container1 changes and does not require the team to maintain worker infrastructure.	
Read only the changes for one tenant partition key and control how quickly App1 polls for more changes.	
Run several .NET worker instances that automatically share leases, checkpoint progress, and load balance changes for the full container.	

36 Build list · Domain 1

Fabrikam has an Azure Cosmos DB for NoSQL account named account1 that contains a database named db1 and a container named container1.

App1 writes to container1 by using /customerId as the partition key.

You create a new design that uses /tenantId as the partition key. The account write region supports container copy jobs.

You need to move the data to a new container by using an online container copy job and minimize write downtime during cutover.

Which five 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.

ACTIONS

- ☐ A Enable the EnableOnlineContainerCopy capability on account1 and wait at least 30 minutes.
- ☐ B Delete container1 before creating the copy job so that the target container can reuse its name.
- ☐ C Create an online container copy job from container1 to the target container, and monitor the job until it processes the source changes.
- ☐ D Create a target container in db1 that uses /tenantId as the partition key and the required settings.
- ☐ E Stop writes to container1, complete the copy job, and point App1 to the target container.
- ☐ F Create a lease container with a /id partition key for the copy job.
- ☐ G Enable continuous backup and the all versions and deletes change feed feature on account1.

37 Yes or No · Domain 1

Woodgrove Bank has an Azure Cosmos DB for NoSQL account named account1 that contains a container named container1.

You review change feed behavior before implementing event processors.

Which statements are 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.

Statement	Yes	No
Latest version mode captures delete operations when an item expires because of time to live.	☐	☐
All versions and deletes mode can include delete operations, but account1 must use continuous backup and have the all versions and deletes feature enabled.	☐	☐
The change feed guarantees one global order for changes across all partition key values in container1.	☐	☐

38 Multiple choice · Domain 1

Northwind Traders has an Azure Cosmos DB for NoSQL account named account1 that contains a database named db1.

You create an Azure Functions trigger for Azure Cosmos DB that monitors a container named container1.

You will create the partitioned lease container manually.

You need to configure the partition key for the lease container.

Which partition key path should you use?

NOTE: Each correct selection is worth one point.

- ☐ A /processorName
- ☐ B /id
- ☐ C /leaseToken
- ☐ D /partitionKey

39 Multiple response · Domain 1

Adventure Works has an Azure Cosmos DB for NoSQL account named account1 that contains a container named container1.

You are designing change feed consumers for container1.

You need to identify when to use all versions and deletes mode instead of latest version mode.

Which two requirements require all versions and deletes mode?

Each correct answer presents a complete solution. Choose two.

NOTE: Each correct selection is worth one point.

- ☐ A Start a new processor from a specific past time instead of from now.
- ☐ B Record item delete operations for an audit log.
- ☐ C Run the processor without enabling continuous backup on the account.
- ☐ D Synchronize current item images from the beginning of the container with no fixed retention window.
- ☐ E Detect each intermediate update when the same item changes multiple times between reads.

40 Multiple choice · Domain 1

Relecloud has a .NET app named App1 that uses the change feed pull model for an Azure Cosmos DB for NoSQL container named container1.

App1 creates a FeedIterator by calling GetChangeFeedIterator.

A call to ReadNextAsync returns HttpStatusCode.NotModified.

You need to continue processing changes later without losing the current position and without tight polling.

What should App1 do next?

NOTE: Each correct selection is worth one point.

- ☐ A Capture the continuation token, wait briefly, and poll the iterator again later.
- ☐ B Treat the change feed as complete and stop polling permanently.
- ☐ C Recreate container1 to reset the change feed starting point.
- ☐ D Create a lease document so the pull model can checkpoint automatically.

41 Yes or No · Domain 1

Alpine Ski House wants developers to use Azure Cosmos DB Agent Kit with AI coding assistants.

Which statements are 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.

Statement	Yes	No
The Agent Kit is a collection of skills that helps AI coding assistants follow Azure Cosmos DB best practices.	☐	☐
Before using the recommended installation command, the workstation needs an AI coding assistant that supports Agent Skills and Node.js with npm and npx.	☐	☐
The best-performing hosts load the entire Agent Kit into every prompt so every rule is always present.	☐	☐

42 Multiple response · Domain 1

Fourth Coffee is building an AI application on Azure Cosmos DB for NoSQL.

The development team installed Azure Cosmos DB Agent Kit for its AI coding assistant.

You need to identify tasks where the Agent Kit is a best fit.

Which two tasks should the team ask the assistant to help with?

Each correct answer presents a complete solution. Choose two.

NOTE: Each correct selection is worth one point.

- ☐ A Tune Microsoft Teams retention policies.
 - ☐ B Create Exchange Online mail flow rules.
 - ☐ C Apply design patterns for agentic workflows.
 - ☐ D Configure Azure SQL Database elastic pools.
 - ☐ E Plan indexing strategies for vector workloads.
-

43 Multiple choice · Domain 1

Litware developers use an Agent Skills-compatible AI coding assistant.

Their workstations have Node.js, npm, and npx installed.

You need to install Azure Cosmos DB Agent Kit by using the recommended option for most users.

Which command should the developers run?

NOTE: Each correct selection is worth one point.

- ☐ A npx skills add AzureCosmosDB/cosmosdb-agent-kit
 - ☐ B apm install AzureCosmosDB/cosmosdb-agent-kit
 - ☐ C gemini extensions install <https://github.com/AzureCosmosDB/cosmosdb-agent-kit>
 - ☐ D /plugin marketplace add AzureCosmosDB/cosmosdb-agent-kit
-

44 Multiple choice · Domain 1

Proseware is selecting an AI coding assistant host for Azure Cosmos DB Agent Kit.

The team wants the assistant to keep context focused while building an AI application.

You need to choose the host capability that works best with the Agent Kit.

Which capability should the host provide?

NOTE: Each correct selection is worth one point.

- ☐ A A database migration wizard with no AI coding assistant.
- ☐ B A static JSON linter that cannot load skills.
- ☐ C Progressive, on-demand loading of relevant Agent Skills.
- ☐ D Automatic injection of the full kit into every prompt.

45 Drop-down selection · Domain 2

You have an Azure Cosmos DB for NoSQL account named account1 that contains a database named db1 and a container named container1.

An app named App1 runs in Azure App Service by using a managed identity. Key-based authentication is disabled on account1.

You need to complete the C# code so that App1 authenticates by using Microsoft Entra ID.

How should you complete the code?

To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

```
using Azure.Core;
using Azure.Identity;
using Microsoft.Azure.Cosmos;

string endpoint = "https://account1.documents.azure.com:443/";
[A] credential = new [B] ();
CosmosClient client = new(endpoint, credential);
Container container = client.GetContainer("db1", "container1");
await container.ReadItemAsync<dynamic>("item1", new PartitionKey("pk1"));
```

[A] CosmosClientOptions · string · TokenCredential

[B] CosmosClient · DefaultAzureCredential · AzureKeyCredential

46 Drag and drop · Domain 2

You have an Azure Cosmos DB for NoSQL account named account1 that supports several apps and administrators.

You need to choose the least privileged role or mechanism for each requirement.

Which role or mechanism should you use for each requirement?

To answer, drag the appropriate values to the correct targets. Each value may be used 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.

ANSWER CHOICES

- ① Cosmos DB Operator
- ② DocumentDB Account Contributor
- ③ Cosmos DB Built-in Data Reader
- ④ Cosmos DB Built-in Data Contributor
- ⑤ Custom native data plane role

Requirement	Role or mechanism
Allow App1 to perform point reads, NoSQL queries, and change feed reads without writes.	
Allow App2 only to create, replace, upsert, and delete items in a container.	
Allow an operator to manage account settings without reading data or account keys.	
Allow a support engineer to manage account resources when key operations are acceptable.	

47 Build list · Domain 2

You are deploying an Azure Cosmos DB for NoSQL account named account1 for Fabrikam.

App1 runs in subnet1 of VNet1. The account must not be reachable over the public network at any point in the deployment.

You need to configure private access for account1.

Which four 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.

ACTIONS

- A Create a private endpoint for the Sql target subresource in subnet1.
- B Verify that App1 resolves the account endpoint to private endpoint IP addresses.
- C Add 0.0.0.0 to the IP firewall allowed list.
- D Set publicNetworkAccess to Disabled when account1 is created.
- E Integrate the private endpoint with the privatelink.documents.azure.com private DNS zone.
- F Enable Accept connections from within Azure datacenters.

48 Yes or No · Domain 2

You have an Azure Cosmos DB for NoSQL account named account1 that has Dynamic Data Masking enabled.

For each statement, select Yes or No.

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

NOTE: Each correct selection is worth one point.

Statement	Yes	No
For users without unmask permission, DDM masks sensitive values in read and query responses while the stored data remains unchanged.	☐	☐
A client that uses account keys can access containers protected by DDM.	☐	☐
A user without unmask permission can read both latest mode and all versions and deletes mode change feeds from a masked container.	☐	☐

49 Multiple response · Domain 2

You have an Azure Cosmos DB for NoSQL account named account1 that contains a database named db1 and a container named container1.

App1 runs by using a managed identity and has been updated to use DefaultAzureCredential.

App1 must read and write items in container1.

No application can authenticate to account1 by using account keys.

You need to finish the migration from keys to Microsoft Entra ID authentication.

Which two actions should you perform?

Each correct answer presents part of the solution. Choose two.

NOTE: Each correct selection is worth one point.

- ☐ A Add the App1 subnet to the account firewall allowed list.
- ☐ B Store the primary key in Azure Key Vault and read it during startup.
- ☐ C Assign App1 the Cosmos DB Built-in Data Contributor role at /dbs/db1/colls/container1.
- ☐ D Assign App1 the DocumentDB Account Contributor role at the resource group scope.
- ☐ E Set properties.disableLocalAuth to true on account1.

50 Multiple choice · Domain 2

You have an Azure Cosmos DB for NoSQL account named account1 for Wide World Importers. App1 runs in VNet1. App2 runs on-premises and connects to VNet1 by using ExpressRoute private peering. Both apps must access account1 by using private IP addresses. Requests must avoid the standard public route. You need to configure network access to account1.

Which networking feature should you use?

NOTE: Each correct selection is worth one point.

- ☐ A The Accept connections from within Azure datacenters firewall option
- ☐ B A virtual network service endpoint for the App1 subnet only
- ☐ C An Azure Private Link private endpoint with private DNS
- ☐ D An IP firewall rule for the public IP address of each app

51 Multiple response · Domain 2

You have an Azure Cosmos DB for NoSQL account named account1 that contains a database named db1 and a container named container1.

Function1 uses a system-assigned managed identity.

Function1 must perform point reads and NoSQL queries only against container1.

Function1 must not receive access to any other container.

You need to configure native data plane RBAC for Function1.

Which two settings should you use?

Each correct answer presents part of the solution. Choose two.

NOTE: Each correct selection is worth one point.

- ☐ A Role definition: Cosmos DB Operator
- ☐ B Scope: /dbs/db1/colls/container1
- ☐ C Role definition: Cosmos DB Built-in Data Reader
- ☐ D Role definition: Azure Reader
- ☐ E Scope: the Azure resource group that contains account1

52 Multiple choice · Domain 2

You have an older web app named App1 that connects to an Azure Cosmos DB for NoSQL account named account1 by using the account endpoint and primary key.

A planned rewrite will move App1 to managed identity, but the rewrite is not ready.

The endpoint and key must not be stored in application code or clear text configuration files.

App1 runs in Azure App Service.

You need to reduce secret exposure until managed identity is available.

Which approach should you use?

NOTE: Each correct selection is worth one point.

- ☐ A Add the App Service outbound IP addresses to the account firewall only.
- ☐ B Assign Cosmos DB Operator to App1 and keep using the primary key.
- ☐ C Store the endpoint and key as Azure Key Vault secrets and reference them from App Service settings.
- ☐ D Store the endpoint and key in an appsettings.json file deployed with the app.

53 Multiple response · Domain 2

You have an Azure Cosmos DB for NoSQL container named container1 that stores employee documents for Proseware.

DDM is enabled on the account.

All fields must be masked by default.

The id field and each projectId field inside the projects array must remain visible.

You need to define the masking policy for container1.

Which two policy entries should you configure?

Each correct answer presents part of the solution. Choose two.

NOTE: Each correct selection is worth one point.

- ☐ A Add an excludedPaths entry for /projects/[0]/projectId.
- ☐ B Add an includedPaths entry with path /projects/[].
- ☐ C Mask the partition key path so Data Explorer can show document details.
- ☐ D Omit the / included path and define excludedPaths only.
- ☐ E Add excludedPaths entries for /id and /projects/[0]/projectId.
- ☐ F Add an includedPaths entry with path /.

SCENARIO SERIES (3 QUESTIONS)

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 on the review screen.

Contoso has an Azure Cosmos DB for NoSQL account named account1 that contains a database named db1. account1 is replicated to East US and West Europe.

App1 is deployed in both Azure regions and uses the Azure Cosmos DB SDK.

Each healthy App1 deployment must write to the nearest account region during normal operations.

During a regional outage, App1 must continue to accept writes through a healthy region without waiting for an account-level failover operation.

Strong consistency is not required, and the application can tolerate server-side conflict resolution.

You need to configure account1 and App1 to meet the requirements.

54 SOLUTION

You enable multi-region writes on account1. In each App1 deployment, you configure the Azure Cosmos DB SDK to prefer the local account region first.

Does the solution meet the goal? Yes ☐ No ☐

55 SOLUTION

You add West Europe as a read region, enable service-managed failover, and leave multi-region writes disabled.

Does the solution meet the goal? Yes ☐ No ☐

56 SOLUTION

You keep account1 in a single Azure region and configure continuous backup with the 7-day tier.

Does the solution meet the goal? Yes ☐ No ☐

57 Yes or No - Domain 2

Wide World Importers has an Azure Cosmos DB for NoSQL account with multi-region writes enabled.

A container named container1 in db1 must use a merge stored procedure named resolver for custom conflict resolution.

You need to evaluate the conflict resolution design.

You review the following JavaScript stored procedure.

```
01 function resolver(incomingItem, existingItem, isTombstone, conflictingItems) {
02   var collection = getContext().getCollection();
03   if (!incomingItem) {
04     if (existingItem) {
05       collection.deleteDocument(existingItem._self, {}, function (err) { if (err) throw err;
06     });
07   }
08   return;
09 }
10 if (existingItem && incomingItem.mergeVersion < existingItem.mergeVersion) return;
11 for (var i = 0; i < conflictingItems.length; i++) {
12   if (incomingItem.mergeVersion < conflictingItems[i].mergeVersion) return;
13 }
14 // The rest of the procedure deletes conflicts and writes incomingItem.
15 }
```

Which statements are 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.

Statement	Yes	No
Line 01 uses the required parameter list for a custom conflict resolution stored procedure.	☐	☐
The container policy must set Mode to ConflictResolutionMode.Custom and ResolutionProcedure to dbs/db1/colls/container1/sprocs/resolver when Azure Cosmos DB should invoke this procedure automatically.	☐	☐
If resolver throws an exception at runtime, Azure Cosmos DB hides the conflict from the conflicts feed.	☐	☐

Fabrikam is reviewing backup and restore requirements for Azure Cosmos DB for NoSQL accounts.

You need to choose the backup policy or restore option that best matches each requirement.

Which choice should you use for each requirement?

To answer, drag the appropriate values to the correct targets. Each value may be used 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.

ANSWER CHOICES

- ① Continuous backup, 30-day tier
- ② Point-in-time restore in the same account
- ③ Point-in-time restore into a new account
- ④ Continuous backup, 7-day tier
- ⑤ Periodic backup mode

Requirement	Choice
A new account must use scheduled backups, and a support-request restore is acceptable.	
Self-service point-in-time restore is required for accidental deletes for up to six days, with no backup storage charge for the tier.	
Self-service point-in-time restore is required for corruption that might be discovered after 21 days.	
An accidentally deleted container must be restored back into account1 instead of into a separate account.	

59 Build list · Domain 2

Northwind Traders has an Azure Cosmos DB for NoSQL account named account1 that uses continuous backup.

A container named container1 in db1 was accidentally deleted.

container1 must be restored to account1.

You need to restore container1 by using the Azure portal.

Which five 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.

ACTIONS

- ☐ A Submit a support request for a periodic backup restore.
- ☐ B Select Restore for the deleted container event after you verify the restore time.
- ☐ C Create a new target account before starting the restore.
- ☐ D Select the Restore to same account tab.
- ☐ E Enter a database search query and a start and end time that include the deletion, and then refresh the event feed.
- ☐ F Open the Point In Time Restore page for account1.
- ☐ G Track the restore until the container status changes from Creating to Online.

60 Multiple response · Domain 2

Woodgrove Bank has an Azure Cosmos DB for NoSQL account named account1.

account1 is a single-write-region account that has East US as the write region and West US as a read region.

account1 uses provisioned throughput and session consistency.

App1 uses the Microsoft.Azure.Cosmos .NET SDK v3.55.0.

You need to enable per-partition automatic failover without causing application availability issues.

Which two actions should you perform?

Each correct answer presents part of the solution. Choose two.

NOTE: Each correct selection is worth one point.

- ☐ A Convert account1 to serverless capacity mode before enabling per-partition automatic failover.
- ☐ B Enable the Per-partition automatic failover feature on account1 after all application instances are upgraded.
- ☐ C Remove the West US read region so that only the write region remains.
- ☐ D Upgrade every App1 instance to Microsoft.Azure.Cosmos .NET SDK v3.62.0 or later.
- ☐ E Change the account consistency level to bounded staleness before enabling per-partition automatic failover.
- ☐ F Enable multi-region writes on account1 before enabling per-partition automatic failover.

61 Multiple choice · Domain 2

Tailwind Traders has an Azure Cosmos DB for NoSQL account named account1.
account1 has a single write region in East US and read regions in West US and North Europe.
Service-managed failover is enabled, and failover priorities are configured.
East US is experiencing a regional outage.
Write availability must be restored as quickly as possible.
You need to choose the failover action for account1.

Which action should you perform?

NOTE: Each correct selection is worth one point.

- ☐ A Wait for service-managed failover to declare the outage and promote a read region.
- ☐ B Enable multi-region writes on account1 while East US is unavailable.
- ☐ C Restore account1 from continuous backup to a new account in West US.
- ☐ D Perform a forced failover by taking East US offline so the next priority region is promoted.
- ☐ E Use Change write region to move writes from East US to West US during the outage.

SCENARIO SERIES (3 QUESTIONS)

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 on the review screen.

Tailwind Traders has an Azure Cosmos DB for NoSQL account named account1 that contains a database named db1 and a container named container1.

container1 stores product events and is partitioned by /tenantId.

App1 frequently runs the following query: `SELECT * FROM c WHERE c.tenantId = @tenantId ORDER BY c.category ASC, c.updatedAt DESC`

You must reduce the request unit charge of the query.

The returned order must remain category ascending and updatedAt descending.

You need to remediate the query performance problem.

62 SOLUTION

You add a composite index with /category ascending followed by /updatedAt descending, and then wait for the index transformation to complete.

Does the solution meet the goal? Yes ☐ No ☐

63 SOLUTION

You double the provisioned throughput on container1.

Does the solution meet the goal? Yes ☐ No ☐

64 SOLUTION

You exclude /updatedAt/? from the indexing policy because the property is only used for sorting.

Does the solution meet the goal? Yes ☐ No ☐

65 Drop-down selection · Domain 2

Northwind Traders has an Azure Cosmos DB for NoSQL container named container1.

App1 runs the following query frequently: `SELECT * FROM c WHERE c.tenantId = @tenantId AND c.status = @status ORDER BY c.tenantId ASC, c.status ASC, c.score DESC`

Items contain a large `rawText` object that is written by ingestion but is never filtered, projected, or sorted.

You need to complete the indexing policy to optimize the query and reduce write RU consumption for unused data.

How should you complete the indexing policy?

To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

```
{
  "indexingMode": "consistent",
  "automatic": true,
  "includedPaths": [
    { "path": " [A] " }
  ],
  "excludedPaths": [
    { "path": " [B] " }
  ],
  "compositeIndexes": [
    [
      { "path": " [C] ", "order": "ascending" },
      { "path": "/status", "order": "ascending" },
      { "path": "/score", "order": " [D] " }
    ]
  ]
}
```

[A] /* · /tenantId/? · /?

[B] /score/? · /rawText/* · /tenantId/?

[C] /tenantId · /rawText · /status · /score

[D] consistent · descending · ascending

66 Yes or No · Domain 2

Fabrikam enables index metrics for a slow Azure Cosmos DB for NoSQL query by setting `PopulateIndexMetrics` to `true`.

How should you evaluate each statement?

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

NOTE: Each correct selection is worth one point.

Statement	Yes	No
If <code>/price/?</code> appears under Utilized Single Indexes, the query used that indexed path, and removing it can affect the query.	☐	☐
A path under Potential Composite Indexes is conclusive proof that the query will use that index after it is added.	☐	☐
<code>PopulateIndexMetrics</code> should be enabled only while troubleshooting slow queries because collecting index metrics adds overhead.	☐	☐

67 Drop-down selection · Domain 2

Proseware sends Azure Cosmos DB diagnostic logs to a Log Analytics workspace by using resource-specific tables.

The `QueryRuntimeStatistics` and `DataPlaneRequests` categories are enabled.

You need to identify the top 10 query texts by request unit charge in the last 24 hours.

How should you complete the KQL query?

To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

```
let topRequestsByRUcharge = [A]
| where TimeGenerated > ago(24h)
| project RequestCharge, TimeGenerated, ActivityId;
[B]
| project QueryText, ActivityId, DatabaseName, CollectionName
| join kind=inner topRequestsByRUcharge on ActivityId
| project DatabaseName, CollectionName, QueryText, RequestCharge, TimeGenerated
| order by [C] desc
| take 10
```

[A] `CDBQueryRuntimeStatistics · AzureDiagnostics · CDBDataPlaneRequests`

[B] `CDBDataPlaneRequests · CDBQueryRuntimeStatistics · CDBPartitionKeyRUConsumption`

[C] `CollectionName · TimeGenerated · RequestCharge`

68 Build list · Domain 2

Woodgrove Bank has an Azure Cosmos DB for NoSQL account named account1.

You use Azure Monitor to monitor account1.

You need to receive an email notification when account1 has more than 100 rate-limited requests.

Which five 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.

ACTIONS

- ☐ A Select the Total Request Units signal as a metric condition.
- ☐ B Enable the QueryRuntimeStatistics diagnostic category before creating the metric alert.
- ☐ C Select or create an email action group, enter alert rule details, and create the rule.
- ☐ D Configure account-level manual failover before creating the alert rule.
- ☐ E Create a new alert rule and select account1 as the scope.
- ☐ F Add a StatusCode dimension filter with the custom value 429.
- ☐ G Set a static threshold with Total aggregation, Greater than, and a value of 100.

69 Multiple choice · Domain 2

Fabrikam has an Azure Cosmos DB for NoSQL container named container1.

App1 runs the following query: `SELECT VALUE c.id FROM c WHERE UPPER(c.description) = @description`

The query metrics show Index Utilization of 0 percent, Retrieved Document Count of 68,000, and Output Document Count of 25.

You need to reduce the RU charge for the query.

Which change should you make?

NOTE: Each correct selection is worth one point.

- ☐ A Set MaxItemCount to -1 for the query iterator.
- ☐ B Store an uppercase description value at write time and filter on that indexed value.
- ☐ C Increase the provisioned throughput of container1.
- ☐ D Enable PopulateIndexMetrics for all production executions of the query.

70 Multiple choice · Domain 2

Adventure Works has an Azure Cosmos DB for NoSQL account named account1 and a .NET application named App1.

App1 uses direct connectivity for point reads and has an end-to-end P99 latency objective of 75 ms.

Application logs show point reads with client elapsed times of 250 ms. Azure Monitor shows Server Side Latency Direct remains below 10 ms for the same period.

You need to identify the next diagnostic data to collect.

What should you collect?

NOTE: Each correct selection is worth one point.

- ☐ A The Server Side Latency Gateway metric for account1.
- ☐ B The CDBQueryRuntimeStatistics rows for query text.
- ☐ C The CosmosDiagnostics output from slow SDK responses.
- ☐ D The IndexTransformationProgress value for container1.

71 Multiple response · Domain 2

Litware has an Azure Cosmos DB for NoSQL container named container1.

Azure Monitor shows Normalized RU Consumption reaching 100 percent for one PartitionKeyRangeId several times per hour.

The Total Requests by Status Code chart shows that 3 percent of requests receive HTTP 429 responses, and App1 meets its end-to-end latency objective.

You need to interpret the monitoring data.

Which two statements are accurate?

Each correct answer presents a complete solution. Choose two.

NOTE: Each correct selection is worth one point.

- ☐ A Increasing overall throughput always fixes a hot logical partition permanently.
- ☐ B A 100 percent normalized RU value means at least one partition key range used all of its allocated RU/s.
- ☐ C A 100 percent normalized RU value proves every physical partition is saturated.
- ☐ D The Azure Cosmos DB SDK never retries 429 responses, so all 429 responses reach App1.
- ☐ E No immediate throughput increase is required solely because the 429 rate is 3 percent and latency is acceptable.

72 Multiple choice · Domain 2

Coho Winery has an Azure Cosmos DB for NoSQL container named container1 that is partitioned by /tenantId.

Items are approximately 1 KB. App1 retrieves a single item when it already knows the item id and tenantId.

App1 currently uses a SQL query that filters on id and tenantId. Reads must not return stale data.

You need to minimize the RU charge for each read operation.

Which approach should you use?

NOTE: Each correct selection is worth one point.

- ☐ A Use a point read with the item id and the tenantId partition key value.
- ☐ B Add a composite index on /tenantId ascending and /id ascending.
- ☐ C Set MaxConcurrency to -1 on the query request options.
- ☐ D Use the integrated cache with MaxIntegratedCacheStaleness set to 10 years.

73 Drop-down selection · Domain 2

You are designing Azure Cosmos DB fleets for a multitenant application used by Relecloud.

You need to identify the fleet concept that matches each description.

Which concept should you choose for each description?

To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Description	Concept
Top-level resource that corresponds to one multitenant application and organizes accounts across subscriptions or resource groups.	Fleet pool · Fleetspace account · Fleet · Fleetspace
Logical grouping where database accounts can optionally share RU/s.	Fleetspace account · Fleet · Fleet pool · Fleetspace
Shared RU/s capacity used when resources exceed their dedicated RU/s.	Fleet · Fleet pool · Fleetspace account · Fleetspace
Azure Cosmos DB database account registered within a fleetspace.	Fleet · Fleet pool · Fleetspace account · Fleetspace

Choose one option in each row.

74 Yes or No · Domain 2

You have an Azure Cosmos DB fleet for Tailspin Toys. A fleetspace is configured with throughput pooling.

For each statement, select Yes or No.

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

NOTE: Each correct selection is worth one point.

Statement	Yes	No
A resource consumes its dedicated RU/s before it draws from the fleetspace pool.	☐	☐
Resources in accounts from different subscriptions and resource groups can share pool RU/s when the accounts are in the same fleetspace.	☐	☐
A single physical partition can consume unlimited extra RU/s from the pool when the pool maximum is high enough.	☐	☐

75 Multiple response · Domain 2

You are creating a fleetspace with throughput pooling for a fleet used by Lucerne Publishing.

The fleetspace will contain tenant accounts that can draw from the same pool.

You need to meet the documented configuration rules for throughput pooling.

Which two requirements must the accounts in the fleetspace meet?

Each correct answer presents part of the solution. Choose two.

NOTE: Each correct selection is worth one point.

- ☐ A Configure the pool maximum RU/s to be lower than the pool minimum RU/s.
- ☐ B Configure shared RU/s separately on each tenant container instead of at the fleetspace.
- ☐ C Place all accounts in the same Azure subscription and resource group.
- ☐ D Mix single-write and multi-write accounts in the same throughput pool.
- ☐ E Use the same underlying single-write or multi-write service tier configuration.
- ☐ F Use the same regional configuration for all accounts in the fleetspace.

76 Multiple choice · Domain 2

You manage hundreds of Azure Cosmos DB for NoSQL accounts across several subscriptions for Alpine Ski House.

Operations wants long-term trend analysis for usage, provisioning, and costs across the fleet.

The solution must provide fleet-wide aggregated data at an hourly grain.

Analysts must be able to use Fabric or data lake analytics tools.

You need to choose a monitoring solution.

Which solution should you use?

NOTE: Each correct selection is worth one point.

- ☐ A Azure Monitor metrics for each account, analyzed in Metrics Explorer
- ☐ B Azure Monitor logs for each account, analyzed in a Log Analytics workspace
- ☐ C The FleetspaceAutoscaledThroughput metric split by capacity type
- ☐ D Fleet Analytics exported as Delta Lake tables to Fabric OneLake or ADLS Gen2

77 Multiple response · Domain 2

You enable Fleet Analytics for an Azure Cosmos DB fleet used by Fourth Coffee.

You select Send to Fabric workspace and save the destination. The Fabric workspace uses OneLake as the default storage location and is backed by a licensed Fabric capacity.

Data must begin landing in the Fabric workspace.

You need to set the correct expectation for initial data availability.

You need to complete the enablement process.

Which two actions should you perform?

Each correct answer presents part of the solution. Choose two.

NOTE: Each correct selection is worth one point.

- ☐ A Regenerate the primary keys on all accounts before the first export.
- ☐ B Assign Cosmos DB Built-in Data Contributor to the Fabric workspace managed identity on every tenant container.
- ☐ C Add the Cosmos DB Fleet Analytics service principal as a Contributor in the Fabric workspace.
- ☐ D Create Azure Monitor diagnostic settings that send every account to Log Analytics.
- ☐ E Enable analytical store on every container in the fleet.
- ☐ F Allow up to one hour for data to start populating the storage location.

SCENARIO SERIES (3 QUESTIONS)

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 on the review screen.

You have an Azure Cosmos DB for NoSQL account named account1 that contains a database named db1 and a container named container1.

container1 stores product documents for Contoso. Each document has text in c.description and a vector embedding in c.embedding.

Search results must combine keyword relevance for a user query with semantic similarity to the query embedding.

The ranking must be a single fused ranking returned by Azure Cosmos DB for NoSQL.

You need to implement the query for App1.

78 SOLUTION

You use `SELECT TOP 10 * FROM c ORDER BY RANK RRF(VectorDistance(c.embedding, @embedding), FullTextScore(c.description, @term1, @term2))`.

Does the solution meet the goal? Yes ☐ No ☐

79 SOLUTION

You use `SELECT TOP 10 * FROM c WHERE FullTextContains(c.description, @term1) ORDER BY VectorDistance(c.embedding, @embedding)`.

Does the solution meet the goal? Yes ☐ No ☐

80 SOLUTION

You use `SELECT TOP 10 * FROM c ORDER BY RANK FullTextScore(c.description, @term1, @term2)`.

Does the solution meet the goal? Yes ☐ No ☐

81 Drop-down selection · Domain 3

Fabrikam stores support article chunks in container1. Each item contains a text embedding in /embedding. The embeddings are 1,536 dimensions.

App1 runs production searches over more than 50,000 vectors per tenant and must reduce vector storage compared with float32 while keeping indexed vector search.

You need to define the vector embedding policy and indexing policy for a new container.

How should you complete the JSON configuration?

To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

```
{ "vectorEmbeddings": [
  {
    "path": "[A]", "dataType": "[B]",
    "distanceFunction": "cosine", "dimensions": 1536
  }
] }
{
  "includedPaths": [{ "path": "/*" }],
  "excludedPaths": [{ "path": "/_etag/" }, { "path": "[C]" }],
  "vectorIndexes": [{ "path": "/embedding", "type": "[D]" }]
}
```

[A] /description · /embedding/* · /tenantId · /embedding

[B] float16 · string · float32 · double

[C] /embedding · /description/* · /embedding/* · /_etag/?

[D] fullText · quantizedFlat · diskANN · flat

82 Drop-down selection · Domain 3

Northwind Traders has a container named container1 with tenant IDs in c.tenantId and vectors in c.embedding.

App1 receives a query embedding in the parameter @embedding and the tenant in @tenantId.

The query must return only the ten most similar documents for the tenant.

The vector index and the distance function configured in the vector embedding policy should be used.

You need to complete the NoSQL query.

How should you complete the query?

To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

```
SELECT TOP [A] c.id, c.title,  
           VectorDistance(c.embedding, @embedding) AS score  
FROM c  
WHERE c.tenantId = @tenantId  
ORDER BY [B] (c.embedding, @embedding)
```

[A] 0 * 10

[B] RRF · FullTextScore · VectorDistance · FullTextContains

83 Drag and drop · Domain 3

Tailwind Traders plans three vector-enabled containers for App1.

You need to choose the vector index type for each container.

Which vector index type should you use for each requirement?

To answer, drag the appropriate values to the correct targets. Each value may be used 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.

ANSWER CHOICES

- 1 fullTextIndexes
- 2 flat
- 3 diskANN
- 4 quantizedFlat

Requirement

Vector index type

A small, filtered search scope uses 384-dimensional vectors and requires exact nearest-neighbor recall.

A tenant-scoped workload usually searches fewer than 50,000 vectors and can accept a small recall tradeoff for lower RU cost than exact search.

A production document search runs over more than 50,000 vectors per search scope and needs the lowest latency and RU cost with high recall.

84 Build list · Domain 3

Adventure Works uses Azure Cosmos DB for NoSQL as an operational vector store for product manuals.

App1 answers employee questions by using Azure OpenAI.

Responses must be grounded in the most relevant documents stored in Azure Cosmos DB.

You need to implement the request flow for a RAG answer.

Which five 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.

ACTIONS

- A Train the embedding model inside Azure Cosmos DB.
- B Build a prompt that includes the retrieved items as grounding context.
- C Run a vector or hybrid query against Azure Cosmos DB to retrieve the TOP relevant items.
- D Generate an embedding for the user question.
- E Return the generated answer to the user.
- F Send the grounded prompt to the LLM completion deployment.
- G Use only the model pretrained knowledge without retrieval.

85 Yes or No · Domain 3

Woodgrove Bank has a multitenant container with a DiskANN vector index.

Documents include tenantId and embedding properties.

You need to evaluate whether sharded DiskANN is configured and queried correctly.

For each statement, should you select Yes or No?

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

NOTE: Each correct selection is worth one point.

Statement	Yes	No
Adding vectorIndexShardKey to a DiskANN vector index can create separate DiskANN indexes for each unique value of the specified property.	☐	☐
To focus a vector search on a shard, include a WHERE filter on the shard key value, such as c.tenantId = @tenantId.	☐	☐
If vectorIndexShardKey is omitted, Azure Cosmos DB creates one DiskANN index per unique tenant ID value by default.	☐	☐

86 Multiple choice · Domain 3

Litware has container1 with support article text in c.body.

Full-text search is enabled for Azure Cosmos DB for NoSQL.

Queries that use FullTextContains and FullTextScore must avoid unnecessary RU consumption and latency.

You need to configure container1 before loading data.

Which configuration should you use?

NOTE: Each correct selection is worth one point.

- ☐ A Only an includedPaths entry for /body/? in the regular indexing policy.
- ☐ B A container full-text policy for /body and a full-text index for /body in the indexing policy.
- ☐ C A vector embedding policy for /body and a DiskANN vector index.
- ☐ D Only a composite index on /body and /_ts in the indexing policy.

87 Multiple response · Domain 3

Wide World Importers is creating a new container for item descriptions.

Each document includes a 1,536-dimensional embedding and several text fields.

The document including the vector must fit within Azure Cosmos DB for NoSQL item limits.

You want to reduce vector storage from the model float32 output while preserving most text retrieval quality.

Searches will run over more than 50,000 vectors in each search scope and should use an index that supports the embedding dimensions.

You need to choose the vector storage and indexing decisions.

Which two decisions should you make?

Each correct answer presents part of the solution. Choose two.

NOTE: Each correct selection is worth one point.

- ☐ A Set the vector dataType to decimal to preserve exact numeric precision.
- ☐ B Use a flat vector index for the /embedding path.
- ☐ C Plan item size as unlimited because the vector path is excluded from regular indexing.
- ☐ D Use a diskANN vector index for the /embedding path.
- ☐ E Set the vector dataType to float16 when the recall tradeoff is acceptable.
- ☐ F Store the embedding as a two-dimensional array.

88 Multiple choice · Domain 3

Fabrikam Residences uses a DiskANN index on /embedding in container1.

App1 issues tenant and category filtered vector searches.

The queries must reduce RU cost by giving more priority to scalar filters when selecting candidates.

A small recall tradeoff is acceptable, and you can compensate if needed by increasing candidate list multipliers.

You need to set a VectorDistance query-time option.

Which option should you use?

NOTE: Each correct selection is worth one point.

- ☐ A indexingSearchListSize
- ☐ B vectorIndexShardKey
- ☐ C quantizerType
- ☐ D filterPriority

Coho Winery is building App1, a customer service chatbot.

Product and policy documents are stored in Azure Cosmos DB for NoSQL with embeddings.

Answers must use current information from the database instead of relying only on model training data.

The design should minimize hallucinations by grounding the prompt.

You need to choose the retrieval design.

Which design should you use?

NOTE: Each correct selection is worth one point.

- ☐ A Send the user question directly to the LLM and store the response after it is generated.
- ☐ B Embed the user question, retrieve relevant documents from Azure Cosmos DB with vector or hybrid search, and pass the retrieved content to the LLM as grounding.
- ☐ C Run a point read for a fixed document ID and use that document for all prompts.
- ☐ D Use only FullTextContains to return every matching document to the user without LLM integration.
- ☐ E Use only the chat history cache as grounding and skip the document collection.

You have an agent memory store in Azure Cosmos DB for NoSQL for App1.

App1 uses Agent Memory Toolkit to store conversation history and derived memories.

You need to map each piece of agent data to the memory type that the toolkit stores.

Which memory type should you use for each data element?

To answer, drag the appropriate values to the correct targets. Each value 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.

ANSWER CHOICES

- ① fact
- ② user_summary
- ③ turn
- ④ summary

Agent data	Memory type
A raw record of one user message, agent reply, tool result, or system message that App1 might replay for recent conversation history	
A compact record of one long support thread that captures decisions, open issues, and next steps	
A discrete user requirement extracted from a thread, such as the user prefers invoices grouped by project	
A cross-thread profile for a single user that preserves stable preferences and constraints across sessions	

91 Drop-down selection · Domain 3

Contoso is designing Azure Cosmos DB containers for a multitenant agent memory system.

The containers store JSON memory items that are queried by different access patterns.

You need to select a partition key strategy for each container.

Which partition key should you use for each container?

To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Container requirement	Partition key
conversationTurns: Most reads retrieve the latest turns for one conversation thread.	/threadId · /pk populated with a new GUID per item · ["/tenantId", "/threadId"] · /timestamp
tenantMemories: Queries often filter by tenant and then retrieve one thread within that tenant.	["/tenantId", "/threadId"] · /pk populated with a new GUID per item · /threadId · /timestamp
telemetryTurns: Writes are high volume, and the app rarely revisits a specific conversation.	/pk populated with a new GUID per item · ["/tenantId", "/threadId"] · /timestamp · /threadId

Choose one option in each row.

92 Multiple choice · Domain 3

Fabrikam builds a customer service agent that stores conversation state in Azure Cosmos DB for NoSQL.

Each exchange includes a user prompt, an agent reply, and sometimes a tool call and tool result.

The app frequently retrieves recent exchanges, vectorizes a whole exchange, and expires older exchanges. Conversations can grow for weeks.

You need to choose the recommended data model for the chat history container.

Which data model should you use?

NOTE: Each correct selection is worth one point.

- ☐ A One document per response that stores each user, agent, and tool message separately.
- ☐ B One document per user that embeds every thread for that user.
- ☐ C One document per turn that stores the related messages for that exchange.
- ☐ D One document per thread that appends every message to a single array.

93 Multiple response · Domain 3

Woodgrove Bank stores long-term agent memory observations in Azure Cosmos DB for NoSQL.

Users often ask questions that use different wording than the stored observations.

You need to implement semantic memory retrieval across sessions.

Which two actions should you perform?

Each correct answer presents part of the solution. Choose two.

NOTE: Each correct selection is worth one point.

- ☐ A Generate an embedding for each memory observation and store it in a top-level field configured for vector indexing.
- ☐ B Generate an embedding for the current context and query memories by using VectorDistance scoped to the authorized user or thread.
- ☐ C Use a new GUID as the partition key for every memory and omit user or thread filters from retrieval queries.
- ☐ D Store all memories for a thread in one document and update the array after every turn.
- ☐ E Store only raw conversation turns and skip summaries, facts, and extracted observations.
- ☐ F Use full-text search only on the memory content so retrieval depends on exact terms.

94 Multiple choice · Domain 3

Tailwind Traders uses Agent Memory Toolkit for an agent in App1.

App1 writes new turn records during active conversations. The product team wants summaries, facts, and user summaries to be created automatically after new turns are stored.

The implementation must be robust for production and must not slow the request path.

You need to choose the processing model for the derived memories.

Which processing model should you use?

NOTE: Each correct selection is worth one point.

- ☐ A Run memory processing manually in the web request every time a user opens a chat.
- ☐ B Use Azure Durable Functions with the Azure Cosmos DB change feed to process new turns in the background.
- ☐ C Mirror the memory container into Fabric and generate summaries from the SQL analytics endpoint.
- ☐ D Store all turns in local in-memory storage and upload them only when the app shuts down.

95 Multiple choice · Domain 3

Proseware retrieves 20 relevant semantic memories for a user session, but only five memories fit in the model context window.

A recent session changed a user preference last week. Several older profile facts have high importance scores. You need to inject memories that preserve both recent updates and important long-term context.

Which selection strategy should you use?

NOTE: Each correct selection is worth one point.

- ☐ A Randomly sample five memories from the retrieved set to avoid bias.
- ☐ B Select only the five most recent memories.
- ☐ C Use a hybrid selection that reserves most slots for similarity and importance ranking and some slots for recent memories.
- ☐ D Select only the five memories with the highest stored importance value.

96 Drop-down selection · Domain 3

Adventure Works is planning operational analytics for several NoSQL applications.

You need to choose the appropriate Azure Cosmos DB option for each requirement.

Which option should you choose for each requirement?

To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Requirement	Option
An existing Azure Cosmos DB for NoSQL account must remain the production system of record, while analysts need near real-time Fabric analytics with no ETL and no RU impact on transactional workloads.	Cosmos DB in Microsoft Fabric · Azure Cosmos DB for NoSQL with mirroring into Microsoft Fabric · Azure Synapse Link for Azure Cosmos DB
A new Fabric-native app needs a simplified NoSQL operational store, Microsoft Entra authentication, Fabric capacity billing, and automatic OneLake availability.	Cosmos DB in Microsoft Fabric · Azure Cosmos DB for NoSQL with mirroring into Microsoft Fabric · Azure Synapse Link for Azure Cosmos DB
A team is starting a new analytics project over existing Azure Cosmos DB operational data and asks whether to use Azure Synapse Link.	Cosmos DB in Microsoft Fabric · Azure Cosmos DB for NoSQL with mirroring into Microsoft Fabric · Azure Synapse Link for Azure Cosmos DB

Choose one option in each row.

Northwind Traders has an Azure Cosmos DB for NoSQL account named account1 that contains a database named db1. Continuous backup is enabled on account1, and account1 allows public network access from all networks.

The team has a Microsoft Fabric workspace on a Fabric capacity. The team will authenticate by using Microsoft Entra ID, and the connecting identity already has the readMetadata and readAnalytics data actions on account1.

You need to configure db1 as a mirrored database in Microsoft Fabric and verify that replication is running.

Which five 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.

ACTIONS

- ☐ A Configure Network ACL Bypass on account1 for the Fabric workspace.
- ☐ B Create a new Azure Cosmos DB for NoSQL connection by using the account endpoint and Organizational account authentication.
- ☐ C Enable Azure Cosmos DB analytical store on every container in db1.
- ☐ D Select db1 and, optionally, the specific containers to mirror.
- ☐ E Select Mirror database.
- ☐ F In the Fabric workspace, create a Mirrored Azure Cosmos DB item.
- ☐ G Select Monitor replication and wait until the status changes to Running.

98 Drop-down selection · Domain 3

Wide World Importers mirrors an Azure Cosmos DB for NoSQL database named OrdersDB into Fabric.

The SQL analytics endpoint exposes a table named OrdersDB_orders. The items column is represented as a JSON string, and some orders do not have an items array.

You need to complete a T-SQL query that includes orders without an items array and extracts the order_id scalar value.

How should you complete the query?

To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

```
SELECT t.id, p.order_id, p.item_description
FROM OrdersDB_orders AS t
[A] OPENJSON(t.items) WITH (
    order_id [B] '$.order_id',
    item_description varchar(200) '$.item_description'
) AS p;
```

[A] INNER JOIN · OUTER APPLY · CROSS APPLY

[B] datetime · nvarchar(MAX) AS JSON · varchar(100)

99 Yes or No · Domain 3

You are reviewing the expected behavior of Azure Cosmos DB mirroring into Microsoft Fabric for a production analytics workload.

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

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

NOTE: Each correct selection is worth one point.

Statement	Yes	No
After the initial synchronization, individual insert, update, and delete operations are replicated into OneLake in near real-time.	☐	☐
The replication process consumes request units from the source Azure Cosmos DB account.	☐	☐
The SQL analytics endpoint for the mirrored database is read-only and cannot manipulate the source data.	☐	☐

100 Multiple choice · Domain 3

Coho Winery mirrors an Azure Cosmos DB for NoSQL account into Microsoft Fabric by using account key authentication.

The operations team regenerates the read-write account key. Soon after, the mirrored database replication status changes to Failed.

You need to restore replication by using the documented approach.

What should you do?

NOTE: Each correct selection is worth one point.

- ☐ A Stop replication, update the Fabric connection credentials with the new key, and restart replication.
- ☐ B Disable continuous backup, then enable it again to reseed the mirror.
- ☐ C Increase provisioned RU/s on the source containers until replication resumes.
- ☐ D Replace the connection with a read-only account key so mirroring can continue securely.

101 Multiple response · Domain 3

Alpine Ski House uses a Fabric notebook to aggregate mirrored sales data into a DataFrame named dfResults.

The team needs to write the aggregated results back to a container named container1 in an Azure Cosmos DB for NoSQL account by using the Cosmos DB Spark connector.

You need to configure the write operation by using documented connector settings.

Which two code elements should you include?

Each correct answer presents part of the solution. Choose two.

NOTE: Each correct selection is worth one point.

- ☐ A Use an INSERT statement against the SQL analytics endpoint for the mirrored database.
- ☐ B Set `spark.cosmos.read.inferSchema.enabled` to true as the only required write option.
- ☐ C Set `spark.cosmos.accountEndpoint`, `spark.cosmos.accountKey`, `spark.cosmos.database`, and `spark.cosmos.container` in the write configuration.
- ☐ D Write the DataFrame by using `format("cosmos.oltp")` and `options(**writeConfig).save()`.
- ☐ E Use a read-only account key for the Spark connector write configuration.
- ☐ F Write the DataFrame by using `format("delta")` because mirrored data is stored in OneLake as Delta.

Litware mirrors an Azure Cosmos DB for NoSQL database into Fabric.

A data scientist wants to use a Spark notebook to explore the mirrored data together with Lakehouse tables. The notebook should query the OneLake replica, not connect directly to the source Azure Cosmos DB account.

You need to make the mirrored tables available to Spark.

What should you do?

NOTE: Each correct selection is worth one point.

- ☐ A Create a Lakehouse, add a Microsoft OneLake shortcut to the mirrored database tables, and query the shortcut tables with Spark SQL.
- ☐ B Use INSERT INTO from the SQL analytics endpoint to copy mirrored rows into a Lakehouse table.
- ☐ C Open Source database from the mirrored item and use Data Explorer as the Spark source.
- ☐ D Install the Cosmos DB Spark connector and read directly from the source account for all analytical queries.

Case study: Margie's Travel

Questions 103 to 107 refer to this case study.

Overview

Margie's Travel is rebuilding its booking platform by using Azure Cosmos DB for NoSQL. The platform includes a web app, a mobile app, and an AI trip-planning agent.

The production Azure Cosmos DB account is named account1. The account is replicated to East US and West Europe.

- The database is named traveldb.
- The application tier is deployed in both Azure regions.
- Customers can create and modify bookings from either region.

Existing environment

- The travelerProfiles container stores one profile item per traveler. Point reads use travelerId as the item id.
- The bookings container stores booking items with travelerId, tripId, status, region, and revision properties.
- The seatHolds container stores temporary seat hold items with flightDate, destinationId, and holdId properties.
- The travelerItineraryViews container stores denormalized itinerary documents for fast reads.
- The leases container is available for change feed processor state.

Business requirements

- Reads of a traveler's own itinerary must always reflect that traveler's latest writes within the user's session.
- The platform must not require strong linearizability across all travelers and all regions.
- During flash sales, a single flightDate can receive a high volume of seat hold writes.
- The AI trip-planning agent must recall recent conversation turns and durable traveler preferences.
- Raw conversation turns older than 30 days must be deleted automatically. Durable preference memories must not expire.

Technical requirements

- Queries against bookings always include travelerId and often include tripId. Some travelers can accumulate more than 20 GB of booking history.
- Seat hold writes must be distributed across partitions. The application can calculate a suffix from holdId when it writes or reads a seat hold.
- Cancellations in bookings are written as status updates, not physical deletes.
- The travelerItineraryViews container must be updated from bookings asynchronously. Eventual consistency for the denormalized view is acceptable.
- The agentMemory container stores one item per conversation turn with tenantId, conversationId, turnIndex, memoryType, content, and embedding properties.

Planned changes

- The tripQuotes container will be created before production launch and will store a numeric revision property.
- Concurrent writes to the same trip quote from different regions must converge to the item with the highest revision value.
- Applications will use the global account endpoint and route traffic to the closest configured region.

- Tenant governance queries for agent memory include tenantId. Recent-turn queries include both tenantId and conversationId.

103 Drop-down selection · Domain 1

CASE STUDY: MARGIE'S TRAVEL

You need to select partition keys for the containers to meet the technical requirements.

Which partition key should you use for each container?

To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Container	Partition key
bookings	hierarchical: /travelerId, /tripId · synthetic: /travelerId_tripId · /travelerId · /destinationId
seatHolds	synthetic: /flightDateHoldSuffix · /destinationId · hierarchical: /destinationId, /flightDate · /flightDate
travelerProfiles	/travelerId · /loyaltyTier · /homeAirport · synthetic: /travelerId_homeAirport

Choose one option in each row.

104 Multiple choice · Domain 1

CASE STUDY: MARGIE'S TRAVEL

You need to configure read consistency to meet the itinerary requirements.

Which consistency level should you configure?

NOTE: Each correct selection is worth one point.

- ☐ A Strong
- ☐ B Bounded staleness
- ☐ C Eventual
- ☐ D Session

105 Drag and drop · Domain 1

CASE STUDY: MARGIE'S TRAVEL

You need to keep the denormalized itinerary container synchronized to meet the technical requirements.

Which implementation should you use for each requirement?

To answer, drag the appropriate values to the correct targets. Each value may be used 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.

ANSWER CHOICES

- 1 Upsert documents in travelerItineraryViews
- 2 Lease container named leases
- 3 All versions and deletes mode with continuous backup
- 4 Change feed processor on bookings in latest version mode
- 5 Scheduled cross-partition query of all bookings
- 6 Pull model readers with application-managed checkpoints

Requirement	Implementation
Detect booking creates, updates, and cancellation markers	
Coordinate processing across multiple worker instances	
Persist the derived read model	

106 Multiple response · Domain 2

CASE STUDY: MARGIE'S TRAVEL

You need to configure the tripQuotes workload to meet the multi-region write requirements.

Which two actions should you perform?

Each correct answer presents part of the solution. Choose two.

NOTE: Each correct selection is worth one point.

- A Enable per-partition automatic failover on a serverless account.
- B Add a custom conflict resolution policy after tripQuotes is created.
- C Configure Strong consistency on account1.
- D Create tripQuotes with a Last Write Wins conflict resolution path of /revision.
- E Enable multi-region writes on account1.
- F Use Bounded Staleness and direct reads to a remote write region.

CASE STUDY: MARGIE'S TRAVEL

You are reviewing the planned agentMemory container for the Margie's Travel case study.

For each statement, select Yes or No.

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

NOTE: Each correct selection is worth one point.

Statement	Yes	No
A hierarchical partition key of /tenantId and /conversationId supports tenant-scoped queries and efficient conversation-scoped reads.	☐	☐
Storing one document per turn is a supported model for recent history and semantic retrieval.	☐	☐
Setting ttl to 2592000 on raw turn items is sufficient when DefaultTimeToLive is null on the container.	☐	☐

Case study: Lamna Healthcare Company

Questions 108 to 112 refer to this case study.

Overview

Lamna Healthcare Company is building a patient-support assistant that answers questions by retrieving approved care guidance documents.

The solution uses an Azure Cosmos DB for NoSQL account named `account1` and a database named `caredb`.

- The assistant is used by patients, support agents, and clinicians.
- All new application authentication must use Microsoft Entra ID.
- A Microsoft Fabric workspace and capacity are available for analytics.

Existing environment

- The `patients` container stores protected health information, including contact email and member identifier fields.
- The `guidance` container stores care guidance documents with `body`, `embedding`, `specialty`, and `documentStatus` properties.
- The `chatSessions` container stores support-session state with `id`, `patientId`, `status`, and `ttl` properties.
- The `guidance` container will contain more than 50,000 approved document embeddings.
- Source Azure Cosmos DB accounts can be configured for continuous backup if required.

Business requirements

- Support agents must see masked PHI values when they query patient records.
- Clinicians must be able to view unmasked PHI values when their role permits it.
- The assistant must query guidance documents but must not write to them.
- Analysts must query operational data without consuming request units on the transactional Azure Cosmos DB account.
- Closed chat sessions must be retained for 30 days and then deleted automatically.

Technical requirements

- Key-based authentication must be disabled for new secure workloads.
- Dynamic Data Masking must be used for PHI fields in the `patients` container.
- The `guidance` container will use a full-text policy for `/body` and a vector embedding policy for `/embedding`.
- Hybrid search must combine vector similarity with keyword relevance and must weight semantic similarity twice as much as keyword relevance.
- Only guidance documents with `documentStatus` equal to `approved` can be returned to the assistant.

Operations requirements

- Operations must receive an alert when requests with status code 429 exceed 100 during the evaluation window.
- Query troubleshooting must use query metrics and index metrics only when investigating slow queries.
- Fabric mirroring must replicate selected containers in near real time and must be monitored until replication is running.
- Nested JSON in mirrored data can be expanded by analysts with T-SQL when needed.
- A chat session can be closed only if its current status is open.

108 Drag and drop · Domain 2

CASE STUDY: LAMNA HEALTHCARE COMPANY

You need to configure security controls to meet the Lamna requirements.

Which control should you use for each requirement?

To answer, drag the appropriate values to the correct targets. Each value may be used 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.

ANSWER CHOICES

- 1 Read-only account key in application settings
- 2 Disable key-based authentication and use Microsoft Entra ID
- 3 Custom data plane role with items/unmask
- 4 Dynamic Data Masking policy on PHI paths
- 5 Cosmos DB Built-in Data Reader role at the guidance container scope
- 6 Azure RBAC Reader role on the resource group

Requirement	Control
Prevent new applications from connecting with account keys	
Allow the assistant API to query guidance documents without writing items	
Allow clinicians to view original PHI values	
Show redacted PHI to support staff without changing stored values	

109 Drop-down selection · Domain 3

CASE STUDY: LAMNA HEALTHCARE COMPANY

You need to complete the query to meet the retrieval requirements.

How should you complete the query?

To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

```
SELECT TOP @k c.id, c.title
FROM c
WHERE c.documentStatus = "approved"
ORDER BY RANK [A] ( [B] (c.embedding, @queryVector), [C] (c.body, @term1, @term2), [2, 1])
```

- [A] RRF · ORDER BY · VectorDistance · FullTextContains
- [B] RRF · FullTextContains · VectorDistance · FullTextScore
- [C] RRF · FullTextScore · FullTextContains · VectorDistance

110 Build list · Domain 3

CASE STUDY: LAMNA HEALTHCARE COMPANY

You need to configure analytics for Lamna without consuming request units on the transactional account.

Which four 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.

ACTIONS

- ☐ **A** In Fabric, create a Mirrored Azure Cosmos DB item in the workspace.
- ☐ **B** Query the source database from Fabric Data Explorer for analyst reports.
- ☐ **C** Enable Azure Synapse Link analytical store on each container.
- ☐ **D** Create a connection to the Azure Cosmos DB for NoSQL account and select the database and containers.
- ☐ **E** Enable continuous backup on the source Azure Cosmos DB account.
- ☐ **F** Select Mirror database and monitor replication until the status is Running.
- ☐ **G** Deploy a change feed processor to write aggregate documents to OneLake.

111 Multiple choice · Domain 2

CASE STUDY: LAMNA HEALTHCARE COMPANY

You need to configure an alert for Lamna's operations team.

Which signal and dimension should you use?

NOTE: Each correct selection is worth one point.

- ☐ **A** Query metrics from the x-ms-documentdb-query-metrics response header
- ☐ **B** Normalized RU Consumption metric split by PartitionKeyRangeId
- ☐ **C** Total Request Units metric with the StatusCode dimension set to 429
- ☐ **D** Index metrics from queries with populate_index_metrics enabled

CASE STUDY: LAMNA HEALTHCARE COMPANY

The code is used for the chatSessions container in the Lamna case study.

You review the following Python azure-cosmos code.

```

01 from azure.cosmos import exceptions
02 operations = [
03     {"op": "replace", "path": "/status", "value": "closed"},
04     {"op": "set", "path": "/ttl", "value": 30 * 24 * 60 * 60}
05 ]
06 try:
07     container.patch_item(
08         item=session_id,
09         partition_key=patient_id,
10         patch_operations=operations,
11         filter_predicate="FROM c WHERE c.status = 'open'")
12 except exceptions.CosmosHttpResponseError:
13     raise

```

For each statement, select Yes or No.

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

NOTE: Each correct selection is worth one point.

Statement	Yes	No
The call sends only the modified paths instead of replacing the whole session document.	☐	☐
The filter predicate causes the patch to fail if the current item is no longer open.	☐	☐
The ttl value on line 04 will be honored when DefaultTimeToLive is null on the container.	☐	☐

Part 2: Answers and explanations

A quick answer key first ("see" means a table answer, shown in full below), then the reasoning for every question.

1	see	2	C	3	A	4	D	5	C	6	BC	7	see	8	see
9	see	10	CE	11	see	12	see	13	see	14	see	15	see	16	see
17	C	18	A	19	C	20	CDF	21	E	22	see	23	see	24	see
25	see	26	D	27	B	28	AEF	29	AF	30	D	31	AB	32	B
33	C	34	see	35	see	36	GADCE	37	see	38	B	39	BE	40	A
41	see	42	CE	43	A	44	C	45	see	46	see	47	DAEB	48	see
49	CE	50	C	51	BC	52	C	53	EF	54	Y	55	N	56	N
57	see	58	see	59	FDEBG	60	BD	61	D	62	Y	63	N	64	N
65	see	66	see	67	see	68	EAFGC	69	B	70	C	71	BE	72	A
73	see	74	see	75	EF	76	D	77	CF	78	Y	79	N	80	N
81	see	82	see	83	see	84	DCBFE	85	see	86	B	87	DE	88	D
89	B	90	see	91	see	92	C	93	AB	94	B	95	C	96	see
97	FBDEG	98	see	99	see	100	A	101	CD	102	A	103	see	104	D
105	see	106	DE	107	see	108	see	109	see	110	EADF	111	C	112	see

1 Drop-down selection · Design and develop database solutions

ANSWER

A user's reads must include the writes made in the same client session.	Session
A leaderboard can lag by at most five seconds or 100 updates across regions.	Bounded staleness
An activity feed must never show writes out of order, but it does not require the latest value.	Consistent prefix

Session consistency provides read-your-writes and related guarantees within a client session. Bounded staleness lets you configure a maximum staleness by versions or time. Consistent prefix ensures reads never see out-of-order writes, while allowing reads that might not include the latest writes.

Microsoft Learn: [Consistency level choices - Azure Cosmos DB](#)

2 Multiple choice · Design and develop database solutions

ANSWER: C

C 7

A physical partition can provide up to 10,000 RU/s and store up to 50 GB. Throughput requires $\text{ceiling}(65,000 / 10,000) = 7$ physical partitions. Storage requires $\text{ceiling}(180 / 50) = 4$ physical partitions. The container needs at least the larger count, which is 7.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Thirteen adds the throughput-based and storage-based counts, but the required minimum is the larger of the two counts.
- **B.** Five partitions would support 50,000 RU/s, which is less than the provisioned throughput.
- **D.** Four partitions would satisfy the storage limit, but they would support only up to 40,000 RU/s.
- **E.** Six partitions would support 60,000 RU/s, which is still less than the provisioned throughput.

Microsoft Learn: [Partitioning and horizontal scaling - Azure Cosmos DB](#)

3 Multiple choice · Design and develop database solutions

ANSWER: A

A 1,200 RU/s

A point read of a 1-KB item by ID and partition key costs approximately 1 RU. Strong and bounded staleness consistency consume approximately twice as many RUs for read operations compared to relaxed consistency levels. The estimate is $600 \text{ reads per second} \times 1 \text{ RU} \times 2 = 1,200 \text{ RU/s}$.

WHY THE OTHER CHOICES ARE WRONG

- **B.** This estimate uses 1 RU per point read, but it does not account for the approximate doubling for strong consistency reads.
- **C.** This estimate doubles the already adjusted requirement and overestimates the documented consistency effect.
- **D.** This estimate is lower than the documented cost of the point reads before applying the consistency effect.

Microsoft Learn: [Request Units as a throughput and performance currency - Azure Cosmos DB](#)

4 Multiple choice · Design and develop database solutions

ANSWER: D

D Set `ItemRequestOptions.ConsistencyLevel` to `Eventual` on the dashboard read requests.

The account default consistency applies to all databases and containers under the account by default. Clients can override the default consistency on a per-request basis for reads by using request options. With the traditional consistency override, the SDK can relax consistency for the targeted read requests while the account default remains unchanged.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Changing the account default would affect reads that should continue to use the existing default consistency level.
- **B.** Autoscale changes throughput capacity behavior, but it does not lower the read consistency level for a specific request.
- **C.** Changing the partition key path is a data distribution setting and does not override read consistency.

Microsoft Learn: [Manage Consistency - Azure Cosmos DB](#)

5 Multiple choice · Design and develop database solutions

ANSWER: C

- C** CreateDatabaseIfNotExistsAsync and CreateContainerIfNotExistsAsync

The .NET SDK provides CreateDatabaseIfNotExistsAsync to create a database only when it does not already exist. It also provides CreateContainerIfNotExistsAsync to create a container only when it does not already exist. These methods are appropriate for setup code that can run repeatedly.

WHY THE OTHER CHOICES ARE WRONG

- **A.** GetDatabase and GetContainer return client-side references, but they do not create missing resources.
- **B.** The create methods throw an exception when a database or container with the same name already exists.
- **D.** Read methods validate existing resources, but they do not create missing resources.

Microsoft Learn: [Create a Database Using .NET - Azure Cosmos DB](#) · [Create a Container Using .NET - Azure Cosmos DB](#)

6 Multiple response · Design and develop database solutions

ANSWERS: B, C

- B** Exclude the unused payload and debug paths from indexing.
- C** Remove redundant large properties from the telemetry documents.

RU charges increase with item size, property count, and the number of indexed properties. Excluding properties that are never used by queries reduces indexing work for writes. Removing redundant large properties reduces item size, which also reduces write RU consumption.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Adding a region changes global distribution and availability, but it does not reduce the RU charge of each write.
- **D.** Strong consistency affects read costs and does not reduce write RU consumption.
- **E.** Point reads cost fewer RUs than queries, so replacing point reads with queries would not reduce RU consumption.
- **F.** Indexing more properties increases write indexing work instead of reducing it.

Microsoft Learn: [Request Units as a throughput and performance currency - Azure Cosmos DB](#)

7 Drag and drop · Design and develop database solutions

ANSWER

A prototype has intermittent requests, long idle periods, and no requirement for multiple Azure regions.	Serverless
A production API has steady traffic near the configured RU/s every hour, and the team wants to manage a fixed RU/s value.	Standard (manual) provisioned throughput
A retail events API has unpredictable spikes and requires provisioned-throughput SLAs without custom scaling code.	Autoscale provisioned throughput
Three low-traffic containers can share capacity, and no individual container needs a predictable throughput guarantee.	Database-level shared throughput

Serverless fits intermittent workloads because billing is based on consumed RUs and no throughput is provisioned. Manual provisioned throughput fits steady, predictable traffic, while autoscale fits variable or unpredictable traffic that still needs provisioned-throughput behavior. Database-level shared throughput can be used when containers can share capacity and predictable performance for a specific container is not required.

Microsoft Learn: [Serverless Consumption-Based Account Type - Azure Cosmos DB](#) · [How to Choose Between Manual and Autoscale - Azure Cosmos DB](#) · [Provision Throughput for Containers and Databases - Azure Cosmos DB](#)

8 Yes or No · Design and develop database solutions

ANSWERS

Container1 can scale between 3,000 RU/s and 30,000 RU/s.	Yes
For each hour, billing is based on the highest RU/s value that the system scaled to during that hour.	Yes
If requests require more than 30,000 RU/s in a second, Azure Cosmos DB automatically scales beyond 30,000 RU/s to avoid rate limiting.	No

With autoscale, the configured maximum throughput is the upper bound. A maximum of 30,000 RU/s gives a range of $0.1 \times 30,000 = 3,000$ RU/s through 30,000 RU/s. Billing uses the highest RU/s reached during each hour, and demand beyond the maximum can be rate-limited.

Microsoft Learn: [Create Containers and Databases with Autoscale Throughput - Azure Cosmos DB](#) · [How to Choose Between Manual and Autoscale - Azure Cosmos DB](#)

9 Drop-down selection · Design and develop database solutions

ANSWER

Credential type	DefaultAzureCredential
Preferred regions option	ApplicationPreferredRegions
Connection mode	ConnectionMode.Direct

The .NET getting-started guidance shows DefaultAzureCredential with a CosmosClient constructor that accepts an endpoint and TokenCredential. The .NET best practices recommend a single CosmosClient instance for the application lifetime and using ApplicationPreferredRegions for preferred regional access. Direct mode is the default and best-performing connectivity mode for the .NET SDK.

Microsoft Learn: [Get Started Using .NET - Azure Cosmos DB](#) · [Best Practices for .NET SDK V3 - Azure Cosmos DB](#) · [ConnectionMode Enum \(Microsoft.Azure.Cosmos\) - Azure for .NET Developers](#)

10 Multiple response · Design and develop database solutions

ANSWERS: C, E

- C** from azure.identity import DefaultAzureCredential
- E** client = CosmosClient(ENDPOINT, DefaultAzureCredential())

The Python getting-started guidance imports DefaultAzureCredential from azure.identity and creates a CosmosClient with the endpoint and credential. DefaultAzureCredential supports local development identities and managed identities in Azure. CosmosClient is the starting point for operations against databases and containers in the Python SDK.

WHY THE OTHER CHOICES ARE WRONG

- **A.** PartitionKey creates a partition key object and cannot be used as an authentication credential.
- **B.** CosmosClient needs a credential for this connection pattern, so the endpoint alone is incomplete.
- **D.** PartitionKey is used for partition key definitions or operations, not for Microsoft Entra authentication.
- **F.** Passing a master key uses key-based authentication rather than Microsoft Entra authentication.

Microsoft Learn: [Get started using Python - Azure Cosmos DB](#) · [Best practices for Python SDK - Azure Cosmos DB](#)

11 Drop-down selection · Design and develop database solutions**ANSWER**

Status operation	Replace
Inventory operation	Increment
Review flag operation	Remove
SDK method	PatchItemAsync

PatchItemAsync applies a list of PatchOperation values to one item identified by id and partition key. PatchOperation.Replace updates the existing status path and fails if that path is missing, PatchOperation.Increment changes the numeric counter, and PatchOperation.Remove deletes the flag path. This avoids the read, modify, and replace pattern for the whole document.

Microsoft Learn: [Partial Document Update - Azure Cosmos DB](#) · [Get Started with Partial Document Update - Azure Cosmos DB](#) · [PatchOperation Class \(Microsoft.Azure.Cosmos\) - Azure for .NET Developers](#)

12 Drop-down selection · Design and develop database solutions**ANSWER**

Request option	IfMatchEtag
Save method	ReplaceItemAsync
Exception type	CosmosException
HTTP status code	PreconditionFailed

Optimistic concurrency uses the item ETag and an If-Match condition so that the service updates the item only when the server-side ETag still matches. If another write changed the item, the service rejects the request with HTTP 412 Precondition Failed. App1 should catch CosmosException for that status and reread the latest item before retrying.

Microsoft Learn: [Database Transactions and Optimistic Concurrency Control - Azure Cosmos DB](#) · [Implement optimistic concurrency control - Training](#)

13 Drop-down selection · Design and develop database solutions

ANSWER

Batch factory method	CreateTransactionalBatch
Add create operation	CreateItem
Execute method	ExecuteAsync
Success check	IsSuccessStatusCode

A transactional batch is created from a container for one PartitionKey value. The batch fluent methods add point operations, ExecuteAsync sends them as one transactional request, and IsSuccessStatusCode lets the application check whether the transaction committed. If any operation fails, the batch rolls back.

Microsoft Learn: [Transactional Batch Operations - Azure Cosmos DB](#) · [Create a transactional batch with the SDK - Training](#)

14 Drop-down selection · Design and develop database solutions

ANSWER

Container method	query_items
Query parameters argument	parameters
Partition scope argument	partition_key

Parameterized queries use placeholders such as @tenantId and a parameters array whose names include the leading @ symbol. In the Python SDK, query_items accepts the SQL text, the parameters list, and a partition_key value to target a single logical partition. This avoids string concatenation while keeping the query scoped to one tenant.

Microsoft Learn: [Parameterized Queries - Query Language for Cosmos DB \(in Azure and Fabric\)](#) · [azure.cosmos.ContainerProxy class](#) · [Query a Container - Azure Cosmos DB](#)

15 Yes or No · Design and develop database solutions

ANSWERS

Line 01 sets 50 as the maximum number of items that the query engine should return in a single query execution.	Yes
Line 09 captures the token that can be supplied on the next request to resume the query when more results remain.	Yes
Because MaxItemCount is 50, every page before the final page is guaranteed to contain exactly 50 orders.	No

MaxItemCount controls the maximum number of items per query execution, but the service can return fewer items for reasons such as throttling, response size, or execution time. Continuation tokens let a client resume stateless query execution when additional results exist. RequestCharge on the feed response reports the RU charge for that page.

Microsoft Learn: [Pagination - Query Language for Cosmos DB \(in Azure and Fabric\)](#) · [Query Items Using .NET - Azure Cosmos DB](#)

16 Drag and drop · Design and develop database solutions**ANSWER**

Retrieve one known session document when App1 has both the id and tenantId values.	Point read with ReadItemAsync
Return all delayed orders for a tenant when the number of matching orders is not known.	Query with GetItemQueryIterator
Create an order header and a reservation item atomically for the same tenantId value.	TransactionalBatch
Ingest 200,000 independent telemetry items across many tenantId values as quickly as possible.	Bulk execution with AllowBulkExecution = true

A point read is the efficient choice when both id and partition key are known for one item. Queries handle result sets found by predicates, transactional batches group same-partition point operations into one transaction, and bulk execution optimizes many concurrent independent operations. The change feed processor is not a direct CRUD, query, or bulk ingestion operation for these requirements.

Microsoft Learn: [Read an Item Using .NET - Azure Cosmos DB](#) · [Transactional Batch Operations - Azure Cosmos DB](#) · [Bulk Import Data for NoSQL Account by Using the .NET SDK - Azure Cosmos DB](#)

17 Multiple choice · Design and develop database solutions**ANSWER: C**

C Use ReadItemAsync with the id and a PartitionKey created from customerId.

A point read uses the item id and the partition key value to retrieve one item directly. Microsoft Learn states that a 1-KB item point read costs 1 RU, while queries are less efficient and their RU cost depends on query work. Because App1 already has both key values, ReadItemAsync is the appropriate operation.

WHY THE OTHER CHOICES ARE WRONG

- **A.** A query that filters by id and partition key is still a query, not a point read, and uses the query engine.
- **B.** Reading all items scans more data than needed and does not use the known key values directly.
- **D.** A cross-partition query and client-side filtering is less efficient than a point read for one known item.

Microsoft Learn: [Read an Item Using .NET - Azure Cosmos DB](#) · [Optimize Request Cost - Azure Cosmos DB](#)

18 Multiple choice · Design and develop database solutions

ANSWER: A

- A** The items never expire because container-level TTL is off, so the item ttl value has no effect.

When DefaultTimeToLive is missing or null, items are not expired automatically. In that state, an item-level ttl property has no effect until TTL is enabled at the container level. Setting DefaultTimeToLive to -1 enables TTL with no default expiration so item-level overrides can expire selected items.

WHY THE OTHER CHOICES ARE WRONG

- **B.** DefaultTimeToLive uses null, -1, or a nonzero value. A value of 0 is not the setting that enables item expiration.
- **C.** The item ttl property by itself does not enable expiration when container TTL is off.
- **D.** A per-item ttl value is honored only when DefaultTimeToLive is present and not null on the container.

Microsoft Learn: [Expire Data With Time to Live - Azure Cosmos DB](#) · [Configure and Manage Time to Live - Azure Cosmos DB](#)

19 Multiple choice · Design and develop database solutions

ANSWER: C

- C** ReplaceItemAsync

ReplaceItemAsync replaces an existing item and requires the id parameter to match the item unique identifier. UpsertItemAsync is used when the application should create a new item or replace an existing item with the same id. Because App1 must fail rather than create a missing product, ReplaceItemAsync is the correct method.

WHY THE OTHER CHOICES ARE WRONG

- **A.** CreateItemAsync creates a new item and throws a conflict if an item with the same id already exists.
- **B.** PatchItemAsync applies partial path changes and is not the method for replacing the full document.
- **D.** UpsertItemAsync creates the item when it does not exist, which violates the requirement to avoid accidental creation.

Microsoft Learn: [Create an Item Using .NET - Azure Cosmos DB](#)

20 Multiple response · Design and develop database solutions

ANSWERS: C, D, F

- C** All operations in the batch must use the same logical partition key value.
- D** The batch can contain at most 100 operations.
- F** The request payload must not exceed 2 MB, and execution time is limited to 5 seconds.

TransactionalBatch groups point operations that must succeed or fail together for the same partition key in a container. The documented limits include a maximum of 100 operations and a request payload that does not exceed 2 MB, with a maximum execution time of 5 seconds. If an operation fails, the failed operation reports its error and the batch rolls back.

WHY THE OTHER CHOICES ARE WRONG

- **A.** TransactionalBatch is for point operations such as create, read, replace, upsert, delete, and patch, not for multi-item SQL queries.
- **B.** TransactionalBatch is scoped to operations with the same logical partition key, not merely the same container.
- **E.** If any operation fails, the entire transactional batch is rolled back.

Microsoft Learn: [Transactional Batch Operations - Azure Cosmos DB](#)

21 Multiple choice · Design and develop database solutions**ANSWER: E**

- E** The SDK converts the concurrent point operations into internal batches, dispatches them across partitions, and maps individual results back to the original tasks.

With AllowBulkExecution enabled, the .NET SDK can group concurrent operations into service calls and distribute them across partitions. The application still uses common item methods such as CreateItemAsync and waits for the tasks, while the SDK maps results back to the original callers. Bulk execution improves throughput utilization for independent operations, but it does not make them a transaction or remove RU charges.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Bulk execution optimizes independent operations but does not provide one cross-partition ACID transaction.
- **B.** Bulk execution optimizes throughput utilization but write operations still consume request units.
- **C.** Bulk support uses the same item operation methods, such as CreateItemAsync, with concurrent tasks.
- **D.** Bulk execution groups and dispatches operations concurrently rather than serializing every create.

Microsoft Learn: [Bulk Import Data for NoSQL Account by Using the .NET SDK - Azure Cosmos DB](#) · [Create bulk operations with the SDK - Training](#)

22 Drop-down selection · Design and develop database solutions**ANSWER**

ChatHistory: thousands of small tenants, each tenant stores less than 2 GB, and all transactional queries filter by tenantId.	/tenantId
ActivityEvents: some tenants can exceed 20 GB, queries filter by tenantId and often userId, and each tenant must scale beyond one logical partition.	/tenantId then /userId then /id (hierarchical)

A partition-key-per-tenant model is recommended for cost-efficient B2C workloads when each tenant fits within a logical partition and tenantId is in the query filter. HPKs support up to three levels and let a tenant prefix exceed the single logical partition limits by using lower levels such as userId and id. Queries that include the first level, or a prefix of the hierarchy, are routed to only the relevant subset of partitions.

Microsoft Learn: [Multitenancy and Azure Cosmos DB - Azure Architecture Center](#) · [Hierarchical Partition Keys - Azure Cosmos DB](#) · [Partitioning and horizontal scaling - Azure Cosmos DB](#)

23 Drop-down selection · Design and develop database solutions**ANSWER**

Orders: writes are heavy at month end for each tenant. Queries usually filter by tenantId and orderMonth. No single existing property has both high cardinality and query alignment.	/tenantId_orderMonth (synthetic concatenation)
ProductReads: every operation is a point read or write by item id. The app rarely filters on other properties and needs maximum write distribution.	/id
DeviceTelemetry: 50,000 devices write once per second, device data can exceed 20 GB, and real-time queries filter by deviceId and a time slice.	/deviceId then /timestamp then /id (hierarchical)

Synthetic keys are useful when no single property provides both high cardinality and query alignment, such as combining tenantId and orderMonth. Using /id is appropriate for workloads that are mostly point reads and writes and rarely run broader filters. For high-volume telemetry where one device can exceed 20 GB, an HPK with deviceId followed by a time value and id preserves query routing while avoiding the single logical partition limit.

Microsoft Learn: [Create a Synthetic Partition Key - Azure Cosmos DB](#) · [Partitioning and horizontal scaling - Azure Cosmos DB](#) · [Choose a partition key for a real-world IoT application - Azure Cosmos DB](#)

24 Drag and drop · Design and develop database solutions**ANSWER**

Order to shipping address. The address is captured at checkout, is read with the order, and does not change after checkout.	Embed the related data in the parent item
Post to comments. A popular post can receive an unbounded number of comments, and comments are paged independently.	Reference the related data as separate items
Portfolio to stock quote. The quote is shared by many portfolios and can update many times per day.	Reference the related data as separate items
Book list to author summary. Each book list always shows the author's name and thumbnail, while the full author item is maintained separately and changes rarely.	Create a denormalized read projection

Azure Cosmos DB guidance favors embedding for contained, one-to-few data that changes infrequently and is queried together. Referencing is better for unbounded, one-to-many, many-to-many, or frequently updated related data. A denormalized read projection duplicates only the fields needed by a read pattern when the source item remains separate.

Microsoft Learn: [Data Modeling - Azure Cosmos DB](#)

25 Yes or No · Design and develop database solutions**ANSWERS**

Using /status as the partition key can create a hot logical partition and can also run into the 20-GB logical partition limit if most items remain Pending.	Yes
In the HPK container, a query that filters by tenantId is routed to the subset of physical partitions that contain that tenant's data.	Yes
In the same HPK container, filtering only by userId avoids cross-partition fan-out because userId is a high-cardinality second-level key.	No

Low-cardinality keys such as status can create uneven RU and storage distribution, which can cause hot partitions and logical partitions that approach 20 GB. HPK queries are efficient when they specify the first-level key or a prefix of the hierarchy. A query that skips tenantId and filters only on userId cannot use the hierarchy prefix and fans out.

Microsoft Learn: [Partitioning and horizontal scaling - Azure Cosmos DB](#) · [Frequently Asked Questions on Hierarchical Partition Keys - Azure Cosmos DB](#)

26 Multiple choice · Design and develop database solutions**ANSWER: D**

D 9

Each physical partition can provide up to 10,000 RU/s and store up to 50 GB. The throughput requirement is $\text{ceil}(85,000 / 10,000) = 9$ physical partitions, and the storage requirement is $\text{ceil}(320 / 50) = 7$ physical partitions. The workload needs the larger value, so the minimum is 9 physical partitions.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Eight partitions provide 80,000 RU/s, which is less than the required 85,000 RU/s.
- **B.** Seven partitions satisfy 320 GB of storage, but they provide only 70,000 RU/s of throughput capacity.
- **C.** Adding the throughput-based and storage-based counts double-counts the requirement. Azure Cosmos DB needs the larger of the two counts.

Microsoft Learn: [Partitioning and horizontal scaling - Azure Cosmos DB](#)

27 Multiple choice · Design and develop database solutions

ANSWER: B

- B** Embed recentReviews in the product item and store each review as a separate item with productId.

The product details page has a bounded read requirement for only the three most recent reviews, so embedding that small projection is efficient. The complete review collection is unbounded, so reviews should be stored as separate items that can be queried by productId and paged. This hybrid model is aligned to both read latency and growth requirements.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Storing only ids avoids the unbounded array but does not satisfy the one-read requirement for the product details page.
- **C.** Embedding all reviews creates an unbounded array inside a single item.
- **D.** Azure Cosmos DB query joins operate within an item, not across containers or top-level items, and this approach adds more round trips.

Microsoft Learn: [Data Modeling - Azure Cosmos DB](#)

28 Multiple response · Design and develop database solutions

ANSWERS: A, E, F

- A** Append a random numeric suffix to a time-based key during writes.
- E** Append a suffix calculated from a queryable value, such as a hash of VIN.
- F** Concatenate tenantId and eventDate into a client-created partitionKey property.

Microsoft documents three synthetic partition key techniques: concatenating multiple properties, appending a random suffix, and appending a precalculated suffix. A random suffix spreads writes but makes direct reads harder because the suffix is unknown. A precalculated suffix can spread writes while allowing the application to compute the specific partition key for a known lookup value.

WHY THE OTHER CHOICES ARE WRONG

- **B.** A constant value sends all items to one logical partition and cannot distribute the workload.
- **C.** Partition key values are immutable after an item is created, so patching the key is not a rebalancing method.
- **D.** A low-cardinality status key is a partitioning anti-pattern, not a synthetic key technique for distribution.

Microsoft Learn: [Create a Synthetic Partition Key - Azure Cosmos DB](#) · [Partitioning and horizontal scaling - Azure Cosmos DB](#)

29 Multiple response · Design and develop database solutions

ANSWERS: A, F

- A** Queries that specify tenantId, or tenantId and userId, are routed to relevant physical partitions.
- F** A tenantId prefix can exceed 20 GB because lower levels create full logical partition paths.

With HPKs, the logical partition is the full partition key path, so a first-level tenant prefix can scale beyond 20 GB by using lower levels. Queries that include the first level or a prefix of the hierarchy can be routed to relevant physical partitions instead of full fan-out. HPKs still require careful cardinality planning, especially for write-heavy workloads.

WHY THE OTHER CHOICES ARE WRONG

- **B.** HPKs support up to three levels, not an unlimited number of levels.
- **C.** A low-cardinality first level can limit writes to a small number of physical partitions and create hot partitions.
- **D.** Filtering only on a middle level of the path is a cross-partition query.
- **E.** Adding HPKs to existing containers is not supported; a new container and data copy are required.

Microsoft Learn: [Hierarchical Partition Keys - Azure Cosmos DB](#) · [Frequently Asked Questions on Hierarchical Partition Keys - Azure Cosmos DB](#)

30 Multiple choice · Design and develop database solutions

ANSWER: D

- D** Create a destination container with /userId, copy the data, and update App1 to use the new container.

Partition key values are immutable, and changing the partition key path is not an in-place metadata update on the existing container. The documented change process creates or selects a destination container with the desired partition key and copies data to it. After the copy is complete, the application starts using the new container and can optionally remove the old one.

WHY THE OTHER CHOICES ARE WRONG

- **A.** A partition key value cannot be changed in place by replacing an item.
- **B.** Unique keys do not change the partition key and do not move items to a new logical partition.
- **C.** Physical partition splits do not change the container's partition key path.

Microsoft Learn: [Partitioning and horizontal scaling - Azure Cosmos DB](#) · [Change partition key - Azure Cosmos DB](#)

31 Multiple response · Design and develop database solutions

ANSWERS: A, B

- A** Use the change feed to process source changes and update denormalized copies or materialized views.
- B** Duplicate author name and aggregate count fields in the post items used by the feed.

Denormalizing fields that are needed by a read pattern can reduce the number of reads required to render a page. Azure Cosmos DB change feed design patterns include using the change feed for data movement and for maintaining materialized views or denormalized copies. This approach trades immediate consistency for efficient reads, which matches the stated tolerance for eventual consistency.

WHY THE OTHER CHOICES ARE WRONG

- **C.** Query-time cross-item joins are not how Azure Cosmos DB models relationships and would add round trips and RU cost.
- **D.** A single user item can grow without bound and would require large rewrites as activity changes.
- **E.** The change feed is for processing changes, not for serving a current feed view to users.

Microsoft Learn: [Model and Partition Data using a Real-World Example - Azure Cosmos DB](#) · [Change Feed Design Patterns - Azure Cosmos DB](#) · [Data Modeling - Azure Cosmos DB](#)

32 Multiple choice · Design and develop database solutions

ANSWER: B

- B** Use the Azure Cosmos DB Cost Estimator and configure data, traffic, operations, consistency, regions, vector operations, and change feed consumers.

The Azure Cosmos DB Cost Estimator is designed to estimate required RU/s and monthly cost for Azure Cosmos DB for NoSQL workloads before deployment. The estimator accepts inputs such as data size, document types, operation rates, query complexity, consistency, regions, vector operations, and change feed consumers. Those inputs match the workload characteristics in the scenario.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Storage alone is not enough because throughput and operation rates drive request unit requirements and cost.
- **C.** Physical partition capacity does not model operation mix, item sizes, query complexity, consistency, regions, or vector operations.
- **D.** Azure Monitor metrics require an existing workload and do not provide a predeployment estimate for the new application.

Microsoft Learn: [Estimate Costs using the Cost Estimator - API for NoSQL - Azure Cosmos DB](#) · [Request Units as a throughput and performance currency - Azure Cosmos DB](#)

33 Multiple choice · Design and develop database solutions**ANSWER: C**

- C** Queries that filter by customerId and orderDate will fan out because they do not filter by the partition key.

Using /id can work for workloads that are mostly point reads and writes, with few broader filters. For mixed query workloads, /id often causes cross-partition queries because application filters such as customerId and orderDate do not include the partition key. A better partition key must align with common query predicates while still distributing storage and RU consumption.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Using /id gives each item its own partition key value, so writes are distributed rather than limited to one key value.
- **B.** Point reads by id and partition key are the most efficient read pattern, but the scenario says the workload rarely uses them.
- **D.** Using /id does not group orders by customer, so a customer is not a single logical partition under this design.

Microsoft Learn: [Partitioning and horizontal scaling - Azure Cosmos DB](#) · [Optimize Request Cost - Azure Cosmos DB](#)

34 Drop-down selection · Design and develop database solutions**ANSWER**

Instance method	WithInstanceName
Lease method	WithLeaseContainer
Builder method	Build
Start method	StartAsync

The .NET change feed processor is created from the monitored container by calling `GetChangeFeedProcessorBuilder`. `WithInstanceName` supplies the unique compute instance name, and `WithLeaseContainer` identifies the container that stores lease state. `Build` returns the `ChangeFeedProcessor` instance, and `StartAsync` starts processing.

Microsoft Learn: [Change Feed Processor - Azure Cosmos DB](#)

35 Drag and drop · Design and develop database solutions

ANSWER

Run a serverless handler that starts automatically when container1 changes and does not require the team to maintain worker infrastructure.	Azure Functions trigger for Azure Cosmos DB
Read only the changes for one tenant partition key and control how quickly App1 polls for more changes.	Pull model with a FeedIterator from GetChangeFeedIterator
Run several .NET worker instances that automatically share leases, checkpoint progress, and load balance changes for the full container.	Change feed processor

Azure Functions is the simplest serverless push model because it uses the change feed processor behind the scenes. The pull model is appropriate when an app needs low-level control, such as reading one partition key value or controlling the polling pace. The change feed processor is the SDK-hosted push model that uses leases for automatic checkpointing and load balancing across workers.

Microsoft Learn: [Read Change Feed - Azure Cosmos DB](#) · [Change Feed Pull Model - Azure Cosmos DB](#) · [Use Change Feed With Azure Functions - Azure Cosmos DB](#)

36 Build list · Design and develop database solutions

CORRECT ORDER: G → A → D → C → E

1. Enable continuous backup and the all versions and deletes change feed feature on account1.
2. Enable the EnableOnlineContainerCopy capability on account1 and wait at least 30 minutes.
3. Create a target container in db1 that uses /tenantId as the partition key and the required settings.
4. Create an online container copy job from container1 to the target container, and monitor the job until it processes the source changes.
5. Stop writes to container1, complete the copy job, and point App1 to the target container.

Online container copy jobs for Azure Cosmos DB for NoSQL require continuous backup and all versions and deletes change feed mode on the source account. The online copy capability must be enabled before an online job is created. After you create the target container with the desired partition key, create and monitor the online job, stop source writes, call the completion API, and move the application to the target container.

WHY THE OTHER CHOICES ARE WRONG

- **B.** The source container must remain available during the copy job so the job can copy existing data and replicate incremental changes.
- **F.** A lease container with a /id partition key is used by change feed processor based workloads, not by container copy jobs.

Microsoft Learn: [Container Copy Jobs - Azure Cosmos DB](#) · [Create and Manage Intra-account Container Copy Jobs - Azure Cosmos DB](#) · [Change partition key - Azure Cosmos DB](#)

37 Yes or No · Design and develop database solutions**ANSWERS**

Latest version mode captures delete operations when an item expires because of time to live.	No
All versions and deletes mode can include delete operations, but account1 must use continuous backup and have the all versions and deletes feature enabled.	Yes
The change feed guarantees one global order for changes across all partition key values in container1.	No

Latest version mode returns the latest item image for creates and updates, and deletes are not captured. All versions and deletes mode adds records for creates, updates, deletes, and TTL expirations, but it depends on continuous backup and is limited to the backup retention window. The sort order guarantee applies within a partition key, not across all partition key values.

Microsoft Learn: [Work with the Change Feed - Azure Cosmos DB](#) · [Change Feed Modes - Azure Cosmos DB](#)

38 Multiple choice · Design and develop database solutions**ANSWER: B**

B /id

The lease container maintains state across multiple and dynamic Azure Function instances. The Azure Functions trigger documentation states that partitioned lease containers are required to have a /id partition key definition. You can create the lease container manually or let the trigger create it automatically when configured to do so.

WHY THE OTHER CHOICES ARE WRONG

- **A.** /processorName can distinguish deployment units conceptually, but it is not the required lease container partition key path.
- **C.** /leaseToken is not the required partition key definition for a partitioned lease container.
- **D.** /partitionKey is a generic property name and is not the required path for a partitioned lease container.

Microsoft Learn: [Use Change Feed With Azure Functions - Azure Cosmos DB](#)

39 Multiple response · Design and develop database solutions

ANSWERS: B, E

- B** Record item delete operations for an audit log.
- E** Detect each intermediate update when the same item changes multiple times between reads.

All versions and deletes mode records creates, updates, deletes, and intermediate changes between reads. Latest version mode returns only the latest version of changed items from creates and updates, and it does not capture deletes. All versions and deletes mode also depends on the continuous backup retention window.

WHY THE OTHER CHOICES ARE WRONG

- **A.** All versions and deletes mode starts from now or a prior checkpoint within the retention period, not from an arbitrary past time for a new reader.
- **C.** All versions and deletes mode requires continuous backup, so this requirement cannot be satisfied by that mode.
- **D.** Latest version mode is the mode designed for current item images, and it has no fixed retention period for available changes.

Microsoft Learn: [Change Feed Modes - Azure Cosmos DB](#)

40 Multiple choice · Design and develop database solutions

ANSWER: A

- A** Capture the continuation token, wait briefly, and poll the iterator again later.

In the pull model, the client explicitly polls the change feed. When there are no new changes, the SDK returns NotModified, and the app should wait before checking again. The continuation token can be saved so the FeedIterator can resume from the last processed point.

WHY THE OTHER CHOICES ARE WRONG

- **B.** The change feed is effectively an infinite list, so no new changes now does not mean processing is permanently complete.
- **C.** Recreating the container is unrelated to normal pull model polling and would not preserve the current feed position.
- **D.** The pull model uses continuation tokens that the application stores, not automatic lease checkpointing.

Microsoft Learn: [Change Feed Pull Model - Azure Cosmos DB](#)

41 Yes or No · Design and develop database solutions

ANSWERS

The Agent Kit is a collection of skills that helps AI coding assistants follow Azure Cosmos DB best practices.	Yes
Before using the recommended installation command, the workstation needs an AI coding assistant that supports Agent Skills and Node.js with npm and npx.	Yes
The best-performing hosts load the entire Agent Kit into every prompt so every rule is always present.	No

Azure Cosmos DB Agent Kit uses Agent Skills so an AI coding assistant can load relevant Azure Cosmos DB guidance. The recommended setup requires an Agent Skills-compatible assistant and Node.js tooling for the npx command. The compatibility guidance favors focused, on-demand skill loading to keep the context relevant.

Microsoft Learn: [Azure Cosmos DB Agent Kit overview](#) · [Azure Cosmos DB for NoSQL](#) · [Set up the Azure Cosmos DB Agent Kit](#) - [Azure Cosmos DB for NoSQL](#) · [Where this works best](#) - [Azure Cosmos DB for NoSQL](#)

42 Multiple response · Design and develop database solutions

ANSWERS: C, E

- C** Apply design patterns for agentic workflows.
- E** Plan indexing strategies for vector workloads.

The Agent Kit usage page lists indexing strategies for common and vector workloads as a best use case. It also lists applying design patterns for agentic workflows, change feed, and materialized views. The kit is scoped to Azure Cosmos DB development guidance, not unrelated Microsoft 365 or Azure SQL administration tasks.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Microsoft Teams retention policy configuration is not an Azure Cosmos DB development task.
- **B.** Exchange Online mail flow rules are not part of the Azure Cosmos DB Agent Kit use cases.
- **D.** Azure SQL Database elastic pools are outside the Azure Cosmos DB guidance areas listed for Agent Kit.

Microsoft Learn: [Use the Azure Cosmos DB Agent Kit](#) - [Azure Cosmos DB for NoSQL](#) · [Azure Cosmos DB Agent Kit overview](#) - [Azure Cosmos DB for NoSQL](#)

43 Multiple choice · Design and develop database solutions

ANSWER: A

A npx skills add AzureCosmosDB/cosmosdb-agent-kit

The setup page states that the recommended option for most users is the single npx command. Other commands are supported entry points for specific agents. After installation, the kit activates automatically when the assistant works on Azure Cosmos DB tasks.

WHY THE OTHER CHOICES ARE WRONG

- **B.** The APM command is a supported entry point, but it is not the recommended single command for most users.
- **C.** This command is specific to Gemini CLI, not the general recommended option for most users.
- **D.** This is one GitHub Copilot CLI plugin command, but it is not the general recommended option for most users and does not include the install command.

Microsoft Learn: [Set up the Azure Cosmos DB Agent Kit - Azure Cosmos DB for NoSQL](#)

44 Multiple choice · Design and develop database solutions

ANSWER: C

C Progressive, on-demand loading of relevant Agent Skills.

Azure Cosmos DB Agent Kit is designed for progressive, on-demand skill delivery. This approach helps keep the context focused and lets the right rule activate for the current task. The compatibility page lists GitHub Copilot, Claude Code, Gemini CLI, Cursor, and other Agent Skills-compatible tools as good fits.

WHY THE OTHER CHOICES ARE WRONG

- **A.** The kit is designed for AI coding assistants, not for a non-assistant migration wizard.
- **B.** A static linter that cannot load skills cannot use the Agent Skills-based kit.
- **D.** The compatibility guidance says the kit works best when the host loads only the relevant skill instead of injecting the full skill set into every prompt.

Microsoft Learn: [Where this works best - Azure Cosmos DB for NoSQL](#)

45 Drop-down selection · Secure, optimize, and deploy database solutions

ANSWER

Credential variable type	TokenCredential
Credential implementation	DefaultAzureCredential

The Azure Cosmos DB for NoSQL .NET SDK can create a CosmosClient by using an account endpoint and a TokenCredential. DefaultAzureCredential from Azure Identity supports managed identities and other Microsoft Entra ID credential sources, so it works after native data plane RBAC is assigned.

Microsoft Learn: [Connect using role-based access control and Microsoft Entra ID - Azure Cosmos DB](#)

46 Drag and drop · Secure, optimize, and deploy database solutions**ANSWER**

Allow App1 to perform point reads, NoSQL queries, and change feed reads without writes.	Cosmos DB Built-in Data Reader
Allow App2 only to create, replace, upsert, and delete items in a container.	Custom native data plane role
Allow an operator to manage account settings without reading data or account keys.	Cosmos DB Operator
Allow a support engineer to manage account resources when key operations are acceptable.	DocumentDB Account Contributor

Azure Cosmos DB for NoSQL native data plane roles are distinct from Azure control plane roles. Cosmos DB Built-in Data Reader grants item reads, queries, and change feed reads, which matches App1. Cosmos DB Built-in Data Contributor grants wildcard container and item data actions, so a custom native data plane role limited to the item write actions is the least privileged choice for App2. Cosmos DB Operator manages accounts without access to data or keys, while DocumentDB Account Contributor manages account resources and can perform key operations.

Microsoft Learn: [Data plane security reference - Azure Cosmos DB](#) · [Azure built-in roles for Databases - Azure RBAC](#) · [Connect using role-based access control and Microsoft Entra ID - Azure Cosmos DB](#)

47 Build list · Secure, optimize, and deploy database solutions**CORRECT ORDER: D → A → E → B**

1. Set publicNetworkAccess to Disabled when account1 is created.
2. Create a private endpoint for the Sql target subresource in subnet1.
3. Integrate the private endpoint with the privatelink.documents.azure.com private DNS zone.
4. Verify that App1 resolves the account endpoint to private endpoint IP addresses.

The private endpoint article states that publicNetworkAccess set to Disabled blocks public and virtual network traffic even if sources are otherwise allowed. Azure Cosmos DB for NoSQL private endpoints use the Sql group ID and the privatelink.documents.azure.com private DNS zone, and DNS resolution determines whether requests use the private endpoint.

WHY THE OTHER CHOICES ARE WRONG

- **C.** The 0.0.0.0 firewall setting allows Azure datacenter IP ranges and is not a private-only configuration.
- **F.** Allowing connections from Azure datacenters permits a wide Azure source range and does not meet private-only access.

Microsoft Learn: [Configure Azure Private Link - Azure Cosmos DB](#) · [Secure your account - Azure Cosmos DB](#)

48 Yes or No · Secure, optimize, and deploy database solutions

ANSWERS

For users without unmask permission, DDM masks sensitive values in read and query responses while the stored data remains unchanged.	Yes
A client that uses account keys can access containers protected by DDM.	No
A user without unmask permission can read both latest mode and all versions and deletes mode change feeds from a masked container.	No

Dynamic Data Masking limits exposure by redacting data for nonprivileged users at read and query time. It relies on Azure Cosmos DB data plane RBAC for the unmask data action, and account-key access is not supported for DDM-protected access.

Microsoft Learn: [Dynamic Data Masking \(DDM\) - Azure Cosmos DB](#) · [Data plane security reference - Azure Cosmos DB](#)

49 Multiple response · Secure, optimize, and deploy database solutions

ANSWERS: C, E

- C** Assign App1 the Cosmos DB Built-in Data Contributor role at /dbs/db1/colls/container1.
- E** Set properties.disableLocalAuth to true on account1.

Disabling key-based authentication on an Azure Cosmos DB account requires setting disableLocalAuth to true. After keys are disabled, App1 still needs native data plane RBAC permissions, such as Cosmos DB Built-in Data Contributor at the database or container scope it uses.

WHY THE OTHER CHOICES ARE WRONG

- **A.** A firewall rule controls network origin but does not grant Microsoft Entra ID data plane permissions or disable keys.
- **B.** Using a key from Key Vault still uses key-based authentication, which must be disabled.
- **D.** DocumentDB Account Contributor is an Azure control plane role and is broader than the data plane access App1 needs.

Microsoft Learn: [Connect using role-based access control and Microsoft Entra ID - Azure Cosmos DB](#) · [Data plane security reference - Azure Cosmos DB](#)

50 Multiple choice · Secure, optimize, and deploy database solutions

ANSWER: C

- C** An Azure Private Link private endpoint with private DNS

Azure Private Link exposes the Azure Cosmos DB account through private endpoint IP addresses in a virtual network. The documentation states that resources mapped to Private Link can also be accessed from on-premises over private peering through VPN or Azure ExpressRoute.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Accepting connections from Azure datacenters allows a broad Azure source range and does not provide private endpoint connectivity.
- **B.** A service endpoint identifies a subnet for access control, but the requirement includes private IP access from on-premises through private peering.
- **D.** IP firewall rules still authorize access to the public endpoint from approved public IP addresses.

Microsoft Learn: [Configure Azure Private Link - Azure Cosmos DB](#) · [Configure IP Firewall - Azure Cosmos DB](#)

51 Multiple response · Secure, optimize, and deploy database solutions

ANSWERS: B, C

- B** Scope: /dbs/db1/colls/container1
- C** Role definition: Cosmos DB Built-in Data Reader

The Cosmos DB Built-in Data Reader role includes readMetadata, item read, executeQuery, and readChangeFeed data actions, and does not include item writes. Data plane role assignments can be scoped to the entire account, a database, or a specific container by using a relative scope such as /dbs/db1/colls/container1.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Cosmos DB Operator manages account resources but does not provide item read and query permissions.
- **D.** Azure Reader is a control plane role and does not grant item reads or query execution.
- **E.** Native data plane RBAC scopes are within the Azure Cosmos DB account hierarchy, such as account, database, or container.

Microsoft Learn: [Connect using role-based access control and Microsoft Entra ID - Azure Cosmos DB · Data plane security reference - Azure Cosmos DB](#)

52 Multiple choice · Secure, optimize, and deploy database solutions

ANSWER: C

- C** Store the endpoint and key as Azure Key Vault secrets and reference them from App Service settings.

The Key Vault guidance recommends managed identity first, and then Key Vault when managed identity or certificate-based approaches do not meet the need. The article shows storing the Azure Cosmos DB endpoint and key as Key Vault secrets and injecting references into Azure App Service app settings.

WHY THE OTHER CHOICES ARE WRONG

- **A.** A firewall rule limits network origin but does not protect the endpoint and key from exposure in configuration.
- **B.** Cosmos DB Operator is a control plane role and does not remove the need to protect the key used by the app.
- **D.** A clear text configuration file does not meet the requirement to avoid storing secrets in configuration files.

Microsoft Learn: [Tutorial: Store and use Azure Cosmos DB credentials with Azure Key Vault - Azure Cosmos DB](#)

53 Multiple response · Secure, optimize, and deploy database solutions

ANSWERS: E, F

- E** Add excludedPaths entries for /id and /projects/{}/projectId.
- F** Add an includedPaths entry with path /.

To mask every field by default, the policy includes the root path /. The documentation supports excluded paths only when / is included, and array elements use the {} marker for a field such as /projects/{}/projectId rather than a specific numeric index.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Masking values on specific array indexes is not supported.
- **B.** A DDM path cannot end with the array marker {}.
- **C.** If the ID or partition key is masked, the document view in Data Explorer is not available.
- **D.** Excluded paths are supported only when all paths are included in the masking policy.

Microsoft Learn: [Dynamic Data Masking \(DDM\) - Azure Cosmos DB](#)

54 Yes or No · Secure, optimize, and deploy database solutions

ANSWER: YES

Solution: You enable multi-region writes on account1. In each App1 deployment, you configure the Azure Cosmos DB SDK to prefer the local account region first.

Yes. Multi-region writes enable every configured region to serve reads and writes, and the SDK can route operations to the preferred local region. During a region outage, correctly configured SDKs route operations to healthy regions without an account-level failover operation.

Microsoft Learn: [Distribute Data Globally - Azure Cosmos DB](#) · [Configure Multi-Region Writes - Azure Cosmos DB](#) · [Reliability in Azure Cosmos DB](#)

55 Yes or No · Secure, optimize, and deploy database solutions

ANSWER: NO

Solution: You add West Europe as a read region, enable service-managed failover, and leave multi-region writes disabled.

No. A single-write account directs writes to the current write region and does not make each healthy region writable during normal operations. Service-managed failover can eventually promote another region, but it is an account-level failover path and can take significant time during an outage.

Microsoft Learn: [Reliability in Azure Cosmos DB](#) · [Manage by Using the Azure Portal - Azure Cosmos DB](#)

56 Yes or No · Secure, optimize, and deploy database solutions

ANSWER: NO

Solution: You keep account1 in a single Azure region and configure continuous backup with the 7-day tier.

No. Continuous backup provides point-in-time restore for recovery from accidental changes or deletions, but it does not keep the application accepting writes during a regional outage. A single-region account loses read and write access during a region-wide outage until the service recovers or the account is restored elsewhere.

Microsoft Learn: [Online Backup and On-Demand Data Restore - Azure Cosmos DB](#) · [Continuous Backup with Point-in-Time Restore - Azure Cosmos DB](#) · [Reliability in Azure Cosmos DB](#)

57 Yes or No · Secure, optimize, and deploy database solutions**ANSWERS**

Line 01 uses the required parameter list for a custom conflict resolution stored procedure.	Yes
The container policy must set Mode to ConflictResolutionMode.Custom and ResolutionProcedure to dbs/db1/colls/container1/sprocs/resolver when Azure Cosmos DB should invoke this procedure automatically.	Yes
If resolver throws an exception at runtime, Azure Cosmos DB hides the conflict from the conflicts feed.	No

Custom conflict resolution is available for Azure Cosmos DB for NoSQL and is configured when the container is created. Last-writer-wins uses _ts by default or a custom numeric path, while a custom merge procedure can implement application semantics. If the procedure is missing or fails, the conflict must be handled from the conflicts feed.

Microsoft Learn: [Manage Conflicts Between Regions - Azure Cosmos DB](#) · [Conflict Resolution Types and Resolution Policies - Azure Cosmos DB](#)

58 Drag and drop · Secure, optimize, and deploy database solutions**ANSWER**

A new account must use scheduled backups, and a support-request restore is acceptable.	Periodic backup mode
Self-service point-in-time restore is required for accidental deletes for up to six days, with no backup storage charge for the tier.	Continuous backup, 7-day tier
Self-service point-in-time restore is required for corruption that might be discovered after 21 days.	Continuous backup, 30-day tier
An accidentally deleted container must be restored back into account1 instead of into a separate account.	Point-in-time restore in the same account

Periodic backup mode is the default scheduled backup model and restores are requested through support. Continuous backup supports self-service point-in-time restore with 7-day, 30-day, or 35-day retention tiers, and the 7-day tier has no backup storage charge. Continuous backup can restore deleted databases or containers into the same account or can restore selected resources into a new account.

Microsoft Learn: [Online Backup and On-Demand Data Restore - Azure Cosmos DB](#) · [Continuous Backup with Point-in-Time Restore - Azure Cosmos DB](#) · [Periodic Backup and Restore Introduction - Azure Cosmos DB](#)

59 Build list · Secure, optimize, and deploy database solutions

CORRECT ORDER: F → D → E → B → G

1. Open the Point In Time Restore page for account1.
2. Select the Restore to same account tab.
3. Enter a database search query and a start and end time that include the deletion, and then refresh the event feed.
4. Select Restore for the deleted container event after you verify the restore time.
5. Track the restore until the container status changes from Creating to Online.

For same-account restore, you open Point In Time Restore and use the Restore to same account tab. The portal workflow filters deletion events by database and time window, lets you verify the restore time, and starts the restore. The restored container is shown as Creating until the operation completes and the status becomes Online.

WHY THE OTHER CHOICES ARE WRONG

- **A.** The account uses continuous backup, and the Azure portal provides a self-service same-account restore path.
- **C.** A same-account restore restores the deleted container into the existing account, so a new target account is not created first.

Microsoft Learn: [Restore Deleted Container or Database to Same Account - Azure Cosmos DB · Continuous Backup with Point-in-Time Restore - Azure Cosmos DB](#)

60 Multiple response · Secure, optimize, and deploy database solutions

ANSWERS: B, D

- B** Enable the Per-partition automatic failover feature on account1 after all application instances are upgraded.
- D** Upgrade every App1 instance to Microsoft.Azure.Cosmos .NET SDK v3.62.0 or later.

Per-partition automatic failover requires a single-write-region Azure Cosmos DB for NoSQL account with at least one read region and a supported consistency level. The .NET SDK must be version 3.62.0 or later, and all application instances should be upgraded before enabling the feature. Session consistency and provisioned throughput satisfy the documented prerequisites.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Per-partition automatic failover requires provisioned throughput, either manual or autoscale, and does not support serverless accounts.
- **C.** The account must have at least one other read region configured.
- **E.** Bounded staleness is not currently supported for per-partition automatic failover.
- **F.** Per-partition automatic failover is for single-write-region accounts that need fast recovery without multi-write conflict resolution.

Microsoft Learn: [Per-partition automatic failover - Azure Cosmos DB · Configure per-partition automatic failover - Azure Cosmos DB](#)

61 Multiple choice · Secure, optimize, and deploy database solutions

ANSWER: D

- D** Perform a forced failover by taking East US offline so the next priority region is promoted.

For a write-region outage on a single-write account, forced failover, also called taking a region offline, is the recommended path when you need to quickly restore write availability. Service-managed failover is automatic but can take one hour or more to initiate. Change write region requires healthy regions and should not be used during outage scenarios.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Service-managed failover can take significant time, potentially one hour or more, so it is not the fastest recovery path.
- **B.** Updating the account to a multi-write configuration is a control-plane change to avoid during outage scenarios.
- **C.** Backup restore recovers data into a restored account, but it is not the failover action that quickly restores write availability for account1.
- **E.** Change write region is for healthy regions and should not be used during a write-region outage.

Microsoft Learn: [Reliability in Azure Cosmos DB · Manage by Using the Azure Portal - Azure Cosmos DB](#)

62 Yes or No · Secure, optimize, and deploy database solutions

ANSWER: YES

Solution: You add a composite index with /category ascending followed by /updatedAt descending, and then wait for the index transformation to complete.

Queries that use ORDER BY on two or more properties require a composite index. The composite index paths must match the ORDER BY property sequence, and the index order must match the ORDER BY order or the opposite order on all paths. Adding the matching composite index directly reduces the RU cost for this query shape after the index transformation completes.

Microsoft Learn: [Indexing policies - Azure Cosmos DB · SQL Query Metrics - Azure Cosmos DB](#)

63 Yes or No · Secure, optimize, and deploy database solutions

ANSWER: NO

Solution: You double the provisioned throughput on container1.

Increasing provisioned throughput can improve throughput or latency when requests are rate limited, but it does not make this query consume fewer RUs. The scenario requires reducing the request unit charge of a multi-property ORDER BY query. A matching composite index addresses that query shape, not a throughput increase.

Microsoft Learn: [Indexing policies - Azure Cosmos DB · SQL Query Metrics - Azure Cosmos DB](#)

64 Yes or No · Secure, optimize, and deploy database solutions**ANSWER: NO**

Solution: You exclude /updatedAt/? from the indexing policy because the property is only used for sorting.

A multi-property ORDER BY query needs a composite index whose paths match the ORDER BY sequence and direction. Excluding /updatedAt/? removes an indexed path that the query needs instead of adding the required composite index. Removing indexed paths can make the query engine stop using the index and scan instead.

Microsoft Learn: [Indexing policies - Azure Cosmos DB](#)

65 Drop-down selection · Secure, optimize, and deploy database solutions**ANSWER**

Included root path	/*
Excluded unused path	/rawText/*
First composite path	/tenantId
Score order	descending

The root path can be included and then unused paths such as rawText can be selectively excluded to reduce write RU consumption. For a filtered ORDER BY query, the properties in the filter should be included first in the ORDER BY clause and in the same sequence in the composite index. The score path must be descending to match the query order.

Microsoft Learn: [Indexing policies - Azure Cosmos DB](#)

66 Yes or No · Secure, optimize, and deploy database solutions**ANSWERS**

If /price/? appears under Utilized Single Indexes, the query used that indexed path, and removing it can affect the query.	Yes
A path under Potential Composite Indexes is conclusive proof that the query will use that index after it is added.	No
PopulateIndexMetrics should be enabled only while troubleshooting slow queries because collecting index metrics adds overhead.	Yes

Index metrics show utilized single and composite indexes plus potential single and composite indexes. Utilized indexes are evidence that a query used those paths. Potential indexes should be tested after you add them because they are recommendations and might not improve performance for the data distribution.

Microsoft Learn: [Indexing Metrics - Azure Cosmos DB](#) · [SQL Query Metrics - Azure Cosmos DB](#)

67 Drop-down selection · Secure, optimize, and deploy database solutions**ANSWER**

Requests table	CDBDataPlaneRequests
Query text table	CDBQueryRuntimeStatistics
Sort column	RequestCharge

For resource-specific diagnostic logs, DataPlaneRequests are written to CDBDataPlaneRequests and query runtime details are written to CDBQueryRuntimeStatistics. Joining those tables on ActivityId correlates the request charge with QueryText. Ordering by RequestCharge descending returns the most expensive queries in the time window.

Microsoft Learn: [Troubleshoot Issues With Advanced Diagnostics Queries - Azure Cosmos DB](#) · [Monitoring Data Reference - Azure Cosmos DB](#)

68 Build list · Secure, optimize, and deploy database solutions**CORRECT ORDER: E → A → F → G → C**

1. Create a new alert rule and select account1 as the scope.
2. Select the Total Request Units signal as a metric condition.
3. Add a StatusCode dimension filter with the custom value 429.
4. Set a static threshold with Total aggregation, Greater than, and a value of 100.
5. Select or create an email action group, enter alert rule details, and create the rule.

Azure Monitor metric alerts can be created for an Azure Cosmos DB account by selecting the account scope, choosing a metric signal, applying dimensions, defining alert logic, and attaching an action group. The documented 429 example uses the Total Request Units signal, filters the StatusCode dimension to 429, and triggers when the Total aggregation is greater than the configured threshold. The action group defines the email notification target.

WHY THE OTHER CHOICES ARE WRONG

- **B.** QueryRuntimeStatistics is useful for query diagnostics, but it is not required for a metric alert on Total Request Units filtered by StatusCode.
- **D.** Manual failover is unrelated to creating an Azure Monitor metric alert for HTTP 429 responses.

Microsoft Learn: [Create Alerts for Using Azure Monitor - Azure Cosmos DB](#) · [Monitoring Data Reference - Azure Cosmos DB](#)

69 Multiple choice · Secure, optimize, and deploy database solutions

ANSWER: B

- B** Store an uppercase description value at write time and filter on that indexed value.

The metrics show a scan: many documents are retrieved and only a few are output, with no index utilization. The documentation shows that a filter using UPPER is not served from the index and recommends storing values in the desired case, then querying the indexed value directly. That change reduces document loading and RU charge for the query.

WHY THE OTHER CHOICES ARE WRONG

- **A.** MaxItemCount changes page sizing, but it does not make a function-based filter use the index.
- **C.** Increasing throughput can reduce rate limiting, but it does not reduce the query's RU charge.
- **D.** PopulateIndexMetrics helps troubleshoot index use and adds overhead; it does not optimize the query itself.

Microsoft Learn: [Get SQL query performance & execution metrics - Azure Cosmos DB](#) · [SQL Query Metrics - Azure Cosmos DB](#)

70 Multiple choice · Secure, optimize, and deploy database solutions

ANSWER: C

- C** The CosmosDiagnostics output from slow SDK responses.

Server-side latency measures time spent on the backend before data is returned and helps rule out backend latency issues. The .NET SDK Diagnostics property records request information, including retries and client elapsed time, and the troubleshooting guidance uses it to investigate client CPU, thread starvation, network transit, and connection issues. Because the backend metric is low but end-to-end latency is high, SDK diagnostics are the next useful data source.

WHY THE OTHER CHOICES ARE WRONG

- **A.** App1 uses direct connectivity, and Server Side Latency Direct has already ruled out high backend latency.
- **B.** The slow operations are point reads, and query runtime logs focus on query execution rather than client-side point-read latency.
- **D.** Index transformation progress is relevant to indexing policy changes, not client elapsed time for point reads.

Microsoft Learn: [How to Monitor the Server-Side Latency for Operations - Azure Cosmos DB](#) · [Troubleshoot Slow Requests in .NET SDK - Azure Cosmos DB](#)

71 Multiple response · Secure, optimize, and deploy database solutions

ANSWERS: B, E

- B** A 100 percent normalized RU value means at least one partition key range used all of its allocated RU/s.
- E** No immediate throughput increase is required solely because the 429 rate is 3 percent and latency is acceptable.

For production workloads, the documentation states that 1 to 5 percent 429 responses can be healthy when end-to-end latency is acceptable. Normalized RU Consumption is defined as the maximum RU/s utilization across partition key ranges during the interval, so a value of 100 percent can reflect a single busy range. If the rate of 429 responses is higher or latency misses the objective, you should investigate hot partitions and throughput needs.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Increasing RU/s can be a temporary mitigation, but a hot logical partition often requires a better partition key or data model for a permanent fix.
- **C.** Normalized RU consumption is the maximum utilization across partition key ranges, not proof that every physical partition is saturated.
- **D.** Azure Cosmos DB SDKs automatically retry 429 responses by default, so many 429 responses can be hidden from the application.

Microsoft Learn: [Monitor Normalized Request Units - Azure Cosmos DB](#) · [Troubleshoot Request Rate Too Large Exceptions - Azure Cosmos DB](#)

72 Multiple choice · Secure, optimize, and deploy database solutions

ANSWER: A

- A** Use a point read with the item id and the tenantId partition key value.

Azure Cosmos DB reads are typically fastest and most efficient when they are point reads by item id and partition key. A 1-KB item read costs 1 RU in the documented key-value access pattern. Because App1 already has both values and cannot return stale data, replacing the query with a point read minimizes RU charge without relying on cached results.

WHY THE OTHER CHOICES ARE WRONG

- **B.** A composite index optimizes some query shapes, but a point read is more efficient when id and partition key are already known.
- **C.** MaxConcurrency can help parallel cross-partition queries, but this workload reads one known item.
- **D.** The integrated cache can reduce repeated read RU charges, but cached reads can be stale within the configured staleness window.

Microsoft Learn: [SQL Query Metrics - Azure Cosmos DB](#) · [Request Unit Charges as a Key Value Store - Azure Cosmos DB](#) · [Integrated Cache - Azure Cosmos DB](#)

73 Drop-down selection · Secure, optimize, and deploy database solutions**ANSWER**

Top-level resource that corresponds to one multitenant application and organizes accounts across subscriptions or resource groups.	Fleet
Logical grouping where database accounts can optionally share RU/s.	Fleetspace
Shared RU/s capacity used when resources exceed their dedicated RU/s.	Fleet pool
Azure Cosmos DB database account registered within a fleetspace.	Fleetspace account

A fleet maps to one multitenant application, and each account in the fleet must belong to a fleetspace. Pooling is an optional fleetspace-level configuration that lets registered fleetspace accounts draw from shared RU/s after dedicated RU/s are consumed.

Microsoft Learn: [Fleets Overview - Azure Cosmos DB](#) · [Create a Fleet \(Preview\) - Azure Cosmos DB](#) · [Fleet Pools - Azure Cosmos DB](#)

74 Yes or No · Secure, optimize, and deploy database solutions**ANSWERS**

A resource consumes its dedicated RU/s before it draws from the fleetspace pool.	Yes
Resources in accounts from different subscriptions and resource groups can share pool RU/s when the accounts are in the same fleetspace.	Yes
A single physical partition can consume unlimited extra RU/s from the pool when the pool maximum is high enough.	No

Fleet pools add shared autoscale RU/s at the fleetspace level while preserving each resource's dedicated RU/s. The pool does not remove physical partition limits, so per-partition consumption is still capped by documented defaults.

Microsoft Learn: [Fleet Pools - Azure Cosmos DB](#) · [Fleets Overview - Azure Cosmos DB](#)

75 Multiple response · Secure, optimize, and deploy database solutions

ANSWERS: E, F

- E** Use the same underlying single-write or multi-write service tier configuration.
- F** Use the same regional configuration for all accounts in the fleetspace.

Throughput pooling requires the database accounts in a fleetspace to share the same regional configuration. The docs also state that accounts must have the same underlying configuration of multi-region writes or single-region writes to participate in the same pool.

WHY THE OTHER CHOICES ARE WRONG

- **A.** The create fleetspace guidance requires the maximum throughput to be not less than the minimum throughput.
- **B.** Pool RU/s are configured at the fleetspace level, not separately as shared throughput on each tenant container.
- **C.** The fleet documentation supports accounts across different subscriptions and resource groups within a fleetspace.
- **D.** Accounts must have the same underlying configuration of single-region writes or multi-region writes to participate in the same pool.

Microsoft Learn: [Fleet Pools - Azure Cosmos DB](#) · [Create a Fleet \(Preview\) - Azure Cosmos DB](#)

76 Multiple choice · Secure, optimize, and deploy database solutions

ANSWER: D

- D** Fleet Analytics exported as Delta Lake tables to Fabric OneLake or ADLS Gen2

Fleet Analytics consolidates usage and cost data across all Azure Cosmos DB accounts and subscriptions within a fleet. The data is delivered as Apache Delta Lake tables in Fabric OneLake or ADLS Gen2 at an hourly grain for SQL endpoint, KQL, Spark, Power BI, and related analytics scenarios.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Azure Monitor metrics are account-level and use one-minute aggregation, not fleet-wide hourly analytics data.
- **B.** Azure Monitor logs provide account-level request details for troubleshooting rather than fleet-wide aggregated trend tables.
- **C.** FleetspaceAutoscaledThroughput monitors pool scale at the fleet level but is not the fleet analytics dataset for long-term usage and cost analysis.

Microsoft Learn: [Fleet Analytics \(Preview\) - Azure Cosmos DB](#) · [Enable Fleet Analytics \(Preview\) - Azure Cosmos DB](#)

77 Multiple response · Secure, optimize, and deploy database solutions

ANSWERS: C, F

- C** Add the Cosmos DB Fleet Analytics service principal as a Contributor in the Fabric workspace.
- F** Allow up to one hour for data to start populating the storage location.

For a Fabric destination, the enablement steps require granting the shared Cosmos DB Fleet Analytics service principal the Contributor role in the Fabric workspace. The FAQ states that data can take up to one hour to start populating after access is granted, and missing this permission results in no data being written.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Key rotation is not a documented prerequisite for data to be written to Fleet Analytics destinations.
- **B.** Fleet Analytics writes through the documented shared service principal, not through data plane RBAC on each container.
- **D.** Diagnostic settings to Log Analytics are separate from configuring Fleet Analytics destinations.
- **E.** The Fleet Analytics enablement article does not require analytical store on each container.

Microsoft Learn: [Enable Fleet Analytics \(Preview\) - Azure Cosmos DB · Frequently Asked Questions - Fleets \(Preview\) - Azure Cosmos DB for NoSQL](#)

78 Yes or No · Implement AI and analytics capabilities in database solutions

ANSWER: YES

Solution: You use `SELECT TOP 10 * FROM c ORDER BY RANK RRF(VectorDistance(c.embedding, @embedding), FullTextScore(c.description, @term1, @term2))`.

This solution meets the goal because RRF combines scores from VectorDistance and FullTextScore into a single ranking. Hybrid search in Azure Cosmos DB for NoSQL uses RRF to merge vector similarity with BM25 full-text scoring.

Microsoft Learn: [Use Hybrid Search - Azure Cosmos DB for NoSQL · RRF - Query Language for Cosmos DB \(in Azure and Fabric\)](#)

79 Yes or No · Implement AI and analytics capabilities in database solutions

ANSWER: NO

Solution: You use `SELECT TOP 10 * FROM c WHERE FullTextContains(c.description, @term1) ORDER BY VectorDistance(c.embedding, @embedding)`.

This solution does not meet the goal because FullTextContains only filters documents that contain a term. The final order is based on VectorDistance, so the query does not create a fused rank from BM25 keyword relevance and vector similarity.

Microsoft Learn: [Use Hybrid Search - Azure Cosmos DB for NoSQL · FULLTEXTCONTAINS - Query Language for Cosmos DB \(in Azure and Fabric\)](#)

80 Yes or No · Implement AI and analytics capabilities in database solutions

ANSWER: NO

Solution: You use `SELECT TOP 10 * FROM c ORDER BY RANK FullTextScore(c.description, @term1, @term2)`.

This solution does not meet the goal because `FullTextScore` ranks by BM25 full-text relevance only. Hybrid search requires RRF with both `VectorDistance` and `FullTextScore` so that semantic similarity and keyword relevance both contribute to the final rank.

Microsoft Learn: [Use Hybrid Search - Azure Cosmos DB for NoSQL](#) · [FULLTEXTSCORE - Query Language for Cosmos DB \(in Azure and Fabric\)](#)

81 Drop-down selection · Implement AI and analytics capabilities in database solutions

ANSWER

Vector path	/embedding
Vector data type	float16
Excluded regular index path	/embedding/*
Vector index type	diskANN

The vector embedding policy must describe the vector path, data type, dimensions, and distance function. For large search scopes, `diskANN` is the recommended vector index type, and excluding the vector path from regular indexing reduces insertion RU charge and latency.

Microsoft Learn: [Integrated Vector Store - Azure Cosmos DB](#) · [Tips for optimizing vector indexing & search performance - Azure Cosmos DB](#)

82 Drop-down selection · Implement AI and analytics capabilities in database solutions

ANSWER

Result limit	10
Ranking function	VectorDistance

Vector search uses `VectorDistance` to compare the stored vector with the query vector and can order by that value. The documentation recommends using a TOP N clause so the query does not attempt to return more results than needed.

Microsoft Learn: [Integrated Vector Store - Azure Cosmos DB](#) · [VECTORDISTANCE - Query Language for Cosmos DB \(in Azure and Fabric\)](#)

83 Drag and drop · Implement AI and analytics capabilities in database solutions**ANSWER**

A small, filtered search scope uses 384-dimensional vectors and requires exact nearest-neighbor recall.	flat
A tenant-scoped workload usually searches fewer than 50,000 vectors and can accept a small recall tradeoff for lower RU cost than exact search.	quantizedFlat
A production document search runs over more than 50,000 vectors per search scope and needs the lowest latency and RU cost with high recall.	diskANN

The flat index performs exact brute-force search and is limited to 505 dimensions. quantizedFlat supports up to 4,096 dimensions and is a good fit for smaller or well-scoped searches, while diskANN is generally the lowest-latency and lowest-RU choice for larger search scopes.

Microsoft Learn: [Integrated Vector Store - Azure Cosmos DB](#) · [Tips for optimizing vector indexing & search performance - Azure Cosmos DB](#)

84 Build list · Implement AI and analytics capabilities in database solutions**CORRECT ORDER: D → C → B → F → E**

1. Generate an embedding for the user question.
2. Run a vector or hybrid query against Azure Cosmos DB to retrieve the TOP relevant items.
3. Build a prompt that includes the retrieved items as grounding context.
4. Send the grounded prompt to the LLM completion deployment.
5. Return the generated answer to the user.

A RAG flow retrieves relevant data first, then passes that data to the LLM so the response is grounded. The Azure Cosmos DB RAG guidance describes ingesting content, querying for relevant data, using the retrieved data with an LLM, and returning the generated response.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Azure Cosmos DB stores and retrieves vectorized content, but it does not train the embedding model in this RAG flow.
- **G.** Using only pretrained knowledge skips retrieval and does not ground the response in current database content.

Microsoft Learn: [Retrieval Augmented Generation - Azure Cosmos DB](#) · [Tutorial: Build a RAG Chatbot - Azure Cosmos DB for NoSQL](#)

85 Yes or No · Implement AI and analytics capabilities in database solutions

ANSWERS

Adding vectorIndexShardKey to a DiskANN vector index can create separate DiskANN indexes for each unique value of the specified property.	Yes
To focus a vector search on a shard, include a WHERE filter on the shard key value, such as c.tenantId = @tenantId.	Yes
If vectorIndexShardKey is omitted, Azure Cosmos DB creates one DiskANN index per unique tenant ID value by default.	No

Sharded DiskANN is useful when searches are scoped by a property such as tenant ID or category. The shard key is optional, and queries must filter on the shard key value to search the focused DiskANN shard.

Microsoft Learn: [Sharded DiskANN Focused Vector Search for Better Performance and Lower Cost - Azure Cosmos DB](#) · [Tips for optimizing vector indexing & search performance - Azure Cosmos DB](#)

86 Multiple choice · Implement AI and analytics capabilities in database solutions

ANSWER: B

B A container full-text policy for /body and a full-text index for /body in the indexing policy.

Full-text search should be configured with a container-level full-text policy that declares the text path and language. A full-text index in the indexing policy enables efficient full-text functions and helps avoid higher RU cost and latency.

WHY THE OTHER CHOICES ARE WRONG

- **A.** A regular included path does not define the full-text policy or full-text index used by full-text functions.
- **C.** A vector policy and DiskANN index support vector search, not tokenized full-text search over text.
- **D.** A composite index supports ordered queries over properties, not BM25 full-text scoring.

Microsoft Learn: [Use Full-Text Search - Azure Cosmos DB](#) · [Full Text Indexing and Search - Cosmos DB \(in Azure and Fabric\)](#)

87 Multiple response · Implement AI and analytics capabilities in database solutions

ANSWERS: D, E

D Use a diskANN vector index for the /embedding path.

E Set the vector dataType to float16 when the recall tradeoff is acceptable.

float16 can reduce the vector storage footprint by 50 percent compared with float32, with some possible accuracy impact. diskANN supports up to 4,096 dimensions and is the recommended index for large search scopes, while the overall item must still fit within the 2 MB item limit.

WHY THE OTHER CHOICES ARE WRONG

- **A.** decimal is not one of the supported vector data types.
- **B.** The flat index is limited to 505 dimensions, so it is not valid for 1,536-dimensional embeddings.
- **C.** Azure Cosmos DB for NoSQL still has a maximum item size of 2 MB for the JSON representation.
- **F.** VectorDistance expects one-dimensional vector arrays, and multidimensional arrays do not return a similarity score.

Microsoft Learn: [Integrated Vector Store - Azure Cosmos DB](#) · [Tips for optimizing vector indexing & search performance - Azure Cosmos DB](#) · [Service Quotas and Default Limits - Azure Cosmos DB](#)

88 Multiple choice · Implement AI and analytics capabilities in database solutions

ANSWER: D

D filterPriority

filterPriority controls the relative priority of the WHERE filter versus vector search in DiskANN queries. Higher values can bias candidate selection toward filter matches and reduce RU cost, with a possible recall tradeoff that can be mitigated with candidate list multipliers.

WHY THE OTHER CHOICES ARE WRONG

- **A.** indexingSearchListSize affects DiskANN index build recall and ingestion cost, not query-time filter priority.
- **B.** vectorIndexShardKey partitions a DiskANN index by property value, but it is not a VectorDistance query-time option.
- **C.** quantizerType is an index configuration setting for quantization, not the query-time balance between filters and vector search.

Microsoft Learn: [Tips for optimizing vector indexing & search performance - Azure Cosmos DB](#) · [VECTORDISTANCE - Query Language for Cosmos DB \(in Azure and Fabric\)](#)

89 Multiple choice · Implement AI and analytics capabilities in database solutions

ANSWER: B

B Embed the user question, retrieve relevant documents from Azure Cosmos DB with vector or hybrid search, and pass the retrieved content to the LLM as grounding.

RAG combines retrieval with generation so the model can answer with relevant and current data. Azure Cosmos DB can retrieve relevant vectorized content, and the application then passes the retrieved data to an LLM to generate the grounded response.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Sending the question directly to the LLM skips retrieval and does not ground the response in current database content.
- **C.** A fixed point read does not retrieve the documents most relevant to each user question.
- **D.** Returning matching documents without the LLM does not implement the RAG generation step.
- **E.** Chat history can provide conversational context, but skipping the document collection omits the current grounding data.

Microsoft Learn: [Retrieval Augmented Generation - Azure Cosmos DB](#) · [Tutorial: Build a RAG Chatbot - Azure Cosmos DB for NoSQL](#)

90 Drag and drop · Implement AI and analytics capabilities in database solutions

ANSWER

A raw record of one user message, agent reply, tool result, or system message that App1 might replay for recent conversation history	turn
A compact record of one long support thread that captures decisions, open issues, and next steps	summary
A discrete user requirement extracted from a thread, such as the user prefers invoices grouped by project	fact
A cross-thread profile for a single user that preserves stable preferences and constraints across sessions	user_summary

Agent Memory Toolkit defines turn, summary, fact, and user_summary memory types. Turns preserve raw conversation records, summaries compact one thread, facts capture discrete assertions, and user summaries preserve stable user context across threads.

Microsoft Learn: [Agent Memory Toolkit \(Preview\) - Azure Cosmos DB for NoSQL](#)

91 Drop-down selection · Implement AI and analytics capabilities in database solutions

ANSWER

conversationTurns: Most reads retrieve the latest turns for one conversation thread.	/threadId
tenantMemories: Queries often filter by tenant and then retrieve one thread within that tenant.	["/tenantId", "/threadId"]
telemetryTurns: Writes are high volume, and the app rarely revisits a specific conversation.	/pk populated with a new GUID per item

Azure Cosmos DB partition key choice for memories balances locality with distribution. The agent memory guidance recommends thread keys for efficient conversation retrieval, tenant and thread hierarchical keys for multitenant grouping, and GUID keys for write-heavy items when locality is less important.

Microsoft Learn: [Agent Memory in Azure Cosmos DB for NoSQL - Azure Cosmos DB for NoSQL](#)

92 Multiple choice · Implement AI and analytics capabilities in database solutions

ANSWER: C

- C** One document per turn that stores the related messages for that exchange.

The agent memory guidance recommends one document per turn for chat histories and agent memories. This model keeps each exchange as a natural memory unit, supports recent-turn retrieval and embeddings, and makes it easy to apply TTL to older memories.

WHY THE OTHER CHOICES ARE WRONG

- **A.** A response-per-item model is granular but does not preserve a complete back-and-forth exchange as the memory unit.
- **B.** Embedding every thread under one user document creates unbounded items and does not match the documented memory models.
- **D.** A single thread document can grow large and makes each append an update to a growing item.

Microsoft Learn: [Agent Memory in Azure Cosmos DB for NoSQL - Azure Cosmos DB for NoSQL](#)

93 Multiple response · Implement AI and analytics capabilities in database solutions

ANSWERS: A, B

- A** Generate an embedding for each memory observation and store it in a top-level field configured for vector indexing.
- B** Generate an embedding for the current context and query memories by using VectorDistance scoped to the authorized user or thread.

Semantic memory uses embeddings and vector search so that related memories can be retrieved by meaning rather than by exact words. Azure Cosmos DB queries can use VectorDistance over a vector-indexed embedding field, and the query should remain scoped to the authorized user, tenant, or thread context.

WHY THE OTHER CHOICES ARE WRONG

- **C.** Omitting user or thread filters weakens isolation and makes retrieval less focused.
- **D.** A thread-per-item model is not the recommended semantic memory unit and becomes expensive as the thread grows.
- **E.** Raw turns alone do not provide the derived semantic observations that the agent needs to recall across sessions.
- **F.** Full-text search is useful for lexical matching, but semantic memory requires vector similarity by meaning.

Microsoft Learn: [Agent Memory in Azure Cosmos DB for NoSQL - Azure Cosmos DB for NoSQL · Implement Semantic Memory with Vector Storage - Training](#)

94 Multiple choice · Implement AI and analytics capabilities in database solutions

ANSWER: B

- B** Use Azure Durable Functions with the Azure Cosmos DB change feed to process new turns in the background.

Agent Memory Toolkit can process memory in two ways. For robust production pipelines, the documentation recommends Azure Durable Functions with the Azure Cosmos DB change feed so that new turns trigger background summaries, fact extraction, and user summaries.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Manual orchestration in the request path is positioned for lightweight demos or dev and test workflows, not robust production pipelines.
- **C.** Fabric mirroring is for analytics and does not replace the toolkit pipeline for transforming new turns into memories.
- **D.** Local in-memory storage does not provide the durable shared state or automatic processing pipeline required here.

Microsoft Learn: [Agent Memory Toolkit \(Preview\) - Azure Cosmos DB for NoSQL](#)

95 Multiple choice · Implement AI and analytics capabilities in database solutions

ANSWER: C

- C** Use a hybrid selection that reserves most slots for similarity and importance ranking and some slots for recent memories.

The context window optimization guidance describes ranking by similarity, importance, and recency, plus a hybrid strategy that reserves part of the memory budget for recent information. This prevents older high-importance memories from hiding a recent change while still retaining durable context.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Random sampling does not optimize for relevance, importance, or recency.
- **B.** A recency-only strategy can omit critical long-term facts.
- **D.** An importance-only strategy can omit recent updates that supersede older context.

Microsoft Learn: [Configure Context Window Optimization - Training](#)

96 Drop-down selection · Implement AI and analytics capabilities in database solutions

ANSWER

An existing Azure Cosmos DB for NoSQL account must remain the production system of record, while analysts need near real-time Fabric analytics with no ETL and no RU impact on transactional workloads.	Azure Cosmos DB for NoSQL with mirroring into Microsoft Fabric
A new Fabric-native app needs a simplified NoSQL operational store, Microsoft Entra authentication, Fabric capacity billing, and automatic OneLake availability.	Cosmos DB in Microsoft Fabric
A team is starting a new analytics project over existing Azure Cosmos DB operational data and asks whether to use Azure Synapse Link.	Azure Cosmos DB for NoSQL with mirroring into Microsoft Fabric

Fabric mirroring is the current no-ETL analytics path for existing Azure Cosmos DB for NoSQL data, and it keeps transactional and analytical workloads isolated. Cosmos DB in Microsoft Fabric is the Fabric-integrated NoSQL operational database with simplified management and automatic OneLake integration.

Microsoft Learn: [Analytics and BI - Azure Cosmos DB](#) · [Cosmos DB Database - Microsoft Fabric](#) · [Frequently Asked Questions for Cosmos DB database - Microsoft Fabric](#)

97 Build list · Implement AI and analytics capabilities in database solutions

CORRECT ORDER: F → B → D → E → G

1. In the Fabric workspace, create a Mirrored Azure Cosmos DB item.
2. Create a new Azure Cosmos DB for NoSQL connection by using the account endpoint and Organizational account authentication.
3. Select db1 and, optionally, the specific containers to mirror.
4. Select Mirror database.
5. Select Monitor replication and wait until the status changes to Running.

The account prerequisites are already in place: continuous backup, public network access, and, for Microsoft Entra ID authentication, the readMetadata and readAnalytics data actions. In Fabric, you create the Mirrored Azure Cosmos DB item, create the Azure Cosmos DB for NoSQL connection, select the database and any specific containers, select Mirror database, and then use Monitor replication until the status changes to Running.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Network ACL Bypass is needed only when the account uses virtual networks or private endpoints. account1 allows public network access from all networks.
- **C.** Mirroring does not use analytical store as its change capture source, and analytical store is not a prerequisite.

Microsoft Learn: [Tutorial: Configure Microsoft Fabric Mirrored Databases From Azure Cosmos DB - Microsoft Fabric](#) · [Microsoft Fabric Mirrored Databases From Azure Cosmos DB - Microsoft Fabric](#) · [Limits and Quotas in Microsoft Fabric Mirrored Databases From Azure Cosmos DB - Microsoft Fabric](#)

98 Drop-down selection · Implement AI and analytics capabilities in database solutions

ANSWER

JSON expansion operator	OUTER APPLY
order_id data type	varchar(100)

Mirrored nested data is represented as JSON strings in SQL analytics endpoint tables. The Fabric guidance uses OUTER APPLY with OPENJSON to include rows that might not have the array, and it maps scalar order_id values to bounded varchar types.

Microsoft Learn: [How to: Query nested data in Microsoft Fabric mirrored databases from Azure Cosmos DB - Microsoft Fabric](#)

99 Yes or No · Implement AI and analytics capabilities in database solutions

ANSWERS

After the initial synchronization, individual insert, update, and delete operations are replicated into OneLake in near real-time.	Yes
The replication process consumes request units from the source Azure Cosmos DB account.	No
The SQL analytics endpoint for the mirrored database is read-only and cannot manipulate the source data.	Yes

Mirroring replicates Azure Cosmos DB data into OneLake for analytics with workload isolation from the transactional account. SQL, Power BI, and Spark query compute are billed through Fabric capacity, while replication itself does not use source request units.

Microsoft Learn: [Microsoft Fabric Mirrored Databases From Azure Cosmos DB - Microsoft Fabric](#) · [Frequently asked questions for Microsoft Fabric mirrored databases from Azure Cosmos DB - Microsoft Fabric](#)

100 Multiple choice · Implement AI and analytics capabilities in database solutions

ANSWER: A

A Stop replication, update the Fabric connection credentials with the new key, and restart replication.

The Fabric mirroring documentation states that if account keys are rotated, the connection credentials must be updated. If the credentials are not updated, mirroring fails, and the documented resolution is to stop replication, update the credentials, and restart replication.

WHY THE OTHER CHOICES ARE WRONG

- **B.** Continuous backup is required and cannot be used this way to fix a rotated key.
- **C.** The documented failure cause is invalid credentials after key rotation, not source throughput.
- **D.** Read-only account keys are not supported for Fabric mirroring connections.

Microsoft Learn: [Frequently asked questions for Microsoft Fabric mirrored databases from Azure Cosmos DB - Microsoft Fabric](#) · [Limits and Quotas in Microsoft Fabric Mirrored Databases From Azure Cosmos DB - Microsoft Fabric](#)

101 Multiple response · Implement AI and analytics capabilities in database solutions

ANSWERS: C, D

- C** Set `spark.cosmos.accountEndpoint`, `spark.cosmos.accountKey`, `spark.cosmos.database`, and `spark.cosmos.container` in the write configuration.
- D** Write the `DataFrame` by using `format("cosmos.oltp")` and `options(**writeConfig).save()`.

The Fabric notebook guidance configures `writeConfig` with the Azure Cosmos DB endpoint, account key, database, and container. The `DataFrame` write uses the Cosmos DB Spark connector format `cosmos.oltp` with those options, and write operations consume request units on the Azure Cosmos DB account.

WHY THE OTHER CHOICES ARE WRONG

- **A.** The SQL analytics endpoint is read-only and is not used to write back to Azure Cosmos DB.
- **B.** Schema inference is a read option and does not identify the write target or perform the write.
- **E.** A write operation requires credentials that can write to the target container.
- **F.** The Delta format is used to query mirrored OneLake data, not to write back through the Cosmos DB Spark connector.

Microsoft Learn: [How to: Access mirrored Azure Cosmos DB data in Lakehouse and notebooks from Microsoft Fabric - Microsoft Fabric · Tutorial - Connect Using Spark - Azure Cosmos DB](#)

102 Multiple choice · Implement AI and analytics capabilities in database solutions

ANSWER: A

- A** Create a Lakehouse, add a Microsoft OneLake shortcut to the mirrored database tables, and query the shortcut tables with Spark SQL.

The Fabric guidance for notebooks creates a Lakehouse and then creates a Microsoft OneLake shortcut to the mirrored Azure Cosmos DB database. Spark can then query the Lakehouse tables that point to the mirrored OneLake replica without connecting directly to the Azure Cosmos DB account.

WHY THE OTHER CHOICES ARE WRONG

- **B.** The SQL analytics endpoint is read-only and should not be used to insert mirrored rows into a Lakehouse.
- **C.** The Source database view is a read-only view of live Azure Cosmos DB data and can consume request units; it is not a Lakehouse Spark shortcut.
- **D.** The Spark connector can read a Cosmos DB endpoint directly, but the requirement is to query the OneLake mirrored replica.

Microsoft Learn: [How to: Access mirrored Azure Cosmos DB data in Lakehouse and notebooks from Microsoft Fabric - Microsoft Fabric · Frequently asked questions for Microsoft Fabric mirrored databases from Azure Cosmos DB - Microsoft Fabric](#)

103 Drop-down selection · Design and develop database solutions**ANSWER**

bookings	hierarchical: /travelerId, /tripId
seatHolds	synthetic: /flightDateHoldSuffix
travelerProfiles	/travelerId

Effective partition keys have high cardinality, distribute RU consumption evenly, and align with common queries. Hierarchical partition keys can route prefix queries efficiently and help a large first-level value scale beyond a single logical partition limit. A synthetic suffix is appropriate for seat holds because the application can calculate the suffix and spread writes for a hot flightDate.

Microsoft Learn: [Partitioning and horizontal scaling - Azure Cosmos DB](#) · [Hierarchical Partition Keys - Azure Cosmos DB](#) · [Create a Synthetic Partition Key - Azure Cosmos DB](#)

104 Multiple choice · Design and develop database solutions**ANSWER: D**

D Session

Session consistency guarantees read-your-writes within a client session by using session tokens. This matches the requirement that a traveler sees that traveler's latest itinerary writes without requiring strong consistency across all regions. Session consistency also offers latency and throughput comparable to eventual consistency.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Strong consistency provides the latest committed write globally, but the case does not require global linearizability and strong consistency increases write latency across regions.
- **B.** Bounded staleness permits a configured lag and is primarily useful for near-strong guarantees across regions, not per-user read-your-writes.
- **C.** Eventual consistency does not guarantee read-your-writes within the traveler's session.

Microsoft Learn: [Consistency level choices - Azure Cosmos DB](#)

105 Drag and drop · Design and develop database solutions

ANSWER

Detect booking creates, updates, and cancellation markers	Change feed processor on bookings in latest version mode
Coordinate processing across multiple worker instances	Lease container named leases
Persist the derived read model	Upsert documents in travelerItineraryViews

The change feed is a persistent record of changes that can be processed asynchronously and incrementally. The change feed processor provides automatic checkpointing and at-least-once delivery by using a lease container. Latest version mode is sufficient because the case represents cancellations as updates and the view can be eventually consistent.

Microsoft Learn: [Work with the Change Feed - Azure Cosmos DB](#) · [Change Feed Design Patterns - Azure Cosmos DB](#) · [Change Feed Processor - Azure Cosmos DB](#)

106 Multiple response · Secure, optimize, and deploy database solutions

ANSWERS: D, E

- D** Create tripQuotes with a Last Write Wins conflict resolution path of /revision.
- E** Enable multi-region writes on account1.

Multi-region writes let application instances write to a local Azure Cosmos DB region while the service replicates behind the scenes. In accounts with multiple write regions, conflicting updates can occur and are resolved by a container conflict resolution policy. Last Write Wins for Azure Cosmos DB for NoSQL can use a custom numeric conflict resolution path, such as /revision, and the container must be created with that policy.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Per-partition automatic failover is for single-write-region accounts and does not support serverless accounts.
- **B.** Conflict resolution policy is specified at container creation time and cannot be modified after container creation.
- **C.** Strong consistency cannot be configured for multi-region write accounts, and the case does not require strong consistency.
- **F.** Bounded staleness in a multi-write account creates a dependency on replication lag and does not implement the revision-based conflict rule.

Microsoft Learn: [Multi-region writes - Azure Cosmos DB](#) · [Configure Multi-Region Writes - Azure Cosmos DB](#) · [Conflict Resolution Types and Resolution Policies - Azure Cosmos DB](#)

107 Yes or No · Implement AI and analytics capabilities in database solutions**ANSWERS**

A hierarchical partition key of /tenantId and /conversationId supports tenant-scoped queries and efficient conversation-scoped reads.	Yes
Storing one document per turn is a supported model for recent history and semantic retrieval.	Yes
Setting ttl to 2592000 on raw turn items is sufficient when DefaultTimeToLive is null on the container.	No

Agent memory designs should balance locality for recent-turn queries with distribution and governance. The documented one-document-per-turn model supports recent history, semantic retrieval, and item-level expiration. For TTL to delete raw turns automatically, TTL must be enabled on the container, and item-level ttl can then override the default.

Microsoft Learn: [Agent Memory in Azure Cosmos DB for NoSQL](#) - [Azure Cosmos DB for NoSQL](#) · [Hierarchical Partition Keys](#) - [Azure Cosmos DB](#) · [Expire Data With Time to Live](#) - [Azure Cosmos DB](#)

108 Drag and drop · Secure, optimize, and deploy database solutions**ANSWER**

Prevent new applications from connecting with account keys	Disable key-based authentication and use Microsoft Entra ID
Allow the assistant API to query guidance documents without writing items	Cosmos DB Built-in Data Reader role at the guidance container scope
Allow clinicians to view original PHI values	Custom data plane role with items/unmask
Show redacted PHI to support staff without changing stored values	Dynamic Data Masking policy on PHI paths

Secure Azure Cosmos DB for NoSQL workloads can disable key-based authentication and require Microsoft Entra ID. Native data plane RBAC grants query and item permissions, and the built-in Data Reader role includes read, executeQuery, and readChangeFeed actions. Dynamic Data Masking redacts sensitive fields at query time, while privileged users need the items/unmask data action to see original values.

Microsoft Learn: [Connect using role-based access control and Microsoft Entra ID](#) - [Azure Cosmos DB](#) · [Data plane security reference](#) - [Azure Cosmos DB](#) · [Dynamic Data Masking \(DDM\)](#) - [Azure Cosmos DB](#)

109 Drop-down selection · Implement AI and analytics capabilities in database solutions**ANSWER**

Rank fusion function	RRF
Semantic score	VectorDistance
Keyword score	FullTextScore

Hybrid search in Azure Cosmos DB for NoSQL combines vector search and full-text search by using RRF in an ORDER BY RANK clause. VectorDistance supplies the semantic score, and FullTextScore supplies the full-text BM25 score. The weights array [2, 1] makes the first score, the vector score, twice as important as the keyword score.

Microsoft Learn: [Use Hybrid Search - Azure Cosmos DB for NoSQL](#) · [RRF - Query Language for Cosmos DB \(in Azure and Fabric\)](#) · [Use Full-Text Search - Azure Cosmos DB](#)

110 Build list · Implement AI and analytics capabilities in database solutions**CORRECT ORDER: E → A → D → F**

1. Enable continuous backup on the source Azure Cosmos DB account.
2. In Fabric, create a Mirrored Azure Cosmos DB item in the workspace.
3. Create a connection to the Azure Cosmos DB for NoSQL account and select the database and containers.
4. Select Mirror database and monitor replication until the status is Running.

Fabric mirroring for Azure Cosmos DB requires the source account to use continuous backup. You then create the mirrored database item, connect it to the source Azure Cosmos DB for NoSQL database, select the data to mirror, and start mirroring. Replication status should be monitored until it is running, and analysts should query the mirrored data in OneLake or the SQL analytics endpoint rather than the source database.

WHY THE OTHER CHOICES ARE WRONG

- **B.** Querying the source database from Fabric Data Explorer consumes request units and does not satisfy the analytics requirement.
- **C.** Mirroring uses continuous backup as a prerequisite and does not use analytical store as its change data capture source.
- **G.** A custom change feed processor is not required for Fabric mirroring and would add an application-managed data movement pipeline.

Microsoft Learn: [Microsoft Fabric Mirrored Databases From Azure Cosmos DB - Microsoft Fabric](#) · [Tutorial: Configure Microsoft Fabric Mirrored Databases From Azure Cosmos DB - Microsoft Fabric](#) · [Frequently asked questions for Microsoft Fabric mirrored databases from Azure Cosmos DB - Microsoft Fabric](#)

111 Multiple choice · Secure, optimize, and deploy database solutions

ANSWER: C

- C** Total Request Units metric with the StatusCode dimension set to 429

Azure Monitor alerts for Azure Cosmos DB can use metrics and dimensions. The documented 429 alert uses the Total Request Units signal and filters the StatusCode dimension to 429 before applying a threshold. This directly matches the requirement to notify operations when rate-limited requests exceed the threshold.

WHY THE OTHER CHOICES ARE WRONG

- **A.** Query metrics explain query execution performance, not account-level alerting for rate-limited requests.
- **B.** Normalized RU Consumption by partition key range helps investigate hot partitions, but it does not directly alert on the required status code count.
- **D.** Index metrics help troubleshoot slow queries and recommended indexes, not create a 429 count alert.

Microsoft Learn: [Create Alerts for Using Azure Monitor - Azure Cosmos DB](#)

112 Yes or No · Design and develop database solutions

ANSWERS

The call sends only the modified paths instead of replacing the whole session document.	Yes
The filter predicate causes the patch to fail if the current item is no longer open.	Yes
The ttl value on line 04 will be honored when DefaultTimeToLive is null on the container.	No

The Patch API modifies selected document paths and can use a filter predicate to enforce a precondition. The code therefore closes only currently open sessions and avoids sending the full document. To make the ttl value delete the item after 30 days, TTL must be enabled on the container with DefaultTimeToLive set to a non-null value.

Microsoft Learn: [Get Started with Partial Document Update - Azure Cosmos DB](#) · [Partial Document Update - Azure Cosmos DB](#) · [Expire Data With Time to Live - Azure Cosmos DB](#)