Microsoft.AI-200.v2026-11-05.q116

試験コード：	AI-200
試験名称：	Developing AI Cloud Solutions on Azure
認証ベンダー：	Microsoft
無料問題の数：	116
バージョン：	v2026-11-05
ページの閲覧量：	114
問題集の閲覧量：	1338

https://www.jpnshiken.com/shiken/Microsoft.AI-200.v2026-11-05.q116.html

質問: **1**
セマンティック検索用のAzure Database for PostgreSQLテーブルを設計しています。クエリでは、created_at列に基づいて頻繁にフィルタリングが行われます。スキーマは、ベクトル類似性検索と信頼性の高い日付フィルタリングをサポートする必要があります。テーブルがこれらの要件を満たしていることを確認する必要があります。
どの2つの行動を実行すべきですか？それぞれの正解は解決策の一部を示しています。2つ選択してください。注：正解ごとに1ポイントが加算されます。
A. 埋め込みをpgvector列に格納します。
B. 埋め込みデータをvarchar型の列に格納します。
C. created_at には型付きタイムスタンプ列を使用します。
D. created_at を自由形式の文字列として保存します。
正解: **(**正解を表示します**)**
詳細説明 :セマンティック検索スキーマでは、埋め込みにpgvector型のベクトルを使用する必要があります。これにより、PostgreSQLはベクトル距離演算子とベクトルインデックスを適用できます。作成タイムスタンプには実際のタイムスタンプデータ型を使用する必要があります。これにより、比較、範囲述語、順序付け、およびインデックスは、文字列の順序付けではなく日付のセマンティクスを使用します。ベクトルまたは日付を自由形式の文字データとして格納すると、型の検証が犠牲になり、類似性や日付フィルタリングの効率と信頼性が低下します。
学習ガイドの関連性 :AIデータ管理ワークロード :Cosmos DB、PostgreSQL、キャッシング、ベクトルストレージ、ベクトル検索、一貫性、および接続最適化。
Microsoft Learn 公式資料: AI-200 学習ガイド | Azure PostgreSQL を使用したベクトル類似性検索

質問: **2**
RAGソリューションの埋め込み戦略を選択しています。ドキュメントのページ数は2ページから200ページまで様々です。埋め込みモデルのトークン制限内に収まりつつ、意味的な一貫性を維持する必要があります。どうすればよいでしょうか？
A. 各ドキュメント全体を単一のベクトルとして埋め込む
B. ドキュメントを埋め込む前に、より小さな重複するセグメントに分割する
C. 埋め込みには文書タイトルのみを使用する
D. 文書を画像に変換し、画像埋め込みを使用する
正解: **B (**コメントを発表する**)**
埋め込みモデルにはトークン数の制限があり、長い文書全体を1つのベクトルとして埋め込むと意味が薄れてしまいます。文書をより小さく、わずかに重なり合うセグメントに分割することで、チャンク境界での文脈が維持され、各チャンクがモデルの制限内に収まりながら、検索関連性が向上します。

質問: **3**
あなたはバックエンドワーカーサービスのメッセージ処理コードをレビューしています。
サービスバスプロセッサを初期化して起動する以下のPythonコードを確認してください。

```
from azure.servicebus import ServiceBusClient, ServiceBusReceiveMode
```

```
01 import json
02 from azure.servicebus import ServiceBusClient, ServiceBusReceiveMode
03 from azure.identity import DefaultAzureCredential
04 credential = DefaultAzureCredential()
05 with client.get_queue_receiver(
06    queue_name="inference-requests",
07    receive_mode=ServiceBusReceiveMode.PEEK_LOCK,
08    max_wait_time=30
09 ) as receiver:
10     for msg in receiver:
11         try:
12             payload = json.loads(str(msg))
13             result = run_inference(payload)
14             save_result(result, payload.get("document_id"))
15             receiver.complete_message(msg)
16         except json.JSONDecodeError:
17             receiver.dead_letter_message(
18                 msg,
19                 reason="MalformedPayload",
20                 error_description="Message body is not valid JSON"
21             )
22         except ModelNotFoundError:
23             receiver.dead_letter_message(
24                 msg,
25                 reason="UnsupportedModel",
26                 error_description=f"Model {payload.get('model')} is not available"
27             )
28         except TransientServiceError:
29             receiver.abandon_message(msg)
```



正解:



Explanation:



検証済みの回答 :いいえ、はい、いいえ。
詳細説明 :Service Bus のデフォルトの受信動作は、受信して削除するのではなく、ピークロック方式です。そのため、メッセージを受信しただけではキューから削除されず、正常に処理される必要があります。処理が失敗してメッセージが未処理のままになったり、ロックが失われたりすると、別の配信に使用できるようになります。ただし、無効な JSON はアプリケーションレベルのコンテンツの問題です。Service Bus は、逆シリアル化が失敗したという理由だけでペイロードを JSON として検査し、MaxDeliveryCount まで自動的に再試行することはありません。何が起こるかは、アプリケーションの処理/エラー処理パスによって決まります。
学習ガイドの関連性 :Azureサービス統合 :Service Bus、Event Grid、Azure Functionsトリガー
バインディング、およびイベント駆動型処理。
Microsoft Learn 公式資料: AI-200 学習ガイド | Service Bus メッセージの転送、ロック、および決済 | Service Bus メッセージ、ルーティング、および相関

質問: **4**
ホットスポットに関する質問
あなたは、ある企業向けに検索拡張生成（RAG）ソリューションを開発しています。
AIの応答と埋め込みベクトルはRedisにキャッシュされます。
解決策は以下の要件を満たす必要があります。
AIからの応答は、キャッシュされてから正確に24時間後に期限切れとなる必要があります。
キャッシュされた埋め込みデータは、常に現在のソースデータを反映していなければなりません。
要件を満たすようにRedisを設定する必要があります。
注：正解ごとに1ポイントが加算されます。

Answer Area

Cache expiration and invalidation strategy

Requirement	Action
AI responses must expire exactly 24 hours after they are cached.	Configure allkeys-lru eviction. Reset expiration on every read. Set a Time to Live (TTL) on each key.
Cached embeddings must always reflect the current source data.	Configure volatile-lru eviction. Delete related keys when the document changes. Increase maximum memory allocation.

正解:

Answer Area

Cache expiration and invalidation strategy

Requirement	Action
AI responses must expire exactly 24 hours after they are cached.	Set a Time to Live (TTL) on each key.
Cached embeddings must always reflect the current source data.	Delete related keys when the document changes.

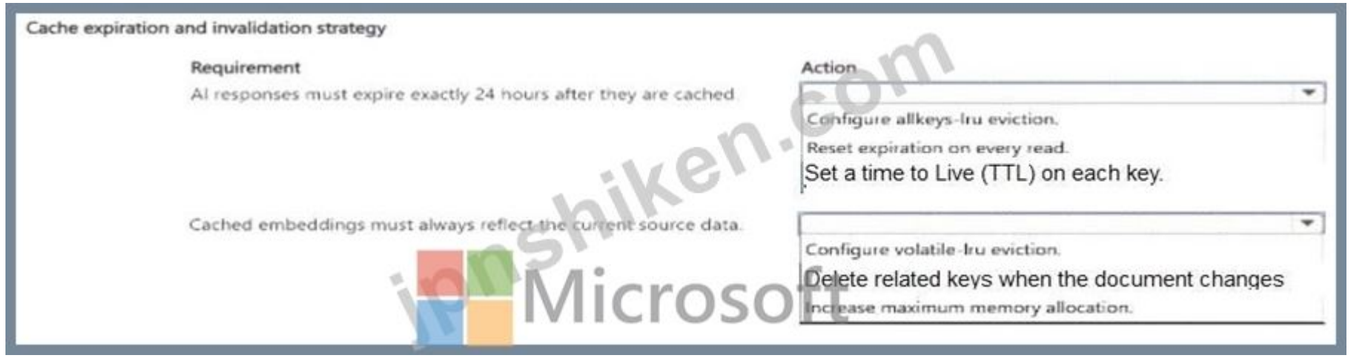
jpnshiken.com
Microsoft

Explanation:
ボックス1：各キーの有効時間（TTL）を設定します。
RedisでAI応答をキャッシュしてから正確に24時間後に期限切れにするには、それらの応答を保持する特定のキーに86,400秒のTime-To-Live（TTL）を設定する必要があります。
ボックス2：文書が変更されたら、関連するキーを削除する
RAGシステムが古い情報を取得するのを防ぐため、ソースデータが変更された場合は、キャッシュされたキーを無効化または更新する必要があります。
正しくない：
キャッシュされた埋め込みや AI レスポンスが最新のソース データを反映していることを保証するために、allkeys-lru (Least Recently Used) を設定することは推奨されません。allkeys-lru は削除ポリシーです。Redis が最大メモリ制限に達した場合にのみデータを削除し、最近アクセスされていないキーを対象とします。基となるソース データが変更されたかどうかは検証しないため、ソース ドキュメントが更新されてもキャッシュされたレスポンスがメモリに残っている場合、ユーザーは危険なほど古い (誤った) AI レスポンスを受け取ることになります。
参照：
https://zenvanriel.com/ai-engineer-blog/ai-caching-strategies/

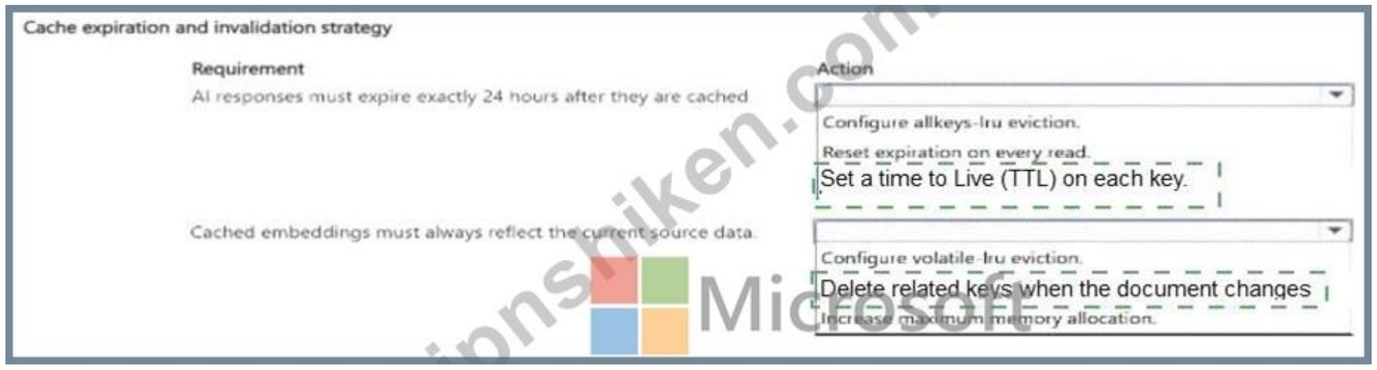
質問: **5**
ACAでアプリの新しいリビジョンをデプロイします。
パフォーマンスを監視しながら、本番環境のトラフィックを段階的に新しいリビジョンに移行する必要があります。さらに、迅速にロールバックできる体制も必要です。
どの2つの行動をとるべきでしょうか？それぞれの正解は、解決策の一部を示しています。
注：正解ごとに1ポイントが加算されます。
A. 単一修正モードを使用します。
B. 改訂を再開します。
C. 複数改訂モードを有効にする。
D. レプリカ数を増やす。
E. トラフィック分割を使用します。
正解: **(正解を表示します)**

Azure Container Apps でトラフィックを段階的に移行し、迅速なロールバックを可能にするには、アプリを複数リビジョンモードに切り替え、アクティブなリビジョン間でパーセンテージベースのトラフィック分割重みを構成します。
参照：
https://learn.microsoft.com/en-us/azure/container-apps/traffic-splitting

質問: **6**
あなたは企業向けに検索拡張生成（RAG）ソリューションを開発しています。AIの応答と埋め込みベクトルはRedisにキャッシュされます。このソリューションは以下の要件を満たす必要があります。
* AIの応答は、キャッシュされてから正確に24時間後に期限切れとなる必要があります。
キャッシュされた埋め込みデータは、常に現在のソースデータを反映していなければなりません。
要件を満たすようにRedisを設定する必要があります。



正解:



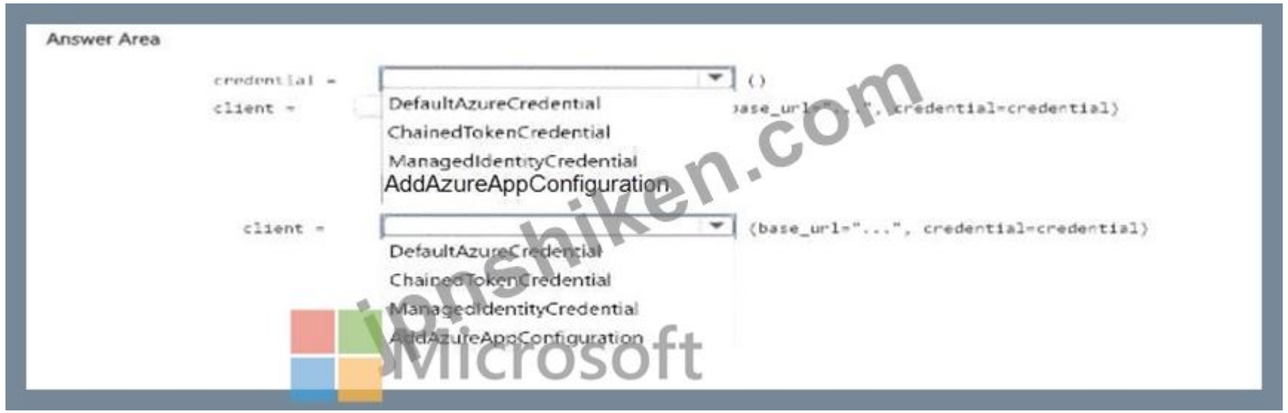
Explanation:



レスポンスの有効期限：各キーに24時間のTTLを設定します。埋め込みの鮮度 :ソースドキュメントが変更されたときに、関連するキャッシュキーを削除します。

詳細説明：固定TTLは、AI応答が無期限に存続したり、読み取りのたびに有効期間が延長されたりするのではなく、定義された期間後に期限切れになる必要があるという要件をエンコードします。鮮度を埋め込むことは、別の整合性要件です。権威ある文書が変更された場合、古い文書から派生したキャッシュされた表現は直ちに無効化されなければなりません。メモリ削除ポリシーは容量管理メカニズムであり、正確な24時間の有効期間やソースデータの整合性を保証するものではありません。
学習ガイドの関連性：AIデータ管理ワークロード：Cosmos DB、PostgreSQL、キャッシング、ベクトルストレージ、ベクトル検索、一貫性、および接続最適化。
Microsoft Learn 公式資料: AI-200 学習ガイド | Azure Managed Redis ドキュメント

質問: **7**
あなたはAzure Functionsアプリを開発しています。
アプリのすべての機能は、以下の要件を満たしています。
成功するまで、または実行試行が10回になるまで実行します。
* 最大15分間、試行間隔を最低20秒以上空けてください。
host.json ファイルを設定する必要があります。
コードセグメントをどのように完成させるべきですか？回答するには、回答欄で適切なオプションを選択してください。
注：正解ごとに1ポイントが加算されます。



正解:



Explanation:



質問: **8**

あなたはAzure Container Apps上で動作する複数のマイクロサービスを開発しています。

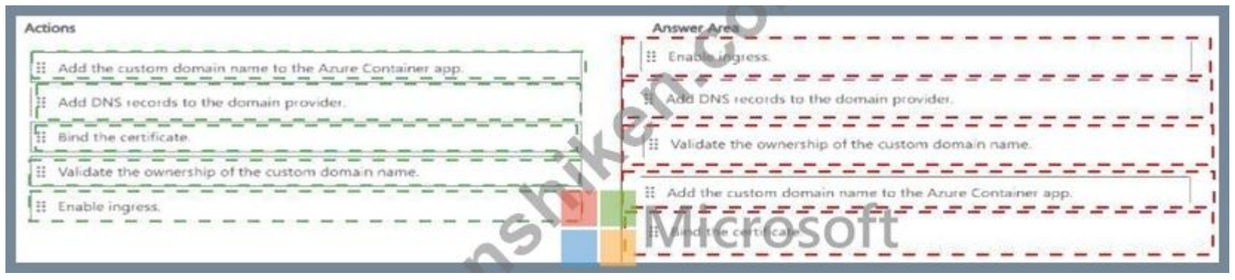
マイクロサービスは、カスタムドメインを使用することでHTTPSアクセスを許可する必要があります。

Azure Container Appsでカスタムドメインを設定する必要があります。

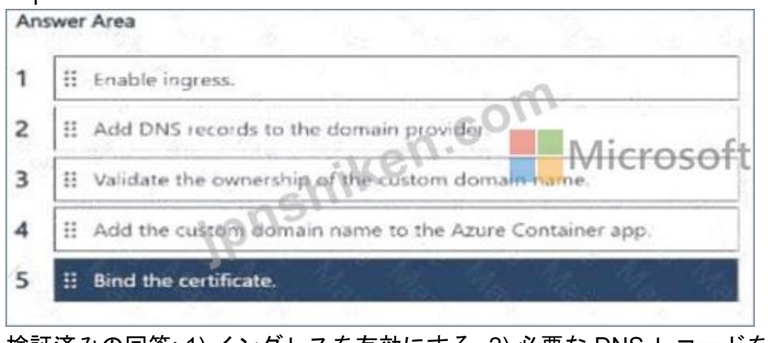
どのような順序で操作を実行すればよいでしょうか？回答するには、操作リストからすべての操作を回答欄に移動し、正しい順序に並べ替えてください。



正解:



Explanation:



検証済みの回答: 1) イングレスを有効にする。2) 必要な DNS レコードを追加する。3) カスタム ドメインの所有権を検証する。4) カスタム ドメインをコンテナ アプリに追加する。5) 証明書をバインドする。

詳細説明 :カスタムHTTPSホスト名には、イングレスエンドポイント、要求元がホスト名を制御していることを示すDNS証明、ホスト名の関連付け自体、およびそのホスト名にバインドされたTLS証明書が必要です。

所有権の検証を成功させるには、DNSレコードが存在している必要があります。所有権が検証されると、ドメインをコンテナアプリに関連付け、証明書をバインドしてHTTPSを提供できるようになります。イングレス検証またはDNS検証を省略すると、ホスト名が正しくルーティングされ、保護されなくなります。
学習ガイドの関連性 :コンテナ化された Azure ワークロード :レジストリの構築、App Service コンテナ、コンテナ アプリの改訂/スケーリング動作、および AKS デプロイの選択肢。
Microsoft Learn 公式資料: AI-200 学習ガイド | コンテナアプリにおけるカスタムドメインと証明書

質問: **9**
Azure Cosmos DB for NoSQL API のネイティブなベクトル検索機能を使用して、検索拡張生成 (RAG) システムを実装しています。
技術記事を格納する Documents」という名前のコンテナがあります。各記事には embedding」という名前のプロパティが含まれています。
システムが、ユーザーの検索クエリと保存されている記事との間で効率的な類似性検索を実行できることを確認する必要があります。
セマンティック検索をサポートするには、データベースリソースを設定する必要があります。
どの構成を使用すべきでしょうか？回答するには、適切な構成を正しい要件に移動してください。各構成は、1回、複数回、またはまったく使用しない場合があります。コンテンツを表示するには、ペイン間の分割バーを移動したり、スクロールしたりする必要がある場合があります。
注：正解ごとに1ポイントが加算されます。

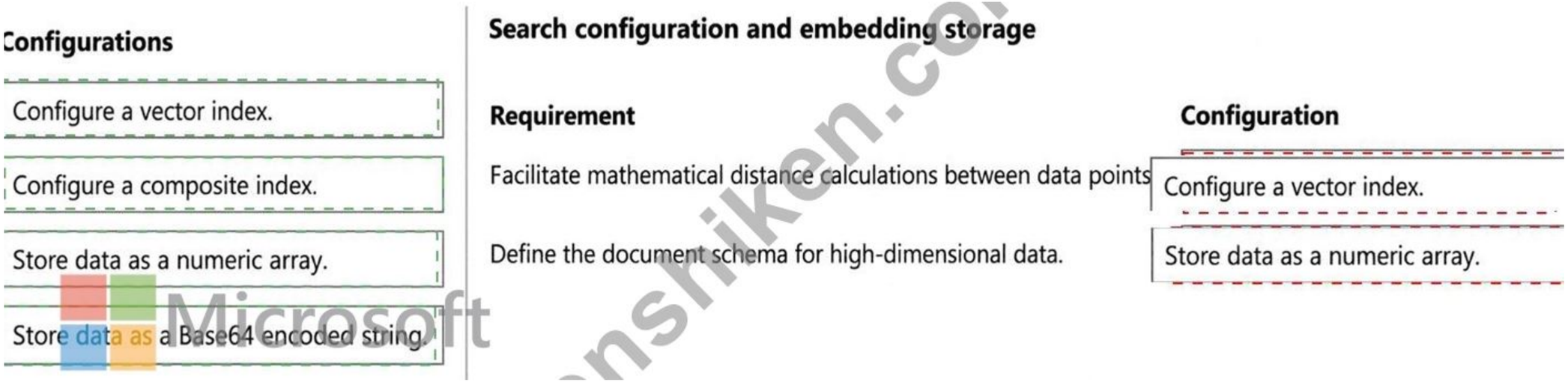
Configurations

- Configure a vector index.
- Configure a composite index.
- Store data as a numeric array.
- Store data as a Base64 encoded string.

Search configuration and embedding storage

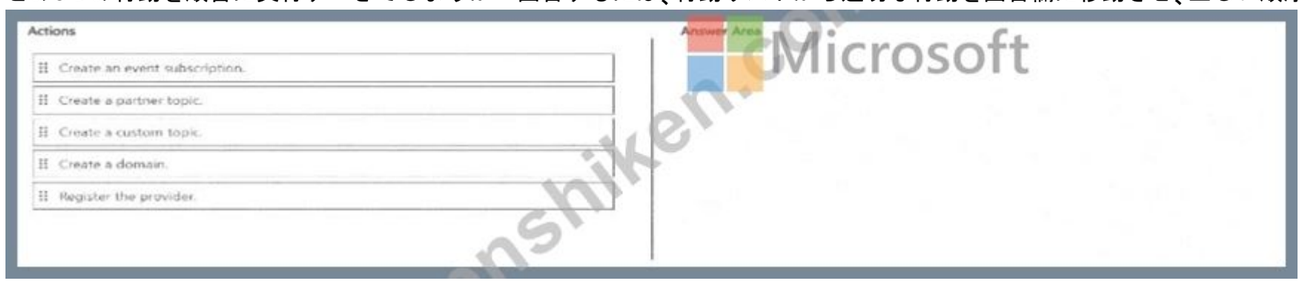
Requirement	Configuration
Facilitate mathematical distance calculations between data points.	
Define the document schema for high-dimensional data.	

正解:

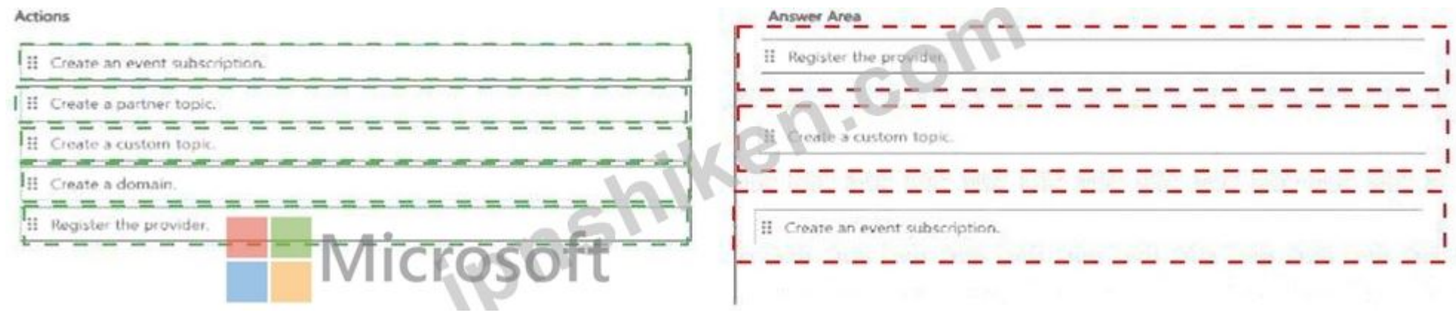


Explanation:
* データポイント間の数学的な距離計算を容易にする :ベクトルインデックスを設定します。
* 高次元データのドキュメントスキーマを定義します。データを数値配列として格納します。
Azure Cosmos DB for NoSQL は、埋め込みを JSON ドキュメントに直接格納し、埋め込みパスにベクトル埋め込みポリシーとベクトルインデックスを定義することで、ネイティブなベクトル検索をサポートします。Microsoft のドキュメントによると、ベクトル検索では、VectorDistance() システム関数を使用して類似度または距離を計算し、クエリベクトルと格納されているベクトルを比較します。ベクトルインデックスを使用すると、インデックスなしのベクトルスキャンと比較して、検索レイテンシの削減、スループットの向上、および RU 消費量の削減により、このプロセスが大幅に改善されます。
埋め込みプロパティ自体は、数値の配列として格納する必要があります。Microsoftの例では、「contentVector」: [2, -1, 4, ...] のような埋め込みプロパティを示し、パス、データ型、次元、距離関数などの属性を持つ関連ベクトルポリシーを定義しています。埋め込みは埋め込みモデルによって生成される高次元の数値ベクトルであるため、このような表現が必要です。
複合インデックスは、複数のスカラープロパティを含むクエリを最適化しますが、ベクトル距離インデックスは提供しません。Cosmos DB はベクトルフィールドに設定された次元数に一致する数値が含まれていることを想定しているため、Base64 エンコードされた文字列をネイティブなベクトル類似度計算に直接使用することはできません。
学習ガイドの参照: Azure Cosmos DB for NoSQL # ベクトル埋め込みポリシー、ベクトルインデックス、VectorDistance()、数値埋め込み配列、ネイティブベクトル検索。

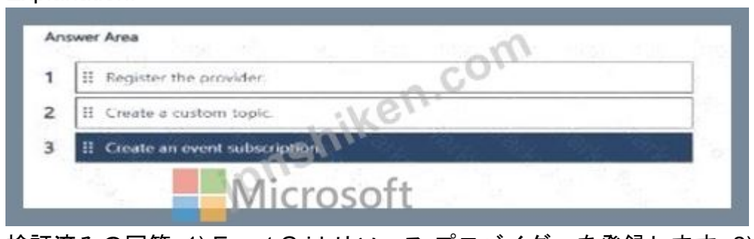
質問: **10**
新しくプロビジョニングされた Azure サブスクリプションがあります。AI 推論イベント用のカスタム Event Grid ワークフローを設計しています。
信頼性の高いイベントを下流のプロセッサにルーティングするには、イベントグリッドのコンポーネントを実装する必要があります。
どの3つの行動を順番に実行すべきでしょうか？回答するには、行動リストから適切な行動を回答欄に移動させ、正しい順序に並べ替えてください。



正解:



Explanation:



検証済みの回答: 1) Event Grid リソース プロバイダーを登録します。2) カスタム トピックを作成します。3) イベント サブスクリプションを作成します。
詳細説明 :Event Gridリソースを作成するには、新しくプロビジョニングされたサブスクリプションにEvent Gridリソースプロバイダが利用可能である必要があります。パブリッシャーは、イベントイングレスリソースとしてカスタムトピックを必要とし、そのトピック上のイベントサブスクリプションによってダウンストリームプロセッサへのルーティングが定義されます。イベントサブスクリプションには、イベントタイプやデータフィールドなどのフィルタを含めることができ、信頼性の高いイベントのみがプロセッサに到達します。上記のカスタムワークフローでは、パートナートピックまたはドメインを作成する必要はありません。
学習ガイドの関連性 :Azureサービス統合 :Service Bus、Event Grid、Azure Functionsトリガー
バインディング、およびイベント駆動型処理。
Microsoft Learn 公式資料: AI-200 学習ガイド | イベント グリッドのカスタム トピックまたはドメインを作成する

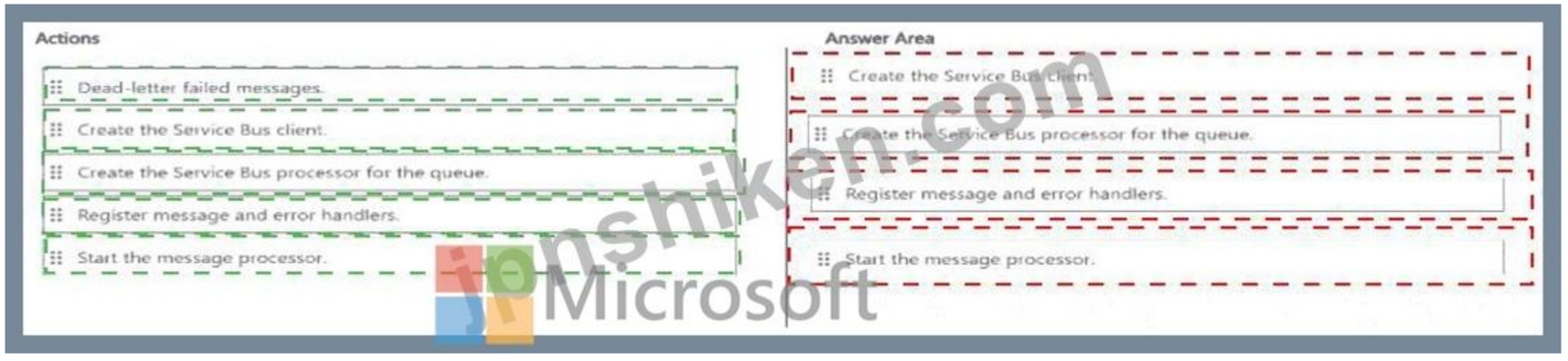
質問: **11**
PostgreSQLへの同時アプリケーションリクエストのスループットを改善する必要があります。
何を実装すべきでしょうか？
A. コネクションプーリングを実装します。
B. shared_buffers を増やします。
C. 読み取りレプリカを有効にします。
D. max_connectionsを増やします。
正解: **A (**コメントを発表する**)**
適切な解決策は、接続プーリングを実装することです。理想的には、Azure Database for PostgreSQL Flexible Server に組み込まれている PgBouncer 機能を使用するのが良いでしょう。接続プーリングを使用すると、データベースセッションを繰り返し開閉するのではなく、多数の同時アプリケーション要求が確立済みの PostgreSQL サーバー接続の小さなセットを共有して再利用できます。Microsoft は、操作ごとに新しい PostgreSQL 接続を作成すると、各接続に独自のバックエンド プロセスが関与するため、サーバー リソースが消費されると指摘しています。したがって、プールされた接続を再利用することで、接続のオーバーヘッドが削減され、CPU とメモリの消費量が減り、高並行ワークロードのスループットが向上します。
これは、Fabrikamが掲げる データベース接続は、接続最適化によって最小限の遅延で高い同時実行性をサポートする」という要件に完全に合致しています。PgBouncerは、特にトランザクションプーリングモードにおいて、多数のクライアント接続を少数のPostgreSQL接続に効率的に多重化できます。

max_connections を増やすことは推奨される解決策ではありません。PostgreSQL の接続が増えるごとにメモリやその他のリソースが消費されるためです。Microsoft は、より多くの同時接続が必要な場合は PgBouncer の使用を明示的に推奨しています。shared_buffers を増やすことは、接続管理のオーバーヘッドではなく、キャッシュとメモリ割り当ての問題に対処するものです。リードレプリカは読み取りワークロードをスケーリングできますが、アプリケーションの接続ライフサイクルを最適化するものではありません。
学習ガイドの参照: Azure Database for PostgreSQL Flexible Server # コネクションプーリング、PgBouncer、コネクション管理、高同時実行パフォーマンス。

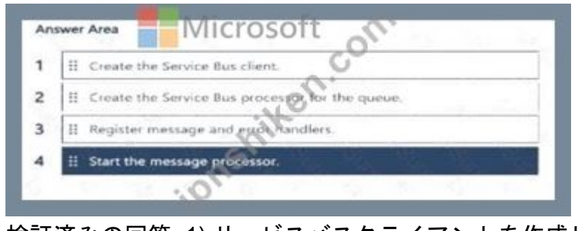
質問: **12**
AIバッチジョブ用のサービスバスプロセッサをプロビジョニングおよび構成しています。
プロセッサは既存のキューに接続し、メッセージおよびエラー処理のためのハンドラを登録した後、メッセージの受信を開始する必要があります。
メッセージおよびエラー処理のために、サービスバスプロセッサをプロビジョニングおよび構成する必要があります。
どの4つの行動を順番に実行すべきでしょうか？回答するには、行動リストから適切な行動を回答欄に移動させ、正しい順序に並べ替えてください。



正解:



Explanation:



検証済みの回答: 1) サービスバスクライアントを作成します。2) キュープロセッサ/レシーバを作成します。3) メッセージハンドラとエラーハンドラを登録します。4) メッセージプロセッサを開始します。
詳細説明 :プロセッサは名前空間クライアント/接続に依存するため、最初にクライアントが作成されます。

次に、そのクライアントからキュー固有のプロセッサまたはレシーバが作成されます。メッセージとエラーのコールバックは、処理開始前に登録しておく必要があります。これにより、受信メッセージとエラーに対して定義済みのハンドラが設定されます。したがって、プロセッサの起動が最終ステップとなります。失敗したメッセージのデッドレター化は、アプリケーションのオプションの決済決定であり、プロセッサの構築と起動の前提条件ではありません。
学習ガイドの関連性 :Azureサービス統合 :Service Bus、Event Grid、Azure Functionsトリガー
バインディング、およびイベント駆動型処理。
Microsoft Learn 公式資料: AI-200 学習ガイド | サービスバスのメッセージ転送、ロック、および決済

質問: **13**
あなたは、ユーザーがHTTPエンドポイント経由で送信した大容量画像のアップロードを処理するAzure Functionsアプリを設計しています。
解決策は以下の条件を満たす必要があります。
- 画像処理を分離することでクライアントのタイムアウトを防ぎます
最初のアップロードリクエスト。
処理失敗時の自動再試行動作をサポートします。
- バックグラウンド処理を、受信するHTTPアップロードの速度とは独立してスケーリングする。
拡張性と信頼性に優れた非同期処理ソリューションを設計する必要があります。
どの2つの行動を実行すべきでしょうか？それぞれの正解は、解決策の一部を示しています。
注：正解ごとに1ポイントが加算されます。
A. 画像処理のためのキュートリガー関数を実装する。
B. ストレージキューで再試行ポリシーを設定します。
C. HTTPトリガー関数内で画像を処理します。
D. アップロードされたメタデータをAzureストレージに永続化します。
正解: **A,C** (コメントを発表する)
要件を満たす、拡張性と信頼性に優れた非同期ソリューションを構築するには、HTTP トリガー関数を使用して画像を Azure Blob Storage に保存し、メッセージをキューに追加してから、キュー トリガー関数を使用して画像を処理する必要があります。
1. キューメッセージングによる分離
HTTP 関数: 初回のユーザーアップロードを受信し、大きな画像ファイルを Azure Blob Storage に直接保存し、小さなメタデータ メッセージ (Blob URI など) を Azure Storage キューまたは Azure Service Bus キューに書き込みます。
即時応答 :HTTP関数は、ステータス追跡URLとともに、クライアントに即座に202 Acceptedステータスを返します。これにより、負荷の高い処理をバックグラウンドに移行することで、クライアントのタイムアウト問題を完全に解消します。
2. 組み込みキュートリガーを活用する
キュートリガー関数 :2つ目の独立したAzure Functionが、キューに新しいメッセージが到着するたびにトリガーされるように構成されています。この関数は、Blob Storageからイメージをダウンロードし、処理を実行します。
自動再試行: Azure Queue ストレージと Service Bus トリガーには、ネイティブの再試行機能が備わっています。処理関数が失敗またはクラッシュした場合、メッセージは自動的にキューに戻され、再試行されます。永続的な失敗は、設定された試行回数 (既定値は 5 回) の後、自動的にポイズンキュー (デッドレターキュー) に移動されます。
独立したスケーリング :Azure 消費プランまたはプレミアムプランでは、キュートリガー関数はキューの長さ（処理対象の画像のバックログ）に基づいてインスタンスをスケールアップまたはスケールダウンします。これは、受信リクエスト量のみに基づいてスケールする HTTP トリガー関数とは独立してスケーリングされます。
参照 :
https://oneuptime.com/blog/post/2026-02-16-azure-functions-python-http-triggers-blob-storage/view

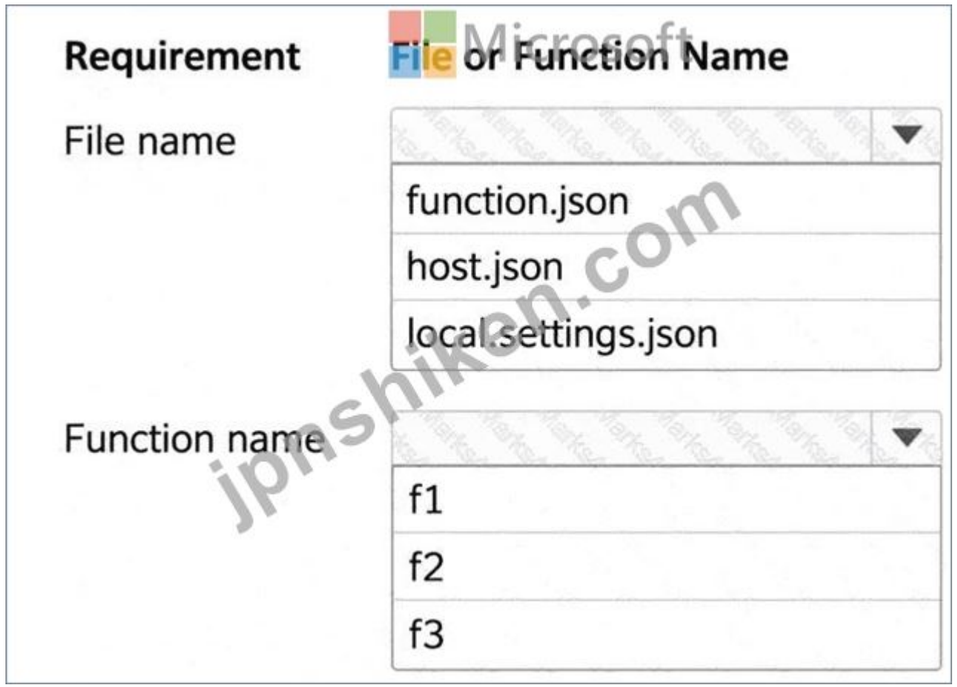
質問: **14**

ホットスポットに関する質問
企業向けの従量課金ホスティングプランを使用している Azure Functions アプリがあります。このアプリには、次の関数が含まれています。

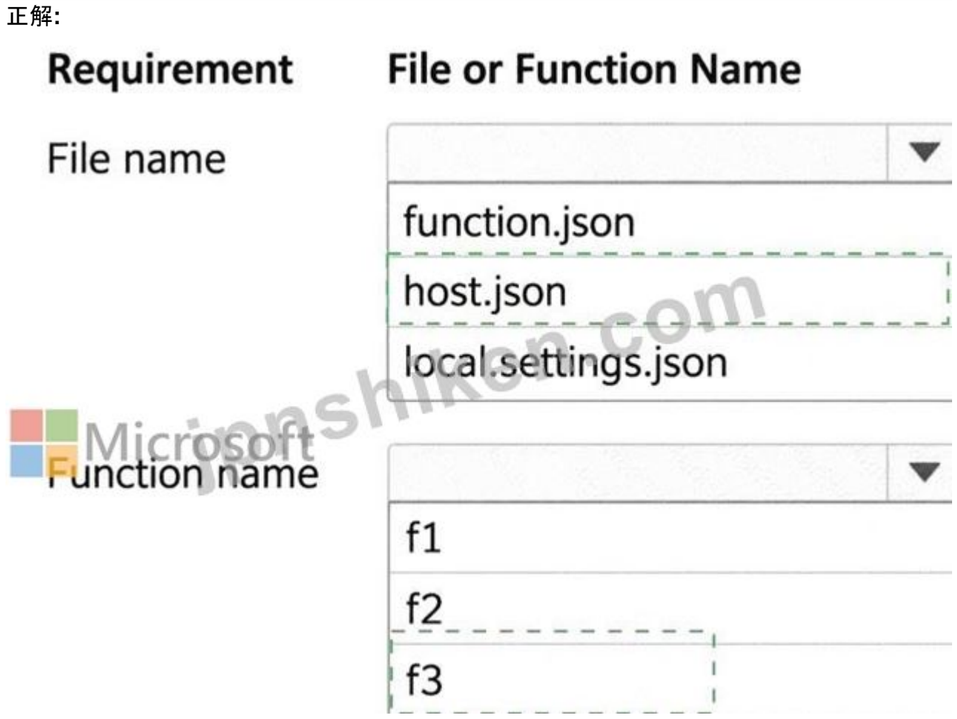
Function Name	Trigger type
f1	HTTP
f2	Timer
f3	Azure Queues

jpnshiken.com
Microsoft

アプリで動的な同時実行機能を有効にする予定です。会社側は、各関数の同時実行レベルを個別に管理することを求めています。
アプリを動的同時実行に対応させるように設定する必要があります。
どのファイル名または関数名を使用すべきですか？回答するには、回答欄で適切な値を選択してください。
注：正解ごとに1ポイントが加算されます。



正解:



Explanation:

* ファイル名: host.json

* 関数名: f3
Azure Functions の動的同時実行機能は、関数アプリのホスト レベルで host.json ファイルによって有効化されます。同時実行構成セクションには dynamicConcurrencyEnabled プロパティが含まれています。この値を true に設定すると、Functions ホストは、固定の手動制限を必要とせず、サポートされているトリガーの同時実行レベルを動的に学習して調整できるようになります。
動的並行処理はグローバルに有効になっていますが、Microsoft は、学習された並行処理レベルは個々の関数ごとに独立して管理されると明言しています。これにより、リソースを大量に消費する関数は低い並行処理レベルで動作し、同じアプリケーション内の軽量な関数はより高い並行処理レベルで実行できるため、ホストの健全性を維持しながらスループットを最大化できます。
記載されている関数の中で、Azure Queue Storage トリガーを使用する f3 は、動的な同時実行をサポートしています。
マイクロソフトは現在、必要な拡張機能バージョンを条件として、Azure Queue Storage、Azure Blob Storage、および Azure Service Bus トリガーの動的同時実行サポートについてドキュメント化しています。HTTP および Timer トリガーは、この動的同時実行モデルには参加しません。
したがって、host.json で動的な同時実行を設定し、f3 は動的に同時実行が管理される関数です。
学習ガイドの参照: Azure Functions # 同時実行; 動的同時実行; host.json; キュー ストレージ トリガーの同時実行。

質問**: 15**
ホットスポットに関する質問
食品配達の料金支払いに使用されるWebサービスがあります。このWebサービスは、データストアとしてAzure Cosmos DBを使用しています。
ユーザーがチップの金額を設定できる新機能を追加する予定です。この新機能を使用するには、Azure Cosmos DB のドキュメントに tip という名前のプロパティが存在し、数値が含まれている必要があります。
ウェブサービスを利用している既存のウェブサイトやモバイルアプリの多くは、しばらくの間、チッププロパティを設定するための更新が行われないだろう。
トリガーをどのように完了すればよいですか？
注：正解ごとに1ポイントが加算されます。

Answer Area

```
function ensureTip() {
    var r = [ __.value(); | __.readDocument('item'); | getContext().getRequest(); | getContext().getResponse(); ]
    var i = r.getBody();
    [ if (!("tip" in i)) { | if (request.getValue("tip") === null) { | if (isNaN(i["tip"]) || i["tip"] === null) { | if (typeof __.pluck("tip") == 'number') { ]
        i["tip"] = 0;
    }
    [ r.setBody(i); | r.setValue(i); | __.upsertDocument(i); | __.replaceDocument(i); ]
}
```

正解:

Microsoft
Ipnshiken.com

Answer Area

```
function ensureTip() {
  var r = [ getContext().getRequest(); ]
  var i = r.getBody();
  [ if (!("tip" in i)) { ]
  i["tip"] = 0;
  }
  [ r.setBody(i); ]
}
```

- var r = : __.value(); / __.readDocument('item'); / **getContext().getRequest();** / getContext().getResponse();
- if (!("tip" in i)) { / if (request.getValue("tip") === null) { / **if (isNaN(i)["tip"] || i["tip"] === null) {** / if (typeof __.pluck("tip") == 'number') {
- r.setBody(i); / r.setValue(i); / __.upsertDocument(i); / __.replaceDocument(i);

jpnshiken.com

Explanation:
ボックス1 :getContext().getRequest();
Y: if (!i["tip"]) { (または、試験問題のバージョンによっては)Z:
ボックス2 :if (isNaN(i["tip"]) || i["tip"] === null) {
ボックス3 :r.setBody(i);

質問: **16**
あなたのソリューションは、例えば以下のような大規模なExcelベースのレポートに含まれる数値データに関する質問に答える必要があります。
「西部地域の第3四半期の売上高はいくらでしたか？」
どの方法が最も適切でしょうか？
A. Excelデータを非構造化テキストとして保存し、ベクトル検索用に埋め込む
B. データを構造化テーブル/SQLソースに変換し、関数呼び出しエージェントを使用してクエリを実行します。
C. モデルのパラメータ知識に頼る
D. Azure AI Visionを使用してスプレッドシートを画像として読み込む
正解: **B (**コメントを発表する**)**

表形式の数値データに対する正確な質問を行うには、ワークブックのデータを構造化されたリレーショナル/テーブル表現に変換し、関数呼び出しによって制御されたクエリ機能をAIアプリケーションに公開します。これにより、四半期、地域、収益、製品、会計期間などの列が型付きフィールドとして保持され、フィルタリング、集計、および決定論的な比較が可能になります。Microsoftは、ユーザーの質問を構造化されたPostgreSQLデータに対するクエリに変換する自然言語からSQLへのパターンを示しており、Foundryの関数呼び出しによって、モデルまたはエージェントが構造化されたパラメーターを使用してアプリケーション定義のクエリツールを呼び出すことができます。
これは、スプレッドシートの行を非構造化テキストとして埋め込むよりも、はるかに信頼性が高い。ベクトル検索は、厳密な算術演算、グループ化、フィルタリング、集計ではなく、意味的な類似性に基づいて最適化されている。 西地域の第3四半期の収益」のようなクエリは、基本的に構造化されたルックアップまたは集計の問題である。SQLはこれらの制約を明示的に表現し、信頼できる数値結果を返すことができる。
Microsoft Foundryは、構造化データを扱うエージェントもサポートしており、エージェントツールが数値ファイルを分析して計算値を返す例も提供しています。
パラメトリックモデルの知識に頼るだけでは、組織固有の最新の財務数値を提供することはできません。Azure AI Vision はスプレッドシートの画像を OCR で認識できますが、既に構造化されたワークブックを画像に変換すると、有用なスキーマが破壊され、抽出エラーのリスクが高まります。
学習ガイドの参照項目 :Microsoft Foundryエージェント、関数呼び出し、構造化データアクセス、自然言語からSQLへの変換、エンタープライズデータに基づくAI応答の構築。

有効的な**AI-200**問題集はJPNTest.com提供され、**AI-200**試験に合格することに役に立ちます！JPNTest.comは今最新**AI-200**試験問題集を提供します。JPNTest.com AI-200試験問題集はもう更新されました。ここで**AI-200**問題集のテストエンジンを手に入れます。最新版のアクセス、https://www.jpntest.com/shiken/AI-200-mondaishu 　**144**問、３０%ディスカウント、特別な割引コード: **JPNshiken**」

質問: **17**
注 :この問題は、同じシナリオを提示する一連の問題の一部です。このシリーズの各問題には、提示された目標を満たす可能性のある独自の解答が含まれています。問題セットによっては、複数の正解がある場合もあれば、正解がない場合もあります。
このセクションの質問に回答すると、後から戻って回答することはできません。そのため、これらの質問は復習画面には表示されません。
Azure App Service API アプリ（api1 という名前）にコンテナをデプロイする予定です。api1 のソースコードは GitHub リポジトリにホストされています。コンテナは実行時に API キーを使用してバックエンド サービスに接続します。
コンテナは、ソースリポジトリやGitコミット履歴にAPIキーを公開することなく、実行時にAPIキーを取得できる必要があります。
APIキーはGitのコミット履歴に含まれないようにし、実行時にコンテナからアクセスできるようにする必要があります。
解決策 :APIキーをAzure Key Vaultに保存し、App Serviceアプリケーションの設定から参照します。
その解決策は目標を達成しているか？
A. はい
B. いいえ
正解: **(正解を表示します)**
正しい：
* API キーを Azure Key Vault に保存し、App Service アプリケーションの設定から参照します。
APIキーをAzure Key Vaultに保存し、App Serviceアプリケーション設定から参照するのが、推奨される安全な方法です。この方法により、機密性の高い認証情報がGitHubリポジトリやGitコミット履歴から完全に削除され、実行時にコンテナ環境に安全に注入されます。
正しくない：
* APIキーをハードコードされた環境変数としてDockerfileに埋め込む。
* APIキーをGitHubリポジトリのシークレットとして保存します。
参照：
https://www.qservicesit.com/full-stack-applications-on-azure

質問: **18**

Azure Database for PostgreSQLインスタンスのデータベースリソースを構成する必要があります。
ビジネス要件と技術要件を満たすために、どのように設定を完了すればよいでしょうか？回答するには、回答欄で適切なオプションを選択してください。
注：正解ごとに1ポイントが加算されます。

PostgreSQL compute, memory, and storage configuration for vector workloads

Requirement	Configuration
Meet the 200ms semantic search latency requirement.	Increase compute vCores. Increase max_connections. Increase memory allocation.
Optimize the environment for high-dimensional pgvector index residency.	Enable read replica. Increase max_connections. Increase memory allocation.
Support the continuous ingestion of transaction-based embeddings.	Enable storage autoscale. Increase compute vCores. Increase backup retention.

Microsoft
Jpnshiken.com

正解:

PostgreSQL compute, memory, and storage configuration for vector workloads

Requirement	Configuration
Meet the 200ms semantic search latency requirement.	Increase compute vCores. / Increase max_connections. / Increase memory allocation.
Optimize the environment for high-dimensional pgvector index residency.	Enable read replica. / Increase max_connections. / Increase memory allocation.
Support the continuous ingestion of transaction-based embeddings.	Enable storage autoscale. / Increase compute vCores. / Increase backup retention.

jpnshiken.com

Explanation:
* 200ミリ秒のセマンティック検索レイテンシ要件を満たす :コンピューティングvCoreを増やす。
* 高次元pgvectorインデックスの常駐環境を最適化する :メモリ割り当てを増やす。
* トランザクションベースの埋め込みの継続的な取り込みをサポートする :ストレージの自動スケーリングを有効にする。
200ミリ秒未満の厳密なベクトル検索レイテンシを目標とする場合、計算用vCore数を増やすことが適切な選択肢です。
ベクトル類似性演算は計算負荷が高く、CPU容量を増やすことで数学的スループットと並列クエリ実行が向上します。Microsoftのpgvectorに関するガイダンスでは、ベクトルワークロードを最適化する際に、クエリプランの最適化、HNSWなどのANNインデックス、および十分な計算リソースの確保を推奨しています。
高次元の pgvector インデックスの常駐性を確保するには、メモリ割り当てを増やしてください。HNSW は優れたクエリ性能を発揮しますが、IVFFlat よりも多くのメモリを消費します。頻繁にアクセスされるベクターインデックス構造をメモリに保持することで、ディスクアクセスを最小限に抑え、レイテンシを大幅に改善できます。Microsoft は、HNSW はより多くのメモリを必要とする一方で、速度とリコールのトレードオフにおいて優れていることを明記しています。
数百万件の埋め込みデータを継続的に取り込むには、ストレージの自動スケーリングを有効にしてください。Azure Database for PostgreSQL Flexible Server は、容量が設定されたしきい値に近づくと割り当て済みストレージを自動的に増やし、データ量の増加に伴うストレージ不足を回避します。Microsoft は、ストレージ需要が動的に増加するワークロードに対して、ストレージの自動拡張を推奨しています。
max_connections を増やしても、ベクトル計算やインデックスの常駐性は直接的に改善されません。リードレプリカは主にリードをスケーリングし、バックアップ保持は取り込み容量には対応していません。
学習ガイドの参照: Azure Database for PostgreSQL Flexible Server # pgvector パフォーマンス最適化、コンピューティングとメモリのサイジング、HNSW インデックス作成、ストレージの自動拡張。

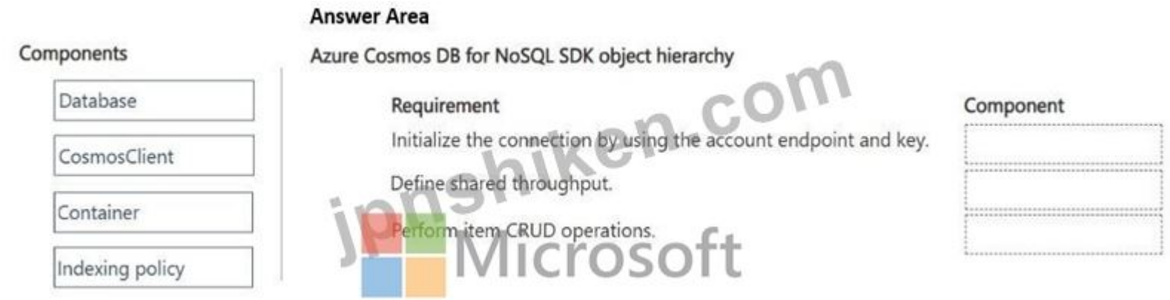
質問: **19**
カスタムAI推論イベントをイベントグリッドトピックに発行します。

下流サービスは、confidenceScoreの値が0.80より大きい場合にのみイベントを受信する必要があります。
イベントペイロード内の数値に基づいてイベントをフィルタリングする必要があります。
どのフィルターを設定すればよいですか？
A. イベントタイプ
B. 被験者
C. 上級
正解**: C (**コメントを発表する**)**
ペイロード値（例data.confidenceScore）を指すキーパスと、比較値0.80を設定したNumberGreaterThan演算子を使用した高度なフィルタを設定する必要があります。
Azure Event Grid で数値しきい値に基づいてカスタムペイロードデータをフィルタリングするには、イベントサブスクリプションを次のプロパティで構成します。
キー: data.confidenceScore (JSON ペイロードの構造に応じてパスを調整してください) 演算子: NumberGreaterThanValue: 0.80 参照:
https://learn.microsoft.com/en-us/azure/event-grid/event-filtering

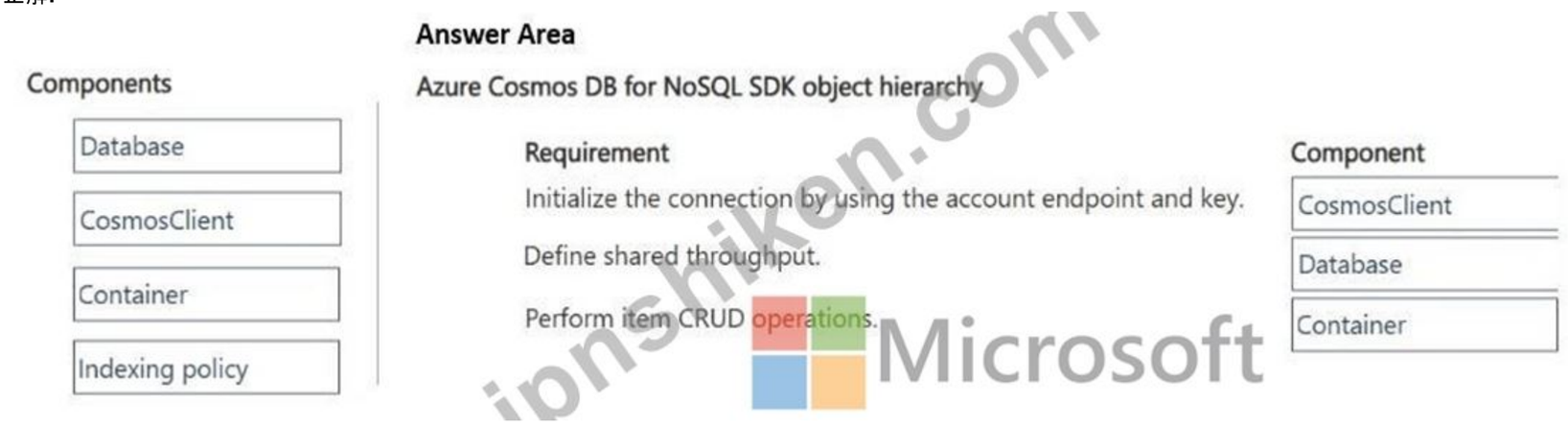
質問**: 20**
コンテナのソースコードはGitリポジトリに保存されています。
コンテナレジストリは、開発者がGitリポジトリにコードをコミットするたびに、新しいコンテナイメージを自動的に構築して保存する必要があります。
外部のビルドインフラストラクチャの使用は最小限に抑える必要があります。
ビルドプロセスをネイティブかつ自動的に管理するには、Azure Container Registry (ACR) を構成する必要があります。
どの2つのACRコンポーネントを使用すべきでしょうか？それぞれの正解は、解決策の一部を示しています。
注：正解ごとに1ポイントが加算されます。
A. Webhook
B. クイックタスク
C. アーティファクトキャッシュルール
D. ソーストリガー型タスク
正解**: (**正解を表示します**)**
Azure Container Registry Tasks は、Azure でホストされるコンテナ構築機能を提供し、外部の Docker ビルド サーバーや専用の CI ビルド インフラストラクチャを維持する必要性を排除します。クイック タスクは、az acr build などのコマンドを使用してビルド コンテキストを ACR マネージド コンピューティングに送信します。Azure はそこでコンテナ イメージを構築し、既定では、正常に構築されたイメージをレジストリに直接プッシュします。Microsoft は、クイック タスクを、ローカルの Docker Engine を必要としない、クラウド ベースのオンデマンド構築およびプッシュ操作と説明しています。
ソースコードの変更後に自動的に実行するには、ソーストリガー型タスクを使用します。ACRタスクは、タスクをGitHubまたはAzure Reposリポジトリに関連付け、構成済みのブランチにコードがコミットされるたびにイメージビルドを自動的に実行できます。Microsoftのドキュメントによると、コミットが発生すると、ソース管理トリガーによってACRタスクがビルドを実行し、結果のイメージをプッシュします。
Webhookは、ここで必須となるスタンドアロンのACRビルドコンポーネントではありません。ACRタスクは、ソーストリガーの設定の一環として、必要なソースリポジトリWebhookを作成および管理します。アーティファクトキャッシュルールは、アップストリームのコンテナアーティファクトをキャッシュし、Gitコミットからアプリケーションイメージをビルドしません。
学習ガイドの参照: Azure Container Registry タスク # クイック タスク、自動的にトリガーされるタスク、ソース コード コミット トリガー、クラウド ベースのイメージ ビルド。

質問**: 21**
ドラッグアンドドロップ問題
あなたは、アプリケーションデータを保存するためにAzure Cosmos DB for NoSQLを使用する.NETアプリケーションを開発しています。
このアプリケーションは、Azure Cosmos DB for NoSQL SDKを使用してデータベースアカウントとやり取りします。
アプリケーションは以下のタスクを実行する必要があります。
アカウントのエンドポイントとキーを使用して接続を初期化します。
共有スループットを定義します。

- アイテムに対して作成、読み取り、更新、削除（CRUD）操作を実行する
容器に保管されている。
アプリケーションがAzure Cosmos DB for NoSQL内のデータにアクセスして管理するために必要なSDKコンポーネントを実装する必要があります。
どのSDKコンポーネントを使用すべきでしょうか？回答するには、適切なコンポーネントを正しい要件に移動してください。各コンポーネントは、1回、複数回、またはまったく使用しない場合があります。コンテンツを表示するには、ペイン間の分割バーを移動したり、スクロールしたりする必要がある場合があります。
注：正解ごとに1ポイントが加算されます。



正解:



Explanation:
ボックス1 :CosmosClient
アカウントエンドポイントとキーを使用してAzure Cosmos DB for NoSQLアカウントへの接続を初期化するには、CosmosClientクラスを使用する必要があります。
ボックス2 :データベース
共有スループットを定義するには、データベースSDKコンポーネントを使用する必要があります。
Azure Cosmos DB では、共有スループット (プロビジョニングされたリクエスト ユニット/秒、RU/s) はデータベース レベルで構成されます。データベースにスループットがプロビジョニングされると、その容量は、その特定のデータベース内に作成されたすべてのコンテナー間で共有されます。
ボックス3 :コンテナ
アイテムの作成、読み取り、更新、削除 (CRUD) 操作を実行するには、Container SDK コンポーネントを使用する必要があります。Azure Cosmos DB for NoSQL SDK for .NET では、個々の JSON ドキュメント (アイテム) はコンテナ内に存在します。Container クラスは、これらのアイテムに対して CRUD 操作を実行するために必要な特定のメソッドを公開します。
参照：
https://learn.microsoft.com/en-us/azure/cosmos-db/how-to-dotnet-get-started

質問: **22**
ドラッグアンドドロップ問題
あなたは、Key Vaultから認証情報を取得するAIアプリケーションが使用する秘密アクセスパターンを確認しています。
各実装方法がセキュリティに与える影響を評価する必要があります。
それぞれの方法の結果はどうなりますか？回答するには、適切な結果を正しい実装に移動してください。各結果は、1回、複数回、またはまったく使用しない場合があります。コンテンツを表示するには、ペイン間の分割バーを移動したり、スクロールしたりする必要がある場合があります。
注：正解ごとに1ポイントが加算されます。



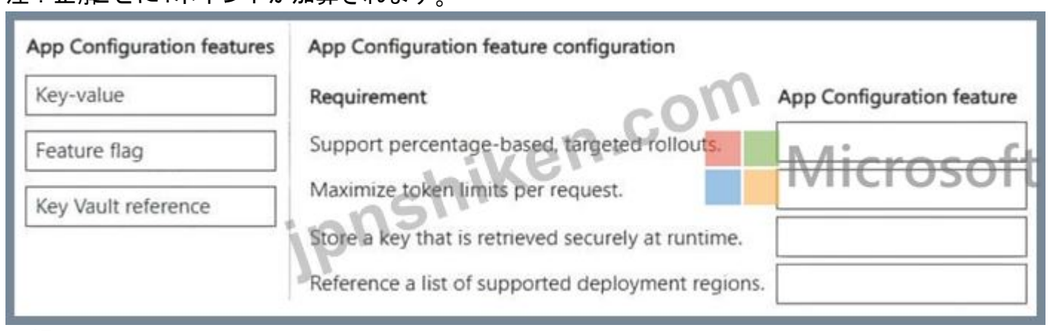
正解:



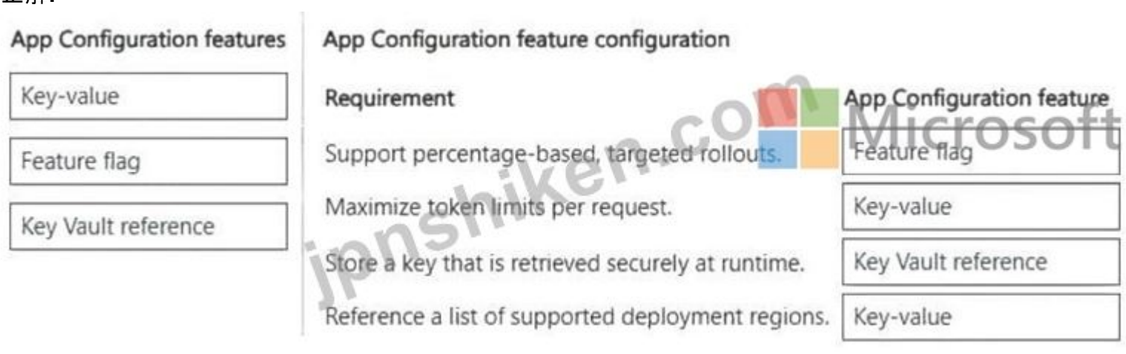
Explanation:
ボックス1：自動回転に対応しています。
AIアプリケーションがバージョンを指定せずにAzure Key Vaultからシークレットを取得する場合、常に最新のアクティブバージョンが取得されます。この選択は、セキュリティと運用に次のような影響を与えます。
最新バージョンの取得：保管庫は現在有効な秘密の値を自動的に提供します。
コード変更不要 :アプリケーションは、コードの再デプロイなしに、ローテーション後すぐに新しい認証情報を読み取ります。
ダウンタイムゼロ：自動化されたローテーションパイプラインにより、アプリが新しいトークンにシームレスに切り替わる間、Key Vault内のシークレットを更新できます。
ボックス2：回転後に手動更新が必要
Key Vaultからシークレットを取得する際に特定のバージョン識別子を使用すると、アプリケーションは常にそのシークレットの正確なバージョンを要求します。この選択では、アプリケーションコードまたは構成によって特定のバージョンIDが固定されるため、シークレットがローテーションされた後に手動で更新する必要があります。ローテーション後、新しい値は取得されません。
ボックス3：認証情報の保存を防止します
認証にマネージドIDを使用することで、認証情報の保存を防ぐことができるのは正しい結果です。
認証情報の管理は不要 :マネージドIDにより、開発者は認証情報を一切管理する必要がなくなります。
トークンの自動処理 :Azure は、Microsoft Entra ID でアプリケーションの ID を自動的に作成し、サービス プリンシパルのライフサイクルをバックグラウンドで処理するため、機密情報や接続文字列がハードコーディングされたり、構成ファイルに保存されたりすることはありません。

ボックス4：資格情報の公開
環境変数にクライアントシークレットを保存することによる主な結果は、認証情報の漏洩リスクが生じることと、ローテーション後に手動での更新が必要になることです。
環境変数に平文の機密情報を保存することは、重大なセキュリティリスクとなります。実行環境（コンテナの詳細、アプリケーションログ、プロセスダンプなど）にアクセスできるプロセス、侵害された依存関係、またはユーザーは、それらを容易に読み取ることができます。
参照：
https://medium.com/simform-engineering/building-automated-secret-rotation-using-azure-key-vault-and-net-01e0701f0a2e
https://www.varonis.com/blog/azure-managed-identities

質問: **23**
ドラッグアンドドロップ問題
Azure App Configurationを使用してランタイム設定を管理する、AzureでホストされているAIアプリケーションを構成しています。
アプリケーションは以下の機能を提供する必要があります。
- パーセンテージに基づいた、対象を絞った展開をサポートします。
- リクエストごとのトークン上限を最大化する。
実行時に安全に取得できるキーを保存します。
サポートされている展開リージョンの一覧を参照してください。
アプリの設定機能を構成する必要があります。
どの機能を設定する必要がありますか？回答するには、適切なアプリ構成機能を正しい構成項目に移動してください。各アプリ構成機能は、1回、複数回、またはまったく使用しない場合があります。コンテンツを表示するには、ペイン間の分割バーを移動したり、スクロールしたりする必要がある場合があります。
注：正解ごとに1ポイントが加算されます。



正解:

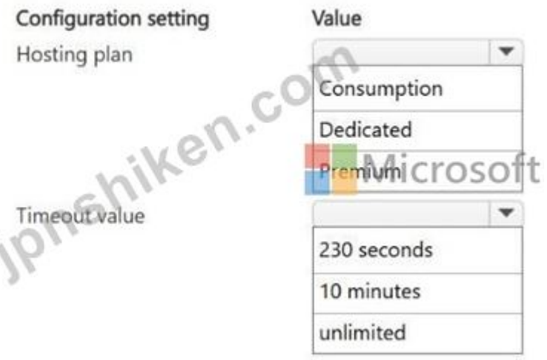


Explanation:

ボックス1：機能フラグ
機能フラグを使用してください。
Azure App Configuration は、コードを再デプロイすることなくアプリケーションの動作を動的に制御するために特別に設計された、組み込みの機能管理機能を提供します。
パーセンテージベースのロールアウト: 機能フラグは、パーセンテージベースのロールアウトや特定のオーディエンスのターゲティングルールをネイティブに処理するターゲティングフィルター（標準の Microsoft.Targeting フィルターなど）をサポートしています。
動的な変更 :Azure ポータルまたは API を介して、機能を安全に切り替えたり、露出率をその場で変更したりできます。
ボックス2 :キーと値のペア
キーバリュー型が適切な機能です。
数値しきい値やトークン制限などの実行時設定を管理するには、標準的なキーと値のペアが最適な選択肢です。
直接データ注入：特定の最大トークン制限（例MaxTokensPerRequest = 4096）を整数または文字列として保存し、アプリケーションが実行時に直接解析できるようにします。
余分なオーバーヘッドなし：他のオプションとは異なり、標準的なキーと値のペアは、二次的なルックアップや条件付きロジックの評価を必要とせずに、アプリケーション構成変数に直接マッピングされます。
ボックス3 :キー保管庫の参照
Azure Key Vault のリファレンスを使用してください。
Azure App Configuration は、機能フラグ、環境名、一般的なアプリケーション設定など、機密性の低いランタイム構成に最適化されています。機密性の高い値（OpenAI/LLM モデルの API キー、データベースのパスワード、接続文字列など）は、常に Azure Key Vault に直接保存する必要があります。
Azure App Configuration 内に Key Vault 参照を作成することで、シークレット値自体ではなく、シークレットを指す URI のみを保存します。AI アプリケーションが実行時に構成データを要求すると、App Configuration プロバイダーは自動的に Key Vault に接続し、参照を解決して、実際のシークレットをアプリケーションのメモリ パイプラインに安全に直接挿入します。
ボックス4 :キーと値のペア
標準的な設定キーと値のペア（またはキー内のJSON配列）を使用してください。
保存方法：地域をカンマ区切りの文字列、または単一のキーを持つJSON配列として保存します。
重要な例: デプロイメント:SupportedRegionsValue 例: ["eastus", "westeurope",
東南アジア」
コンテンツタイプ: アプリケーションが実行時に自動的にリストまたは配列に解析できるように、コンテンツタイプをapplication/jsonに設定します。
参照：
https://learn.microsoft.com/en-us/azure/azure-app-configuration/manage-feature-flags
https://turbo360.com/blog/get-azure-key-value-pairs-from-app-configuration
https://learn.microsoft.com/en-us/azure/app-service/app-service-key-vault-references

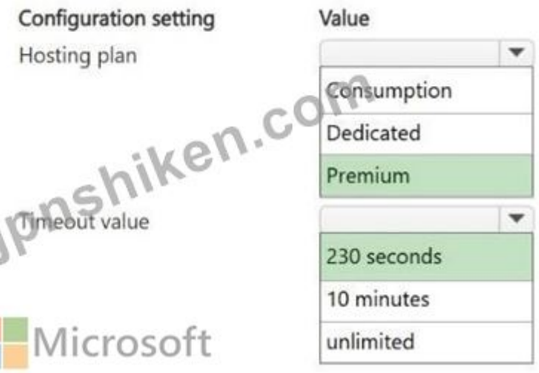
質問**: 24**
ホットスポットに関する質問
あなたはHTTPトリガーを使用したAzure Functionsアプリを開発する予定です。
アプリは以下の機能をサポートする必要があります。
- イベント駆動型スケーリング
- 関数実行にカスタムLinuxイメージを使用できる機能
アプリのホスティングプランと、アプリの機能が受信リクエストに応答するまでにかかる最大時間を特定する必要があります。
どの設定値を使用すべきですか？回答するには、回答欄で適切な値を選択してください。
注：正解ごとに1ポイントが加算されます。

Answer Area



正解:

Answer Area



Explanation:

ボックス1 :プレミアム

お客様のご要望を満たすには、Azure Functions Premiumプラン（Elastic Premiumプランとも呼ばれます）をご利用いただく必要があります。

イベント駆動型スケーリング :Azure スケール コントローラーによって駆動される、動的で自動的なスケールアウト機能を備えています。アイドル状態のインスタンスはゼロまでスケールダウンできるため、アクティブなコンピューティング時間に対してのみ料金が発生します。

カスタムLinuxイメージ：基本の従量課金プランとは異なり、プレミアムプランでは、カスタムLinuxコンテナイメージ内で関数をデプロイおよび実行できます。これにより、独自のカスタムOS依存関係、専用ツール、または特定のランタイム環境を持ち込むことができます。

ボックス2 :230秒

HTTPトリガー関数に使用する適切な最大タイムアウト値は230秒です。

Azure Load Balancer の制限: Azure Functions のホスティング プランや host.json で設定したタイムアウト構成に関係なく、Azure Functions のスケールとホスティングに関するドキュメントには、HTTP トリガー関数がリクエストに応答するまでの時間が 230 秒に制限されていると記載されています。

アイドルタイムアウト :この制約は、基盤となるAzureロードバランサーのデフォルトのアイドルタイムアウトによって厳密に適用されます。関数が230秒以上応答を返さずに実行されると、接続が切断され、タイムアウトエラーが発生します。

参照：

https://learn.microsoft.com/en-us/azure/azure-functions/functions-deployment-technologies

https://learn.microsoft.com/en-us/azure/azure-functions/functions-scale

質問: **25**
あなたはチャットボット向けに意味検索機能を実装しています。
埋め込みデータは既にRedisに保存されています。しかし、ベクトル類似性クエリでは一致する結果が返されません。
ベクトル類似性検索の問題を解決する必要があります。
あなたはどうすべきでしょうか？
A. 埋め込みキーに24時間の有効期限（TTL）を設定します。
B. 地理的レプリケーションを有効にします。
C. 埋め込みフィールド上にHNSWベクトルインデックスを作成します。
D. 読み取りごとに有効期限をリセットします。
正解: **(**正解を表示します**)**

質問: **26**
Redisでは、doc:(id)という形式のキーを使用して埋め込みデータを保存します。埋め込みデータの中には頻繁にアクセスされるものもあれば、ほとんど使用されないものもあります。
頻繁にアクセスされる埋め込みデータのみをメモリに保持するキャッシュ戦略を実装する必要があります。
何を使うべきでしょうか？
A. volatile-ttl
B. allkeys-lru
C. EXPIREコマンド
D. 時間枠の期限切れ
正解: **(**正解を表示します**)**
この戦略を実行するには、allkeys-lru と EXPIRE コマンドを組み合わせて、二次的なフォールバックとして使用する必要があります。
主なメカニズム: Redis の maxmemory-policy を allkeys-lru に設定します。
二次的な対策：安全策として、キーにEXPIREコマンドを適用してください。
allkeys-lruが最良の選択肢である理由
メモリ管理：Redisがメモリ制限に達すると、データセット全体から最も使用頻度の低い（LRU）キーを自動的に削除します。
頻繁なアクセス: 作成時期に関係なく、頻繁にアクセスされる埋め込みがメモリに保持されることを保証します。プレフィックス非依存: すべてのキーをスキャンするため、ドキュメント:(id) フォーマットの参照に最適です。
https://rahulchowdhury.in/blog/redis-caching-patterns-every-mern-dev

質問: **27**
注：この問題は、同じシナリオを提示する一連の問題の一部です。このシリーズの各問題には、提示された目標を満たす可能性のある独自の解答が含まれています。問題セットによっては、複数の正解がある場合もあれば、正解がない場合もあります。
このセクションの質問に回答すると、後から戻って回答することはできません。そのため、これらの質問は復習画面には表示されません。
Azure Monitor Application Insights を使用して、運用環境の API を調査しています。ログ ブレードを開き、時間範囲を「過去 24 時間」に設定します。
エンジニアは、結果コードごとにリクエスト数をカウントし、結果を頻度の高い順から低い順に並べ替えるために、次のクエリを推奨しています。
リクエスト
| リクエスト数 = カウント() を結果コード別に集計
| リクエスト数降順で並べ替え
クエリが、結果コードごとにグループ化され、頻度の高い順から低い順に並べられたリクエスト数を返すかどうかを判断する必要があります。

解決策：結果コードはアルファベット順に並べ替えられます。
その解決策は目標を達成しているか？
A. はい
B. いいえ
正解**: (**正解を表示します**)**
正しい：
* このクエリは、一意のresultCode値ごとに1行を返し、各グループのリクエスト数を示します。
Kustoクエリ言語（KQL）のクエリは、グループ化と集計のメカニズムとして機能するsummarize演算子を使用します。
リクエスト数 = count() を結果コード別に集計します
これは、requestsテーブル内のすべての行を、それぞれの固有のresultCodeに基づいてグループ化します。次に、各グループ内のログの総数をカウントし、その値をrequest_countという新しい列に格納します。
order by request_count desc: これは、集計された行を、リクエスト数が最も多い resultCode が一番上に表示されるように並べ替えます。
正しくない：
* 結果コードはアルファベット順に並べられています。
* このクエリは、個々のリクエストとその結果コードを一覧表示します。
参照：
https://learn.microsoft.com/en-us/azure/azure-monitor/reference/tables/apprequests

質問: 28
ケーススタディ2 - Proseware Inc.
背景
Proseware Inc.は、企業顧客向けにAIを活用した知識管理ソリューションを開発しています。
同社は、セマンティック検索、インテリジェントな文書検索、およびリアルタイムのパートナー連携をサポートするために、プラットフォームの近代化を進めている。
エンジニアリングチームはPythonとAzure SDKを使用しています。アーキテクチャは、コンテナ化されたマイクロサービス、ベクトル検索ワークロード、およびサーバーレスバックエンド処理をサポートするように再設計されています。
計画されたアプリケーションアーキテクチャ
マイクロサービスはDockerを使用してコンテナ化されます。
コンテナ化されたマイクロサービスとAzure Functionsアプリのコードはローカルで開発されますが、GitHubリポジトリに保存されます。
コンテナ化されたマイクロサービス用のカスタムイメージは、Azure Container Registry (ACR) に保存されます。
ベースイメージはDocker Hubに保存されます。カスタムイメージは、ベースイメージが更新されるたびに自動的に再構築される必要があります。
Azure Cosmos DB for NoSQL は、ドキュメント、メタデータ、およびベクトル埋め込みを保存します。
Azure Functions は、NoSQL でホストされているドキュメントの Azure Cosmos DB のベクトル埋め込みを生成し、Service Bus にメッセージを送信して検索インデックスの更新をトリガーします。
Azure Container Apps (ACA) アプリは、NoSQL ドキュメント用の Azure Cosmos DB 全体にわたるセマンティック検索を提供するバックエンド API サービスをホストします。API サービスは、Service Bus メッセージを処理し、検索インデックスを更新します。
Azure Kubernetes Service (AKS) は、既存の Azure Cosmos DB for NoSQL ドキュメントに対して、ベクトル埋め込みの再生成をバッチ処理します (埋め込みモデルが変更されるたびに)。
エクストラネットに接続されたコンテナ化されたWebhookにより、ビジネスパートナーは文書を送信して、内部のAIワークフローで処理し、意味検索や取得を行うことができる。
監視
Azure リソースによって生成されたテレメトリは、Azure Monitor に送信されます。
Log Analyticsワークスペースは、ACAアプリのログ、AKSコンテナのログ、およびAzure Functionsアプリのログを収集するために使用されます。
Azure Functionsの監視は現在、Azure Application Insights SDKの計測機能を使用して実装されています。
ビジネス要件

新規または更新されたAzure Cosmos DB for NoSQLでホストされるドキュメントの埋め込みは、自動的に生成される必要があります。
バックエンドAPIサービスは、営業時間中に自動的にスケールする必要があります。
バックエンドAPIのコールドスタート時の遅延は最小限に抑える必要がある。
機密情報はコンテナイメージの外に保存する必要があります。
開発者は、Azure Functions ホストとアプリ間でテレメトリを関連付けることができなければなりません。
すべてのトレースは、OpenTelemetry SDKの計測機能を使用して実装する必要があります。
開発努力は最小限に抑えなければならない。
技術要件
コンテナイメージは、コードの更新がメインブランチにマージされる前に、自動的に構築され、検証される必要があります。
イメージのビルド自動化はAzure Container Registry内で実行される必要があり、ローカルの開発者マシンや外部のビルドサービスへの依存を排除する必要があります。
イメージビルドが開発者のローカルマシンに依存する状態は解消されなければならない。
ACAにおけるイベント駆動型スケーリングは、Azure Service Busキュー内の保留中のメッセージ数に基づいて行われる必要があります。
Azure Cosmos DB for NoSQL の RU 消費量を最小限に抑える必要があります。
ベクトル類似性検索では、Azure Cosmos DB for NoSQLに保存されている埋め込みデータを使用する必要があります。
パートナー向けのコンテナ化されたWebhookサービスは、Azure App Service上で実行する必要があります。
機密情報は、コンテナイメージ、ソースコード管理システム、またはアプリケーション構成に直接保存してはなりません。実行時に安全にアクセスできる必要があります。
すべてのシークレットはAzure Key Vaultに一元的に保存され、実行時にはマネージドIDを通じてアクセスされる必要があります。
Azure App Service は、外部サービスに依存することなく、実行時にシークレットを提供する必要があります。
リソースとワークロードは、自動化されたバージョン管理パイプラインを通じて、Bicepテンプレートを使用してデプロイする必要があります。再現性と監査可能性を確保するため、ローカルおよびコマンドラインによるデプロイは排除する必要があります。
既知の問題
ベクトル類似性クエリ実行中にRU消費量が急増する。
ビジネス要件と技術要件を満たすために、Azure Functionsのリソースとアプリをデプロイする必要があります。何を使用すればよいでしょうか？
A. GitHub Actions
B. Azure Functions コアツール
C. Azure CLI
D. ローカルGitデプロイメント
正解**: A (**コメントを発表する**)**
GitHub Actionsは、使用すべき適切なツールです。
自動化パイプライン :バージョン管理されたワークフローを通じて、インフラストラクチャのデプロイ（Bicep）とコードのコンパイルをネイティブに自動化します。
ローカルデプロイメントを排除 :ワークロードはリポジトリイベント（ロードのマージなど）によってトリガーされるため、手動によるコマンドライン操作が不要になります。
監査可能性と再現性 :すべてのデプロイメントのログを一元的に保持し、ガバナンスのための明確な監査証跡を保証します。
シナリオ：
技術要件
リソースとワークロードは、自動化されたバージョン管理パイプラインを通じて、Bicepテンプレートを使用してデプロイする必要があります。再現性と監査可能性を確保するため、ローカルおよびコマンドラインによるデプロイは排除する必要があります。
ビジネス要件
開発努力は最小限に抑えなければならない。
参照：
https://github.com/marketplace/actions/azure-functions-action

質問: **29**
Redisキャッシュ内のAI生成サマリーに有効期限要件を実装しています。
AIサマリーは、アクセス頻度に関わらず、作成後5分以内に削除する必要があります。
あなたはどうすべきでしょうか？
A. maxmemory-policyをallkeys-lruに設定します
B. 各キャッシュキーに絶対有効期限を設定する
C. 読み取り操作が成功するたびにPERSISTコマンドを使用してください。
D. 各キャッシュキーにスライディング有効期限を実装する
正解: **(**正解を表示します**)**
サマリー作成時に、各Redisキーに固定の有効期限タイムアウト（例：5分）を設定します。Azure Managed Redisは、設定されたタイムアウトが経過すると、キーを自動的に削除します。
重要な点として、通常の読み取り操作では有効期限は延長されないため、時刻Tに作成されたサマリーは、クライアントがどれだけ頻繁に取得しても、約5分後に期限切れとなります。
これは必須の絶対有効期限動作です。Redisでは、アプリケーションは値の書き込み時、または作成後に有効期限コマンドを使用することで有効期限を設定できます。重要な設計上の特性は、キーが読み取られるたびに有効期限が更新されないことです。
allkeys-lru の削除ポリシーは、メモリ負荷に対処するものであり、5 分という確定的な有効期限を設定するものではありません。メモリが不足している場合、使用頻度の低いキーを削除する可能性がありますが、作成後 5 分以内に削除されることを保証するものではありません。PERSIST コマンドは、既存のタイムアウトを削除してしまうため、サマリーが永久に残ってしまう可能性があり、明らかに不適切です。スライディング有効期限も、アクティビティに基づいて有効期限ウィンドウを更新またはリセットするため、頻繁にアクセスされるサマリーが 5 分以上残ってしまう可能性があるため、不適切です。
したがって、鍵ごとに絶対的な5分間の有効期限を設定するのが正しい構成です。
学習ガイドの参照: Azure Managed Redis # キーの有効期限; TTL の動作; キャッシュの有効期限と削除; 固定有効期限とスライディング有効期限。

質問: **30**
複数の Azure Service Bus キューを使用するソリューションを開発しています。Azure Service Bus 名前空間用の Azure Event Grid サブスクリプションを作成します。メッセージの処理には、サブスクライバーとして Azure Functions を使用します。
キューからAzure Event Gridへイベントが送信できることを確認する必要があります。ソリューションは最小権限の原則に従い、コストを最小限に抑える必要があります。
どの Azure Service Bus ロールとティアレベルを使用すべきですか？回答するには、回答領域で適切なオプションを選択してください。
注：正解ごとに1ポイントが加算されます。
Answer Area



Microsoft
ipnshiken.com

正解:



Explanation:



検証済みの回答 :ティア :プレミアム。役割 :Azure Service Bus データレシーバー。

詳細説明 :Microsoft のドキュメントによると、Service Bus キューとトピックの Event Grid 統合には Service Bus Premium 名前空間が必要です。Azure Function サブスクライバーはキューからメッセージを受信する必要があります。この目的のための最小限の権限を持つ組み込みデータプレーンロールは Azure Service Bus Data Receiver です。Data Sender では受信が許可されず、Data Owner は必要以上に権限が広すぎます。したがって、Premium は機能の前提条件を満たし、Data Receiver は明示された最小限の権限の要件を満たします。

学習ガイドの関連性 :Azureサービス統合 :Service Bus、Event Grid、Azure Functionsトリガーバインディング、およびイベント駆動型処理。

Microsoft Learn 公式資料: AI-200 学習ガイド | Service Bus と Event Grid の統合 | Microsoft Entra ID を使用した Service Bus の認証

質問: **31**

あなたは、ある企業からの注文リクエストを処理するAzure Functionsアプリをレビューしています。この関数は以下の要件を満たす必要があります。

* 外部クライアントアプリケーションからの注文を受け付ける。
* セキュリティのためにアクセス制限が必要

処理済みのリクエストペイロードを含むレスポンスを返します。



以下の各記述について、正しい場合は「はい」を選択してください。そうでない場合は「いいえ」を選択してください。

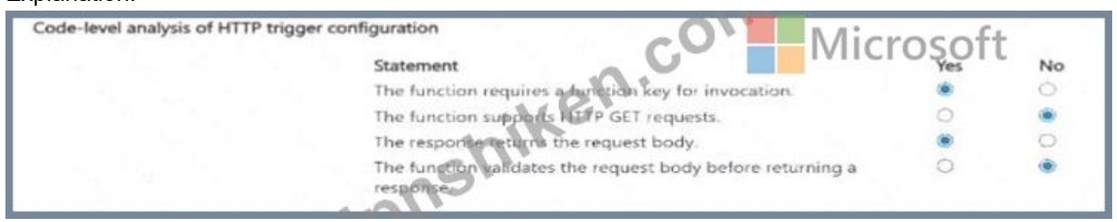
注：正解ごとに1ポイントが加算されます。



正解:



Explanation:



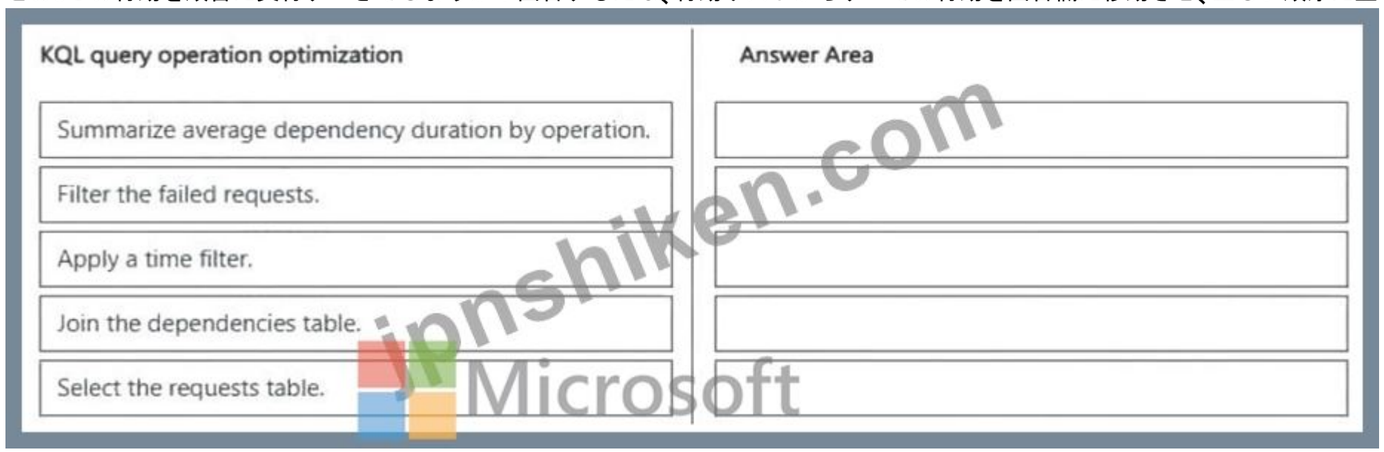
検証済みの回答 :はい、いいえ、はい、いいえ。
詳細な説明: `auth_level=FUNCTION` は、より強力なプラットフォーム認証レイヤーが設定されていない限り、呼び出しにファンクションキーが必要であることを意味するため、最初のステートメントは真です。ルートは明示的に POST を許可しており、GET は許可していません。関数は `req.get_body()` を読み取り、その値を HttpResponse に返すため、レスポンスにはリクエストボディが含まれます。レスポンスが構築される前に、スキーマ、タイプ、必須フィールド、またはその他のペイロード検証は実行されないため、最後のステートメントは偽となります。
学習ガイドの関連性 :Azureサービス統合 :Service Bus、Event Grid、Azure Functionsトリガー
バインディング、およびイベント駆動型処理。
Microsoft Learn 公式資料: AI-200 学習ガイド | Azure Functions HTTP トリガー

有効的な**AI-200**問題集はJPNTest.com提供され、**AI-200**試験に合格することに役に立ちます！JPNTest.comは今最新**AI-200**試験問題集を提供します。JPNTest.com AI-200試験問題集はもう更新されました。ここで**AI-200**問題集のテストエンジンを手に入れます。最新版のアクセス、https://www.jpntest.com/shiken/AI-200-mondaishu 「**144**問、３０%ディスカウント、特別な割引コード: **JPNshiken**」

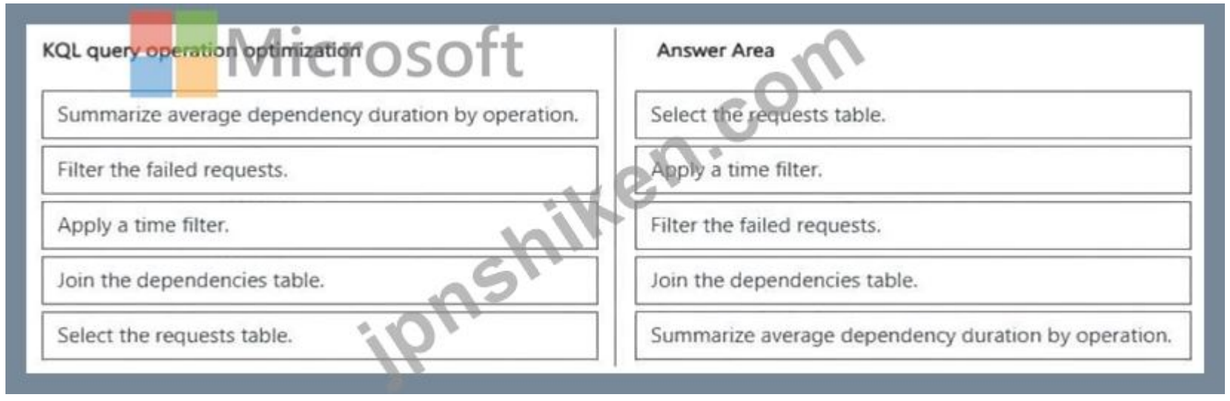
質問: **32**
あなたはAzureにデプロイするJavaアプリケーションを開発しています。このアプリケーションは、機密データをAzure Cosmos DBに保存します。
アプリケーション内の機密データを暗号化するには、Always Encryptedを設定する必要があります。
まず最初に何をすべきでしょうか？
A. 顧客管理キー (CMK) を作成し、そのキーを新しい Azure Key Vault インスタンスに保存します。
B. Azure Cosmos DB SDK を使用してデータ暗号化キー (DEK) を作成し、そのキーを Azure Cosmos DB に保存します。

C. Microsoft Entra ID マネージド ID を作成し、その ID を新しい Azure Key Vault インスタンスに割り当てます。
D. 暗号化する JSON プロパティを含む暗号化ポリシーを組み込むための新しいコンテナを作成します。
正解: **(**正解を表示します**)**
Azure Cosmos DB を使用する Java アプリケーションで Always Encrypted (クライアント側暗号化) を構成するには、まず顧客管理キー (CMK) を作成し、新しい Azure Key Vault インスタンスに保存する必要があります。参照:
https://lennilobel.wordpress.com/2022/08/20/configuring-client-side-encryption-aka-always-encrypted-in-azure-cosmos-db/

質問: **33**
ドラッグアンドドロップ問題
あなたは、1日に数百万件のリクエストを処理するAI検索アプリケーションにおける高遅延について調査しています。テレメトリデータはAzure Monitor Logsに保存されます。
AppRequestsテーブルとAppDependenciesテーブルの情報を関連付けるKQLクエリを作成する必要があります。クエリは以下の要件を満たす必要があります。
過去24時間分のデータのみを含める。
- 失敗したリクエストのみをフィルタリングします。
- 操作ごとにグループ化された依存関係の平均期間を計算します。
クエリは、初期データスキャンを最小限に抑えることでパフォーマンスを最適化する必要があります。
クエリを作成する必要があります。
どの5つの行動を順番に実行すべきでしょうか？回答するには、行動リストからすべての行動を回答欄に移動させ、正しい順序に並べ替えてください。



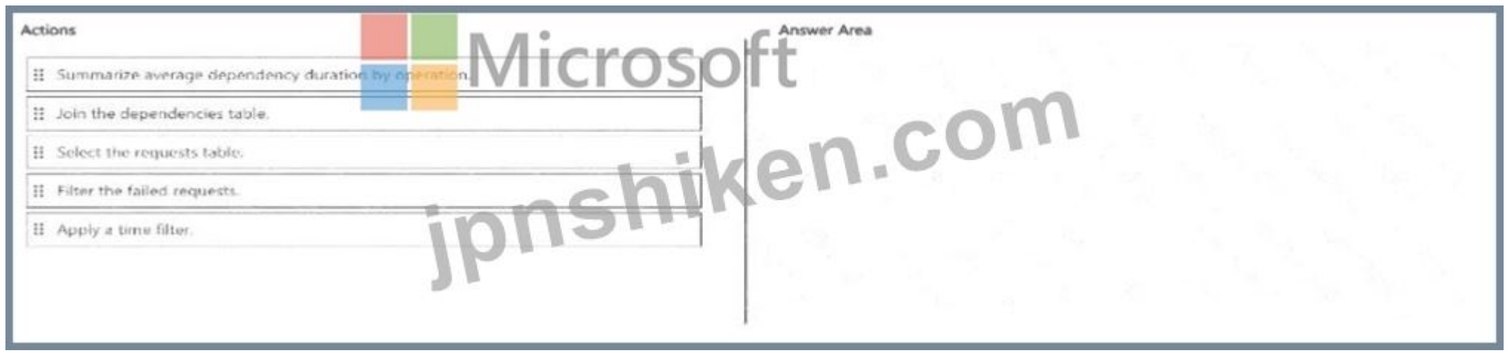
正解:



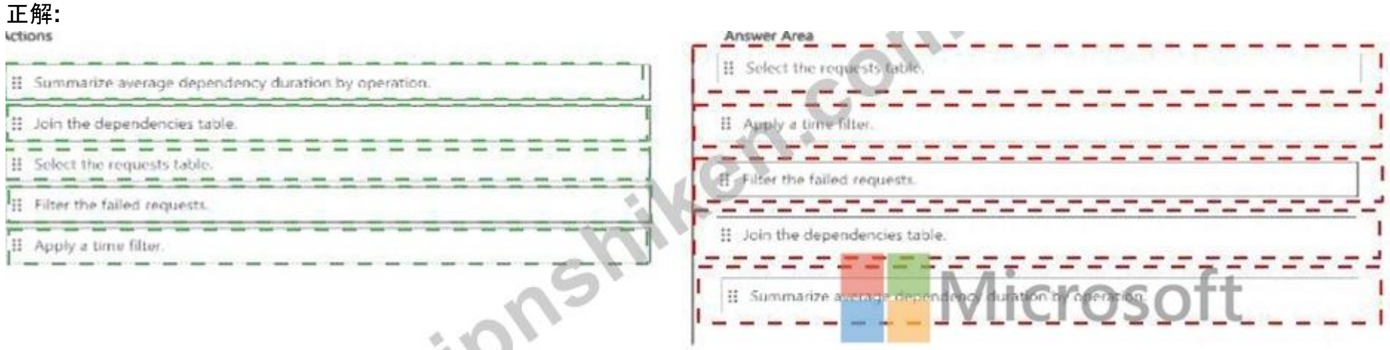
Microsoft
jpnshiken.com

Explanation:
ステップ1 :リクエストテーブルを選択します
プライマリベーステーブルからクエリストリームを開始します。
ステップ2：時間フィルターを適用する
例えば タイムスタンプが1日以上前の場合）
スキャンサイズを縮小してパフォーマンスを最適化するため、データセットを直近24時間分に限定します。
ステップ3：失敗したリクエストをフィルタリングする
success == false または resultCode チェックの場合
リソースを大量に消費する処理を実行する前に、失敗した行のみに絞り込みます。
ステップ4：依存関係テーブルを結合する
(join AppDependencies): 共通の追跡識別子を使用して、フィルタリングされたリクエストのサブセットを依存関係データと関連付けます。
ステップ 5: 操作ごとの平均依存関係期間を要約します (操作ごとに avg(duration) を要約) 操作名ごとにグループ化された最終的な集計メトリックを計算します。
参照：
https://learn.microsoft.com/en-us/azure/azure-monitor/logs/get-started-queries

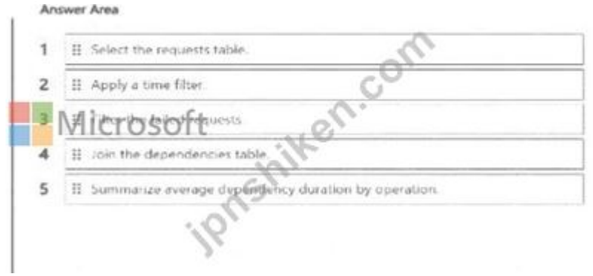
質問**: 34**
あなたは、1日に数百万件のリクエストを処理するAI検索アプリケーションにおける高遅延について調査しています。
テレメトリデータはAzure Monitor Logsに保存されます。
AppRequestsテーブルとAppDependenciesテーブルの情報を関連付けるKQLクエリを作成する必要があります。クエリは以下の要件を満たす必要があります。
* 過去24時間分のデータのみを含めてください。
* 失敗したリクエストのみをフィルタリングします。
* 操作ごとにグループ化された依存関係の平均期間を計算します。
クエリは、初期データスキャンを最小限に抑えることでパフォーマンスを最適化するためのベストプラクティスに従う必要があります。
クエリを作成する必要があります。
どのような順序で操作を実行すればよいでしょうか？回答するには、操作リストからすべての操作を回答欄に移動させ、正しい順序に並べ替えてください。



正解:



Explanation:



検証済みの回答: 1) リクエスト テーブルを選択します。2) 24 時間の時間フィルターを適用します。3) 失敗したリクエストをフィルタリングします。4) 依存関係テーブルを結合します。5) 操作ごとに平均依存関係期間を集計します。

詳細説明 :KQLは、特にスキャンするデータ量を制限する時間述語など、選択性の高いフィルタをできるだけ早い段階で適用すると、最高のパフォーマンスを発揮します。AppRequestsから開始し、過去24時間に限定し、さらに失敗をフィルタリングすることで、結合前に左側のデータセットを削減できます。その後、依存関係テーブルがこれらのリクエストと関連付けられ、最後に集計を実行して操作ごとの平均依存期間を計算します。時間フィルタと失敗フィルタの前に結合または集計を行うと、必要以上に多くのデータが処理されることになります。

学習ガイドの関連性 :セキュリティと運用 :Key Vault、アプリ構成、マネージドID、OpenTelemetry、Azure Monitor、およびKQLベースのトラブルシューティング。

Microsoft Learn 公式資料: AI-200 学習ガイド | Azure Monitor でログクエリを最適化する

質問: **35**

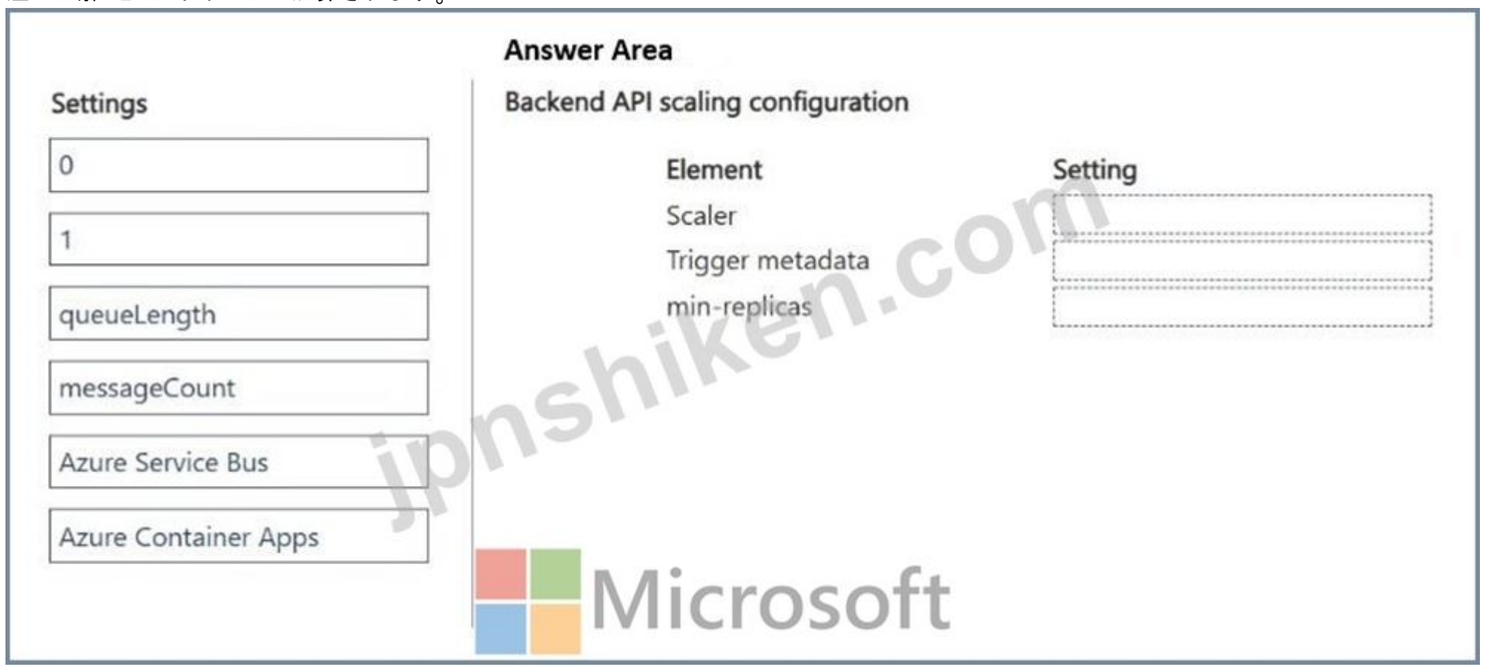
ケーススタディ2 - Proseware Inc.

背景

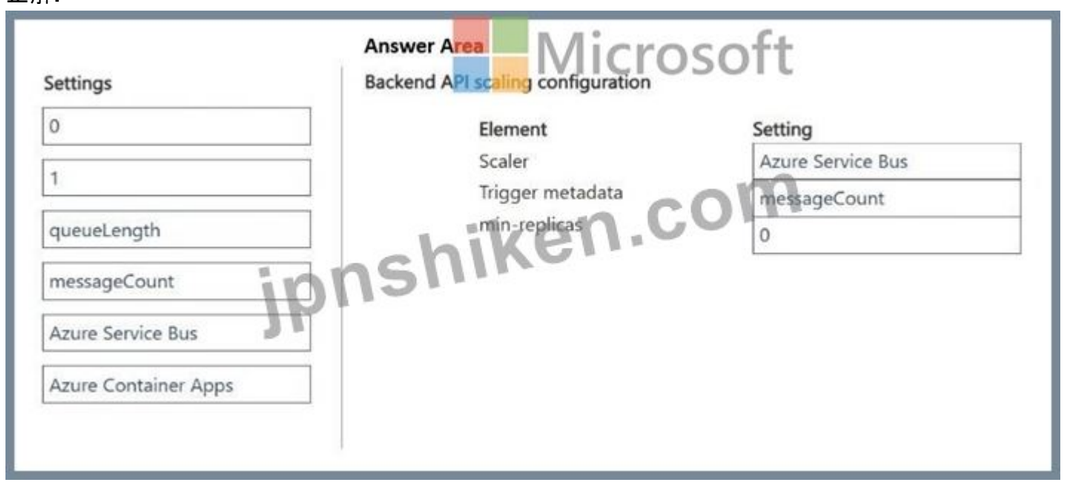
Proseware Inc.は、企業顧客向けにAIを活用した知識管理ソリューションを開発しています。

同社は、セマンティック検索、インテリジェントな文書検索、およびリアルタイムのパートナー連携をサポートするために、プラットフォームの近代化を進めている。
エンジニアリングチームはPythonとAzure SDKを使用しています。アーキテクチャは、コンテナ化されたマイクロサービス、ベクトル検索ワークロード、およびサーバーレスバックエンド処理をサポートするように再設計されています。
計画されたアプリケーションアーキテクチャ
マイクロサービスはDockerを使用してコンテナ化されます。
コンテナ化されたマイクロサービスとAzure Functionsアプリのコードはローカルで開発されますが、GitHubリポジトリに保存されます。
コンテナ化されたマイクロサービス用のカスタムイメージは、Azure Container Registry (ACR) に保存されます。
ベースイメージはDocker Hubに保存されます。カスタムイメージは、ベースイメージが更新されるたびに自動的に再構築される必要があります。
Azure Cosmos DB for NoSQL は、ドキュメント、メタデータ、およびベクトル埋め込みを保存します。
Azure Functions は、NoSQL でホストされているドキュメントの Azure Cosmos DB のベクトル埋め込みを生成し、Service Bus にメッセージを送信して検索インデックスの更新をトリガーします。
Azure Container Apps (ACA) アプリは、NoSQL ドキュメント用の Azure Cosmos DB 全体にわたるセマンティック検索を提供するバックエンド API サービスをホストします。API サービスは、Service Bus メッセージを処理し、検索インデックスを更新します。
Azure Kubernetes Service (AKS) は、既存の Azure Cosmos DB for NoSQL ドキュメントに対して、ベクトル埋め込みの再生成をバッチ処理します (埋め込みモデルが変更されるたびに)。
エクストラネットに接続されたコンテナ化されたWebhookにより、ビジネスパートナーは文書を送信して、内部のAIワークフローで処理し、意味検索や取得を行うことができる。
監視
Azure リソースによって生成されたテレメトリは、Azure Monitor に送信されます。
Log Analyticsワークスペースは、ACAアプリのログ、AKSコンテナのログ、およびAzure Functionsアプリのログを収集するために使用されます。
Azure Functionsの監視は現在、Azure Application Insights SDKの計測機能を使用して実装されています。
ビジネス要件
新規または更新されたAzure Cosmos DB for NoSQLでホストされるドキュメントの埋め込みは、自動的に生成される必要があります。
バックエンドAPIサービスは、営業時間中に自動的にスケールする必要があります。
バックエンドAPIのコールドスタート時の遅延は最小限に抑える必要がある。
機密情報はコンテナイメージの外に保存する必要があります。
開発者は、Azure Functions ホストとアプリ間でテレメトリを関連付けることができなければなりません。
すべてのトレースは、OpenTelemetry SDKの計測機能を使用して実装する必要があります。
開発努力は最小限に抑えなければならない。
技術要件
コンテナイメージは、コードの更新がメインブランチにマージされる前に、自動的に構築され、検証される必要があります。
イメージのビルド自動化はAzure Container Registry内で実行される必要があり、ローカルの開発者マシンや外部のビルドサービスへの依存を排除する必要があります。
イメージビルドが開発者のローカルマシンに依存する状態は解消されなければならない。
ACAにおけるイベント駆動型スケーリングは、Azure Service Busキュー内の保留中のメッセージ数に基づいて行われる必要があります。
Azure Cosmos DB for NoSQL の RU 消費量を最小限に抑える必要があります。
ベクトル類似性検索では、Azure Cosmos DB for NoSQLに保存されている埋め込みデータを使用する必要があります。
パートナー向けのコンテナ化されたWebhookサービスは、Azure App Service上で実行する必要があります。
機密情報は、コンテナイメージ、ソースコード管理システム、またはアプリケーション構成に直接保存してはなりません。実行時に安全にアクセスできる必要があります。
すべてのシークレットはAzure Key Vaultに一元的に保存され、実行時にはマネージドIDを通じてアクセスされる必要があります。
Azure App Service は、外部サービスに依存することなく、実行時にシークレットを提供する必要があります。
リソースとワークロードは、自動化されたバージョン管理パイプラインを通じて、Bicepテンプレートを使用してデプロイする必要があります。再現性と監査可能性を確保するため、ローカルおよびコマンドラインによるデプロイは排除する必要があります。

既知の問題
ベクトル類似性クエリ実行中にRU消費量が急増する。
ドラッグアンドドロップ問題
技術要件を満たすためには、バックエンドAPIサービスに対してイベント駆動型スケーリングを設定する必要があります。
各要素にはどの設定を使用すればよいでしょうか？回答するには、適切な設定を正しい要素に移動してください。各設定は、1回だけ使用することも、複数回使用することも、まったく使用しないこともできます。コンテンツを表示するには、ペイン間の分割バーを移動したり、スクロールしたりする必要がある場合があります。
注：正解ごとに1ポイントが加算されます。



正解:



Explanation:

シナリオ、技術要件。
ACAにおけるイベント駆動型スケーリングは、Azure Service Busキュー内の保留中のメッセージ数に基づいて行われる必要があります。
ボックス1 :Azure Service Bus
スケーラータイプ
スケールルールタイプをカスタムに設定します。
KEDA スケーラー名: カスタム仕様ブロック内のタイプを azure-service-bus に設定します。
ボックス2 :メッセージ数
トリガーメタデータ値
メタデータブロックは、KEDAが必要なレプリカ数を計算する方法を決定します。
以下のコアメタデータパラメータを設定します。
queueName: 対象のAzure Service Busの明示的な文字列名
*-> queue.messageCount: レプリカごとに割り当てられる同時保留メッセージの目標整数しきい値 (例: 5 または 10)。KEDA はこの値を使用して体系的にスケールアウトします。
activationMessageCount: 1 または 0 に設定します。これは、コンテナ アプリを休止状態から復帰させるために必要な正確なメトリックの最小値を定義します。
namespace: Service Bus名前空間の正規名
ボックス3 :0
最小レプリカ構成値
--min-replicas を 0 に設定してください。
この値をゼロに設定すると、サーバーレスのコスト効率が最大限に高まります。KEDAはバックグラウンドでのポーリングを積極的に処理するため、Service Busキューに保留中のメッセージがゼロの場合、バックエンドAPIコンテナは完全にスケールダウンし、アクティブなレプリカをゼロで実行できます。
参照 :
https://oneuptime.com/blog/post/2026-02-16-how-to-set-up-keda-based-auto-scaling-with-queue-triggers-in-azure-container-apps/view

質問: **36**
Azure Cosmos DB for NoSQL 内で高いデータ取り込みレートに対応するため、変更フィードプロセッサの複数のインスタンスをデプロイします。
各プロセッサインスタンスは、異なるパーティションのサブセットを処理する必要がある。
ワークロードがすべてのプロセッサインスタンスに分散されるようにする必要があります。
何を設定すればよいですか？
A. インデックス精度
B. 高い一貫性
C. コンテナのリース
D. オートスケールスループット
正解: **(**正解を表示します**)**
Azure Cosmos DB for NoSQL で高いデータ取り込みレートに対応するには、同じワークフロー名と同一のリースコンテナーを使用して、変更フィードプロセッサの複数のインスタンスをデプロイする必要があります。
参照 :
https://github.com/Azure/azure-cosmos-dotnet-v3/wiki/Comprehensive-Test-Suite-for-Azure-Cosmos-DB-Change-Feed-Processor:-All-Versions-and-Deletes

質問: **37**
セマンティック検索アプリケーションは、Azure Database for PostgreSQL にクエリを実行し、ドキュメントの埋め込みとメタデータを次の列を持つテーブルに格納します。
* 埋め込み (pgvector)
* 部門

* 作成日時
アプリケーションは、指定されたクエリに対して、財務部門の文書のみを対象に、最も類似性の高い上位5件の文書を返す必要があります。メタデータフィルタリングを用いたセマンティック検索を実装してください。
どのクエリコンポーネントを選択すればよいでしょうか？回答するには、適切なクエリコンポーネントを正しい要件に移動してください。各クエリコンポーネントは、1回、複数回、またはまったく使用しない場合があります。コンテンツを表示するには、ペイン間の分割バーを移動したり、スクロールしたりする必要がある場合があります。



正解:



Explanation:

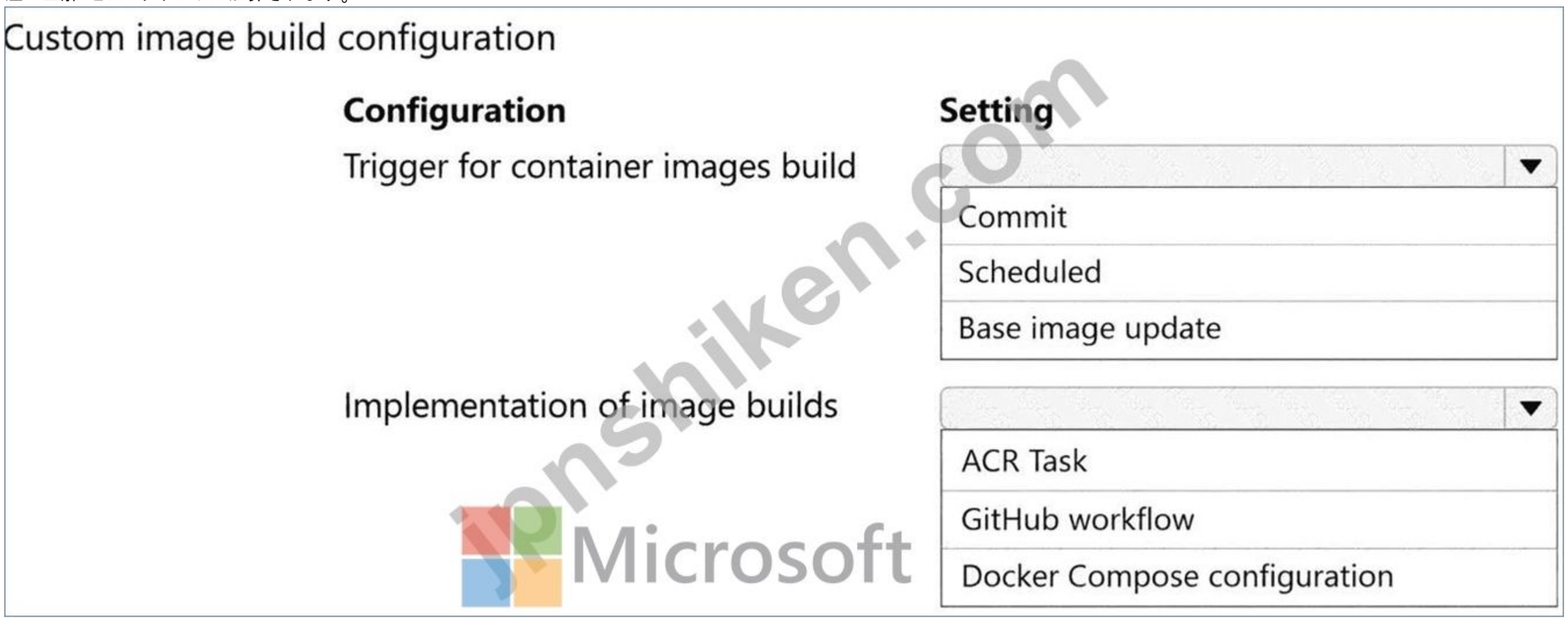


検証済みの回答: フィルター: `WHERE department = ' finance ' `。ランク/上位 5 件: `ORDER BY embedding < = > query_embedding LIMIT 5`。
詳細説明 :メタデータ述語は候補行を財務部門に限定する必要があるため、`WHERE department = ' finance ' ` コンポーネントが必要なフィルタを提供します。pgvector のコサイン距離演算子 ` < = > ` は、指定されたクエリ埋め込みに対するベクトル距離に基づいて行を並べ替え、`LIMIT 5` は最も近い 5 つの一致のみを保持します。作成日で並べ替えると、意味的な類似性よりも最新性が優先され、ワイルドカード部門フィルタでは財務部門のみという要件を満たせません。
学習ガイドの関連性 :AIデータ管理ワークロード :Cosmos DB、PostgreSQL、キャッシング、ベクトルストレージ、ベクトル検索、一貫性、および接続最適化。
Microsoft Learn 公式資料: AI-200 学習ガイド | Azure PostgreSQL を使用したベクトル類似性検索

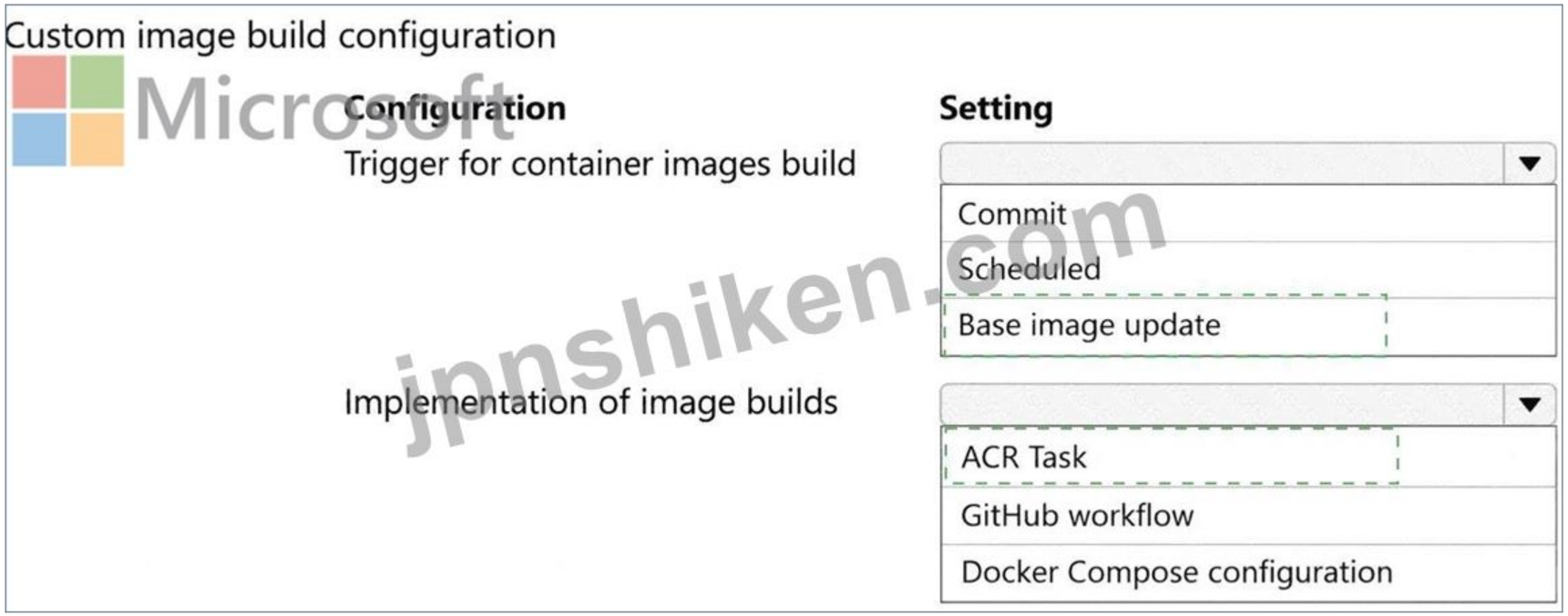
質問: **38**
あなたは、公開APIを提供するAzure Functionsアプリを設計しています。
解決策は以下の条件を満たす必要があります。
- 受信したリクエストデータを検証し、結果をすぐに返します。
発信者。
- Microsoft Entra ID認証をサポートします。
- 同じリクエストが再試行された場合に、冪等な処理を保証します。
・負荷変動に応じて自動的にスケーリングします。
重複処理を避ける。
トリガーを実装する必要があります。

どのトリガーを実装すべきでしょうか？
A. サービスバスに関する話題
B. HTTP
C. Azure Event Grid
D. Azureキューストレージ
正解: **B** (コメントを発表する)
HTTPトリガーとAzure Durable Functions（非同期HTTP開始パターン）、そしてバックエンドデータベースを組み合わせたアーキテクチャは、お客様のすべての要件を満たすのに最適な構成です。
即時検証：HTTPトリガーを使用すると、リクエストとレスポンスの同期通信が可能になり、ペイロードデータを検証して、呼び出し元に200 OKまたは400 Bad Requestを即座に返すことができます。
Entra ID認証：Azure FunctionsネイティブのApp Service認証（Easy Auth）はHTTPエンドポイントと直接統合され、コードが実行される前にMicrosoft Entra IDトークンの検証を強制します。
自動スケーリング：HTTP トリガー関数は、受信する Web トラフィックに基づいて、Azure Functions の従量課金プランまたはプレミアムホスティングプランを使用して、すぐに動的にスケーリングされます。
参照：
https://github.com/microsoft/woodgrove-auth-api/blob/main/README.md

質問: **39**
技術要件に基づいて、イメージ構築の自動化を設定する必要があります。
どの設定を構成すればよいですか？回答するには、回答欄で適切なオプションを選択してください。
注：正解ごとに1ポイントが加算されます。



正解:



Microsoft
jpnshiken.com

Explanation:

* コンテナイメージ構築のトリガー: ベースイメージの更新

* イメージビルドの実装: ACRタスク

適切なトリガーはベースイメージの更新です。Prosewareでは、Docker Hubのベースイメージが変更されるたびに、すべてのカスタムコンテナイメージが自動的に再構築される必要があるためです。Azure Container Registry タスクは、DockerfileのFROM命令からベースイメージの依存関係を検出し、そのベースイメージが更新されたときに自動的に再構築をトリガーできます。Microsoftは、ACRタスクがDocker HubやMicrosoft Container Registry などのパブリックリポジトリにあるベースイメージ、およびAzure Container Registryに保存されているイメージを追跡できることを確認しています。

ビルドの実装は ACR タスクである必要があります。これにより、イメージビルドの自動化が Azure Container Registry 内で実行されるという技術要件が直接満たされ、ローカルの開発者マシンや個別の外部ビルドサービスへの依存がなくなります。ACR タスクはクラウドベースのコンテナ構築を提供し、ソースコードのコミット、ベースイメージの更新、またはスケジュールされたトリガーに基づいてビルドを自動化できます。

GitHubワークフローはCI/CDをオーケストレーションできますが、それを実際のイメージ構築実装として使用すると、ビルドをACR内で実行するという明示的な要件と矛盾します。Docker Composeはマルチコンテナアプリケーションを定義するものであり、ネイティブのACRビルド自動化メカニズムではありません。コミットトリガーは、明示されたベースイメージ更新要件ではなくソース変更に対応し、スケジュールされたビルドは不要な定期的な実行を引き起こします。

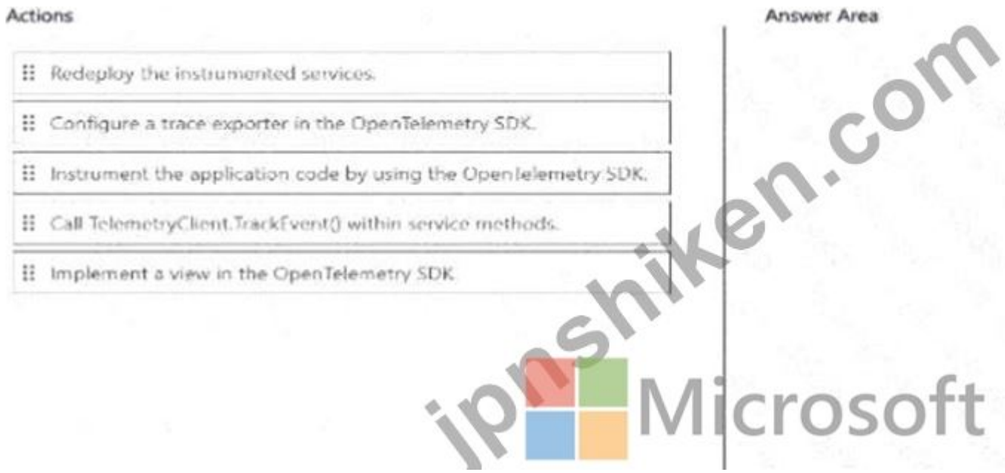
学習ガイドの参照項目：Azure Container Registry タスク、ベースイメージの依存関係追跡、自動再構築、Docker Hub ベースイメージのトリガー、クラウドネイティブコンテナのビルド。

質問: **40**

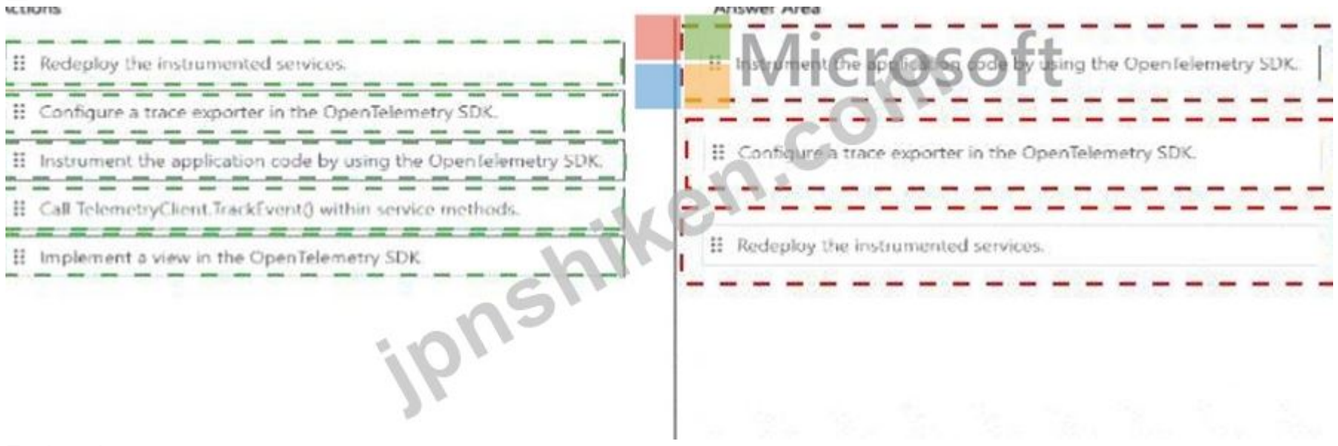
ビジネス要件に従って、トレース相関を実装する必要があります。

どの3つの行動を順番に実行すべきでしょうか？回答するには、行動リストから適切な行動を回答欄に移動させ、正しい順序に並べ替えてください。

注：正解の選択肢は複数あります。正解の選択肢であれば、どの順番を選んでも点数が加算されます。



正解:



Explanation:



検証済みの回答: 1) OpenTelemetry SDK を使用してアプリケーション コードを計測します。2) OpenTelemetry SDK で Azure Monitor トレース エクスポーターを構成します。3) 計測済みのサービスを再デプロイします。
詳細説明 :トレース相関には、アプリケーションが OpenTelemetry スパンを作成し、それらのスパンを Azure Monitor に送信するエクスポーターが必要です。そのため、更新されたサービスがデプロイされる前に、計測機能が実装されている必要があります。以前の Application Insights SDK の計測機能は、すべてのトレースで OpenTelemetry を使用するというケーススタディの要件を満たしていません。OpenTelemetry ビューの構成は、基本的な分散トレースのエクスポートとは無関係であり、TrackEvent の呼び出しは、必要な OpenTelemetry アプローチではなく、Application Insights SDK のパターンです。
学習ガイドの関連性 :セキュリティと運用 :Key Vault、アプリ構成、マネージドID、OpenTelemetry、Azure Monitor、およびKQLベースのトラブルシューティング。
Microsoft Learn 公式資料: AI-200 学習ガイド | Azure Monitor OpenTelemetry を有効にする

質問: 41

あなたはバックエンドワーカーサービスのメッセージ処理コードをレビューしています。
サービスバスプロセッサを初期化して起動する以下のPythonコードを確認してください。

```
from azure.servicebus import ServiceBusClient, ServiceBusReceiveMode

01 import json
02 from azure.servicebus import ServiceBusClient, ServiceBusReceiveMode
03 from azure.identity import DefaultAzureCredential
04 credential = DefaultAzureCredential()
05 with client.get_queue_receiver(
06     queue_name="inference-requests",
```

Service Bus processor behavior and SDK defaults

Statement	Yes	No
Messages are removed from the queue as soon as they are received by the processor.	○	○
If message processing fails due to a transient service error, the message can be retried by another consumer.	○	○
Messages with invalid JSON payloads are retried until the maximum delivery count is reached.	○	○

正解:

Service Bus processor behavior and SDK defaults

Statement	Yes	No
Messages are removed from the queue as soon as they are received by the processor.	○	○
If message processing fails due to a transient service error, the message can be retried by another consumer.	○	○
Messages with invalid JSON payloads are retried until the maximum delivery count is reached.	○	○

Explanation:

Service Bus processor behavior and SDK defaults

Statement	Yes	No
Messages are removed from the queue as soon as they are received by the processor.	○	●
If message processing fails due to a transient service error, the message can be retried by another consumer.	●	○
Messages with invalid JSON payloads are retried until the maximum delivery count is reached.	○	●

検証済みの回答 :いいえ、はい、いいえ。
詳細説明 :Service Bus のデフォルトの受信動作は、受信して削除するのではなく、ピークロック方式です。そのため、メッセージを受信しただけではキューから削除されず、正常に処理される必要があります。処理が失敗してメッセージが未処理のままになったり、ロックが失われたりすると、別の配信に使用できるようになります。ただし、無効な JSON はアプリケーションレベルのコンテンツの問題です。Service Bus は、逆シリアル化が失敗したという理由だけでペイロードを JSON として検査し、MaxDeliveryCount まで自動的に再試行することはありません。何が起こるかは、アプリケーションの処理/エラー処理パスによって決まります。
学習ガイドの関連性 :Azureサービス統合 :Service Bus、Event Grid、Azure Functionsトリガー
バインディング、およびイベント駆動型処理。
Microsoft Learn 公式資料: AI-200 学習ガイド | Service Bus メッセージの転送、ロック、および決済 | Service Bus メッセージ、ルーティング、および相関

質問: **42**

大手小売企業は、オンラインストアと実店舗を運営しています。同社は、全拠点の在庫を効率的に管理するために、在庫レベルをリアルタイムで追跡しています。あなたは、Azureにデプロイされた在庫管理システムによって生成されるイベントを処理するAzure Event Gridソリューションを開発します。
製品の需要における季節的な変化に合わせて動的に調整される購読フィルターを実装する必要があります。
どのイベントフィルターを使用すべきですか？
A. イベントタイプフィールドに、現在のシーズン名に一致するプレフィックスフィルターを適用する。
B. イベントデータ内の季節フィールドを含む複数のデータフィールドを評価するブール条件を使用した高度なフィルター
C. 季節限定のプロモーションコードがタグ付けされたイベントを含めるためにラベルフィルターを使用する購読フィルター
D. 件名が「seasonal/inventory」で終わるイベントを対象とする静的件名フィルター
正解: **(**正解を表示します**)**

質問: **43**
あなたは、AIによる文書処理のためにService Busを使用してメッセージングソリューションを設計しています。
公開したメッセージが複数の独立した消費者に確実に届くようにする必要があります。
各消費者は、メッセージのコピーをそれぞれ受け取る必要がある。
どの2つのService Busエンティティを使用すべきでしょうか？それぞれの正解は、解決策の一部を示しています。
注：正解ごとに1ポイントが加算されます。
A. トピック
B. 定期購読
C. キュー
D. デッドレターキュー
E. メッセージセッション
正解: **(**正解を表示します**)**
トピックとサブスクリプションは、このシナリオに該当するAzure Service Busの2つのエンティティです。
トピック：発行者は、文書処理メッセージを単一のトピックに送信します。このトピックは、中央配信ハブとして機能します。
購読：各ユーザーは、そのトピックに関する個別の購読をそれぞれ作成します。メッセージが届くと、そのコピーがすべての購読者に転送されるため、各ユーザーは自分のコピーを安全かつ個別に処理できます。
参照：
https://learn.microsoft.com/en-us/azure/service-bus-messaging/service-bus-queues-topics-subscriptions

質問: **44**
ホットスポットに関する質問
あなたは、Azure SDK for Pythonを使用して、Key Vaultからデータベース認証情報を取得するAIアプリケーションを開発しています。
アプリケーションは認証にマネージドIDを使用する必要があります。
Key Vaultからシークレットを取得する以下のコードセグメントを確認してください。

```
01  from azure.identity import DefaultAzureCredential
02  from azure.keyvault.secrets import SecretClient
03
04  credential = DefaultAzureCredential()
05  client = SecretClient(
06      vault_url="https://vault.vault.azure.net/",
07      credential=credential
08  )
09
10  secret = client.get_secret("dbPassword", version="123")
```

以下の各記述について、正しい場合は「はい」を選択してください。そうでない場合は「いいえ」を選択してください。
注：正解ごとに1ポイントが加算されます。

Secret retrieval behavior

Statement	Yes	No
The code retrieves the latest version of dbPassword.	○	○
The code authenticates to Key Vault without storing client secrets in the application when deployed to an Azure-hosted environment that has a managed identity enabled.	○	○
If dbPassword is rotated and a new version is created, subsequent calls to this code will retrieve the new version.	○	○

正解:

Secret retrieval behavior

Statement	Yes	No
The code retrieves the latest version of dbPassword.	○	●
The code authenticates to Key Vault without storing client secrets in the application when deployed to an Azure-hosted environment that has a managed identity enabled.	●	○
If dbPassword is rotated and a new version is created, subsequent calls to this code will retrieve the new version.	○	●

Explanation:
ボックス1 :いいえ
いいえ、このコードではdbPasswordの最新バージョンは取得できません。
その代わりに、文字列「123」で指定された特定のバージョンを明示的に取得します。

シークレットの最新のアクティブバージョンを取得するには、get_secret 関数から version パラメータを完全に省略します。python# この行を変更します:
secret = client.get_secret("dbPassword", version="123")
# この行へ:
secret = client.get_secret("dbPassword")
ボックス2 :はい
はい、マネージドIDが有効になっているAzureホスト環境にデプロイした場合、クライアントシークレットを保存することなく、コードはAzure Key Vaultへの認証に成功します。
DefaultAzureCredential の動作: このクラスは、資格情報を順次連結するメカニズムです。
Azure（Azure App Service、Azure VM、Azure Functionsなど）にデプロイされると、アクティブなマネージドID環境エンドポイントを自動的に検索してアクセストークンを取得します。
秘密情報の保存は不要 :プラットフォームは基盤となるAzureホストインフラストラクチャを利用してIDトークンを生成するため、アプリケーションコードや構成設定内にクライアントの秘密情報、パスワード、証明書などをハードコーディングする必要はありません。
ボックス3 :いいえ
いいえ、このコードを後続で呼び出しても新しいバージョンは取得されません。
get_secret メソッドに version="123" を明示的に含めることで、シークレットの特定の不変バージョンを要求していることになります。
参照 :
https://learn.microsoft.com/en-us/azure/key-vault/secrets/quick-create-python

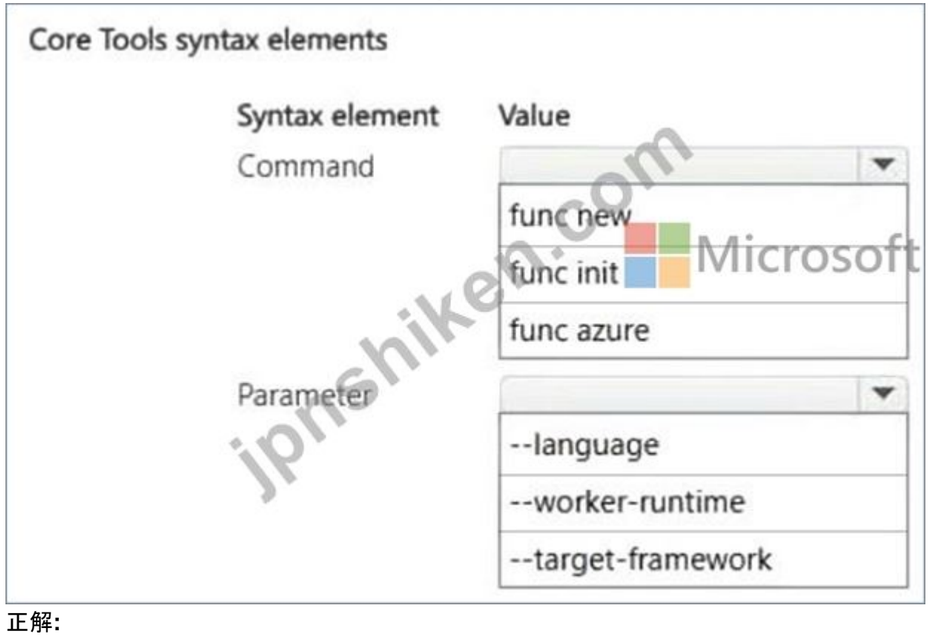
質問: **45**
テキスト読み上げ機能には、標準的なニューラル音声とは異なる、会社のブランドに合ったカスタム音声を追加する必要があります。どのような音声を使用すればよいでしょうか？
A. Azure AI Speech の事前構築済みニューラル音声
B. Azure AI スピーチ カスタム ニューラルボイス
C. Azure AI 言語感情分析
D. Azure AI翻訳機
正解: **(**正解を表示します**)**
カスタムニューラルボイスを使用すると、録音された音声サンプルから独自の合成音声をトレーニングできるため、標準的な既製のニューラル音声を使用するのではなく、ブランド固有の音声を生成できます。

質問: **46**
AIアプリケーションは、アプリケーション構成から構成値を取得します。このアプリケーションは静的構成を使用します。
動的な構成更新をサポートし、アプリケーション要求の遅延を最小限に抑えるソリューションを実装する必要があります。
あなたはどうすべきでしょうか？
A. 環境変数に設定値を注入します。
B. 更新間隔でキャッシュ設定値を設定します。
C. 各リクエストでアプリ構成から構成値を取得します。
D. 設定値をKey Vaultに保存します。
正解: **B (**コメントを発表する**)**
Azure App Configuration プロバイダーのキャッシュおよび動的更新メカニズムを、更新間隔を設定して使用します。アプリケーションは構成をローカル キャッシュに読み込み、定期的に App Configuration の変更を確認します。登録されたキーが変更されると、プロバイダーはアプリケーションの再起動を必要とせずにキャッシュされた構成を更新します。Microsoft は、構成更新チェック間の最小時間を制御するメカニズムとして、SetRefreshInterval および同等のプロバイダー設定について説明しています。
このアプローチは両方の要件を満たします。更新された値をランタイム時に検出して再読み込みできるため、構成は動的になります。また、通常のアプリケーション要求では、毎回リモートのアプリケーション構成要求を行うのではなく、ローカルのキャッシュされたコピーから構成を読み込むため、要求の遅延は低く抑えられます。

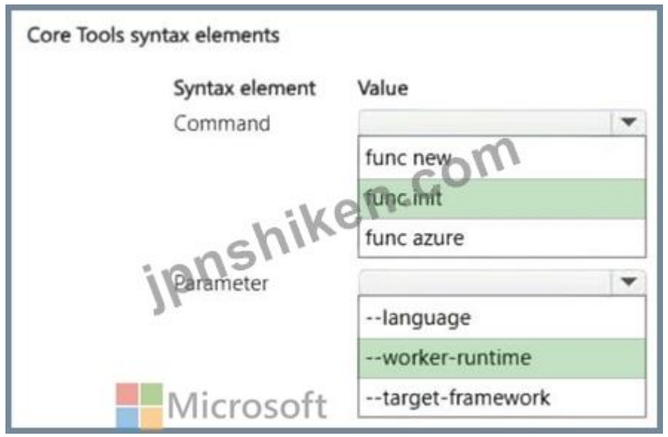
リクエスト駆動型更新の場合、更新は非同期で行われ、受信リクエストをブロックしないことがマイクロソフトによって指摘されています。設定された更新間隔が経過する前に、追加の更新呼び出しが行われても、実質的にリモート操作は実行されません。
マイクロソフトはまた、レイテンシを削減し、過剰なサービス要求、スロットリング、および不要な料金を回避するために、アプリ構成の値を毎回読み込むのではなく、キャッシュすることを推奨しています。
環境変数は、アプリケーションが再起動または再デプロイされない限り静的なままです。リクエストごとに構成を取得すると、レイテンシとサービス負荷が増加します。Key Vaultは主に機密情報を扱うためのものであり、一般的な動的なアプリケーション構成を扱うためのものではありません。
学習ガイドの参照項目 :Azure アプリ構成、動的構成の更新、プロバイダーのキャッシュ、更新間隔、リクエスト駆動型更新、構成の回復力。

有効的な**AI-200**問題集はJPNTest.com提供され、**AI-200**試験に合格することに役に立ちます！JPNTest.comは今最新**AI-200**試験問題集を提供します。JPNTest.com AI-200試験問題集はもう更新されました。ここで**AI-200**問題集のテストエンジンを手に入れます。最新版のアクセス、https://www.jpntest.com/shiken/AI-200-mondaishu　**144**問、３０%ディスカウント、特別な割引コード: **JPNshiken**」

質問: **47**
ホットスポットに関する質問
Azure Functions Core Toolsを使用して、ローカル開発環境にAzure Functionsアプリプロジェクトを作成しています。
テンプレートを使用せずに、PythonまたはC#でプロジェクトを作成する必要があります。
Azure Functions アプリ プロジェクトを作成するために必要なコマンドとそのパラメーターを指定する必要があります。
どのコマンドとパラメータを指定すればよいですか？回答するには、回答欄で適切なオプションを選択してください。
注：正解ごとに1ポイントが加算されます。



正解:



Explanation:

ボックス1 :func init

特定のコード テンプレートを使用せずに Azure Functions Core Tools を使用してコマンドラインから新しい Azure Functions アプリ プロジェクトを初期化するには、func init コマンドと - を組み合わせて使用します。

-worker-runtime パラメータ。

ボックス2 :--worker-runtime

参照 :

https://r-vm.com/python-on-azure-functions

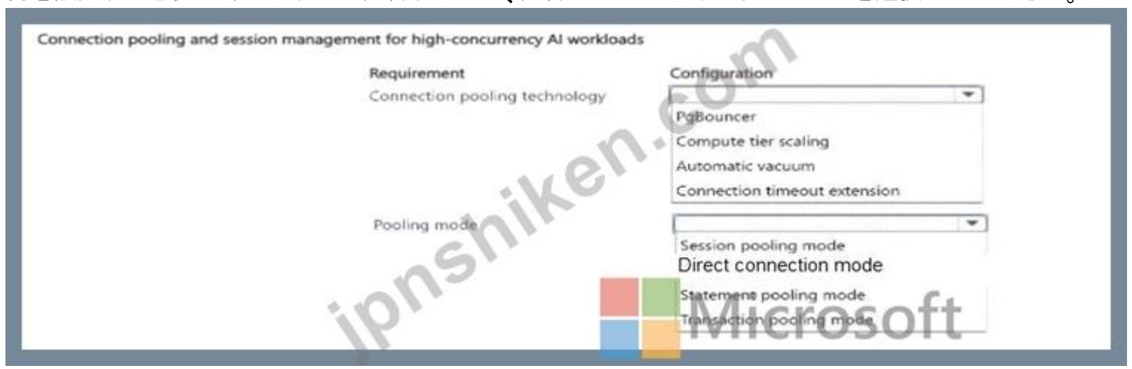
質問: 48

あなたは、Python APIを使用してAzure Database for PostgreSQLに対して類似性クエリを実行するアプリケーションを開発しています。このアプリケーションは、リクエストごとに新しいデータベース接続を作成します。

トラフィックがピークに達すると、アプリケーションが新しいデータベースセッションを開くことに断続的に失敗し、ログには最大接続数に達したことが示されます。

接続設定のオーバーヘッドを削減し、高並行ワークロードにおける再利用を最大化するために、接続プーリング戦略を設定する必要があります。

何を設定する必要がありますか？回答するには、回答エリアで適切なオプションを選択してください。注：正解ごとに1ポイントが加算されます。



正解:



Explanation:



接続プーリング技術 :PgBouncer。プーリングモード :トランザクションプーリングモード。
詳細説明 :Azure Database for PostgreSQL Flexible Server は PgBouncer をサポートしており、接続設定コストを削減し、過剰なクライアント セッションからサーバーを保護します。トランザクション間でサーバー セッションを固定する必要のない高並行 API の場合、トランザクション プーリングは各トランザクション後にサーバー接続をプールに戻すことで再利用を最大化します。セッション プーリングはクライアント セッションの存続期間中サーバー接続を保持するため、前述の接続負荷の高いシナリオでは多重化が少なくなります。
学習ガイドの関連性 :AIデータ管理ワークロード :Cosmos DB、PostgreSQL、キャッシング、ベクトルストレージ、ベクトル検索、一貫性、および接続最適化。
Microsoft Learn 公式資料: AI-200 学習ガイド | Azure Database for PostgreSQL の PgBouncer

質問: **49**
あなたはユーザーサポートシステム用の言語理解モデルをトレーニングしています。
GetContactDetailsという名前の最初のインテントを作成し、200個の例を追加します。
偽陽性の可能性を減らす必要があります。
あなたはどうすべきでしょうか？
A. アクティブラーニングを可能にする。
B. 機械学習エンティティを追加します。
C. GetContactDetails インテントに例を追加します。
D. None インテントに例を追加します。
正解: **(**正解を表示します**)**
意図の誤検出を減らすには、None インテントに代表的な発話を追加します。None インテントは、定義済みのビジネス インテントにマッピングされるべきではないユーザーの発話のために特別に用意されています。Microsoft では、モデルが有効な意図の発話と無関係または曖昧な入力との間のより明確な判断境界を学習できるように、誤検出の可能性のある例を追加することを推奨しています。
例えば、GetContactDetails インテントに 顧客の電話番号を教えてください」といったリクエストが含まれている場合、None インテントには、見た目は似ているものの意味的に関連性のないフレーズを含めるべきです。そうしないと、これらのフレーズが誤って GetContactDetails と分類されてしまう可能性があります。予測時には、発話が None のトレーニング例に似ている場合、または最高インテントスコアが設定された None しきい値を下回る場合に、その発話は None と分類されます。Microsoft は、インテントの識別精度を向上させるために、None インテントに誤検出例を追加することを特に推奨しています。
GetContactDetailsに例を追加するだけでは、一般的にその意図の認識は強化されますが、モデルに十分な否定例を提供することはできません。機械学習エンティティは、意図の分類ではなく、エンティティの抽出に対応します。アクティブラーニングは、レビューのために不確かな発話を特定するのに役立ちますが、要求された直接的な修正措置ではありません。
学習ガイドの参照: Azure AI 言語 # 会話型言語理解; インテント; インテントなし; インテント分類; 誤検出の削減。

質問: **50**
外部API呼び出しに依存するAzure Service Busメッセージを処理します。
APIが一時的に利用できない場合は、配信回数を増やさずにメッセージの処理を遅延させる必要があります。
APIが利用可能なときにメッセージを処理しつつ、メッセージへのアクセス性を維持する方法を見つける必要があります。
どのメッセージ処理を実行すべきですか？
A. デッドレター
B. 延期
C. 放棄する
D. 完了
正解: **(正解を表示します)**
適切なメッセージアクションは「延期（Defer）」です。
メッセージを延期すると、Azure Service Bus はメッセージをメインのアクティブ取得プールから移動し、キューに保持します。重要な点として、この操作によって配信回数が増加したり、メッセージがデッドレターキュー (DLQ) に移動したりすることはありません。メッセージは安全に保存され、アクセス可能な状態のままですが、標準の受信ループでは処理できなくなります。
参照：
https://github.com/Azure/azure-sdk-for-java/issues/43855

質問: **51**
ケーススタディ2 - Proseware Inc.
背景
Proseware Inc.は、企業顧客向けにAIを活用した知識管理ソリューションを開発しています。
同社は、セマンティック検索、インテリジェントな文書検索、およびリアルタイムのパートナー連携をサポートするために、プラットフォームの近代化を進めている。
エンジニアリングチームはPythonとAzure SDKを使用しています。アーキテクチャは、コンテナ化されたマイクロサービス、ベクトル検索ワークロード、およびサーバーレスバックエンド処理をサポートするように再設計されています。
計画されたアプリケーションアーキテクチャ
マイクロサービスはDockerを使用してコンテナ化されます。
コンテナ化されたマイクロサービスとAzure Functionsアプリのコードはローカルで開発されますが、GitHubリポジトリに保存されます。
コンテナ化されたマイクロサービス用のカスタムイメージは、Azure Container Registry (ACR) に保存されます。
ベースイメージはDocker Hubに保存されます。カスタムイメージは、ベースイメージが更新されるたびに自動的に再構築される必要があります。
Azure Cosmos DB for NoSQL は、ドキュメント、メタデータ、およびベクトル埋め込みを保存します。
Azure Functions は、NoSQL でホストされているドキュメントの Azure Cosmos DB のベクトル埋め込みを生成し、Service Bus にメッセージを送信して検索インデックスの更新をトリガーします。
Azure Container Apps (ACA) アプリは、NoSQL ドキュメント用の Azure Cosmos DB 全体にわたるセマンティック検索を提供するバックエンド API サービスをホストします。API サービスは、Service Bus メッセージを処理し、検索インデックスを更新します。
Azure Kubernetes Service (AKS) は、既存の Azure Cosmos DB for NoSQL ドキュメントに対して、ベクトル埋め込みの再生成をバッチ処理します (埋め込みモデルが変更されるたびに)。
エクストラネットに接続されたコンテナ化されたWebhookにより、ビジネスパートナーは文書を送信して、内部のAIワークフローで処理し、意味検索や取得を行うことができる。
監視
Azure リソースによって生成されたテレメトリは、Azure Monitor に送信されます。
Log Analyticsワークスペースは、ACAアプリのログ、AKSコンテナのログ、およびAzure Functionsアプリのログを収集するために使用されます。
Azure Functionsの監視は現在、Azure Application Insights SDKの計測機能を使用して実装されています。
ビジネス要件
新規または更新されたAzure Cosmos DB for NoSQLでホストされるドキュメントの埋め込みは、自動的に生成される必要があります。

バックエンドAPIサービスは、営業時間中に自動的にスケールする必要があります。
バックエンドAPIのコールドスタート時の遅延は最小限に抑える必要がある。
機密情報はコンテナイメージの外に保存する必要があります。
開発者は、Azure Functions ホストとアプリ間でテレメトリを関連付けることができなければなりません。
すべてのトレースは、OpenTelemetry SDKの計測機能を使用して実装する必要があります。
開発努力は最小限に抑えなければならない。
技術要件
コンテナイメージは、コードの更新がメインブランチにマージされる前に、自動的に構築され、検証される必要があります。
イメージのビルド自動化はAzure Container Registry内で実行される必要があり、ローカルの開発者マシンや外部のビルドサービスへの依存を排除する必要があります。
イメージビルドが開発者のローカルマシンに依存する状態は解消されなければならない。
ACAにおけるイベント駆動型スケーリングは、Azure Service Busキュー内の保留中のメッセージ数に基づいて行われる必要があります。
Azure Cosmos DB for NoSQL の RU 消費量を最小限に抑える必要があります。
ベクトル類似性検索では、Azure Cosmos DB for NoSQLに保存されている埋め込みデータを使用する必要があります。
パートナー向けのコンテナ化されたWebhookサービスは、Azure App Service上で実行する必要があります。
機密情報は、コンテナイメージ、ソースコード管理システム、またはアプリケーション構成に直接保存してはなりません。実行時に安全にアクセスできる必要があります。
すべてのシークレットはAzure Key Vaultに一元的に保存され、実行時にはマネージドIDを通じてアクセスされる必要があります。
Azure App Service は、外部サービスに依存することなく、実行時にシークレットを提供する必要があります。
リソースとワークロードは、自動化されたバージョン管理パイプラインを通じて、Bicepテンプレートを使用してデプロイする必要があります。再現性と監査可能性を確保するため、ローカルおよびコマンドラインによるデプロイは排除する必要があります。
既知の問題
ベクトル類似性クエリ実行中にRU消費量が急増する。
計画されたアプリケーションアーキテクチャに従って、バッチ埋め込みワークロードをデプロイする必要があります。
何を使うべきでしょうか？
A. YAML形式のファイル
B. kubectl run および create コマンド
C. XML形式のファイル
D. az aks コマンド
正解: **(**正解を表示します**)**
シナリオ、アプリケーションアーキテクチャ
Azure Functions は、NoSQL でホストされているドキュメントの Azure Cosmos DB のベクトル埋め込みを生成し、Service Bus にメッセージを送信して検索インデックスの更新をトリガーします。
このデプロイメントにはYAML形式のファイルを使用する必要があります。
本番環境レベルのKubernetesアーキテクチャでは、設定ファイルを用いた宣言型管理により、複雑なマイクロサービスに必要な予測可能性、バージョン管理、および自動化が実現されます。
宣言型構成: Kubernetes は、デプロイメント、KEDA ScaledObjects (イベント駆動型の Azure Functions スケーリング用)、シークレットなどのリソースの望ましい状態を定義するために、ネイティブに YAML に依存しています。
複雑なデプロイメント :ワークロードでは、複数のコンポーネント、環境変数、スケーリングルールを設定する必要があり、これらは単一のコマンドでは効率的に管理できません。
GitOps対応：設定をYAML形式で保存することで、ソースコード管理における変更履歴を追跡し、CI/CDパイプラインを通じて自動的にアップデートをデプロイすることが可能になります。
参照：
https://developer.okta.com/blog/2022/05/05/kubernetes-microservices-azure

質問: **52**
ケーススタディ2 - Proseware Inc.
背景
Proseware Inc.は、企業顧客向けにAIを活用した知識管理ソリューションを開発しています。
同社は、セマンティック検索、インテリジェントな文書検索、およびリアルタイムのパートナー連携をサポートするために、プラットフォームの近代化を進めている。
エンジニアリングチームはPythonとAzure SDKを使用しています。アーキテクチャは、コンテナ化されたマイクロサービス、ベクトル検索ワークロード、およびサーバーレスバックエンド処理をサポートするように再設計されています。
計画されたアプリケーションアーキテクチャ
マイクロサービスはDockerを使用してコンテナ化されます。
コンテナ化されたマイクロサービスとAzure Functionsアプリのコードはローカルで開発されますが、GitHubリポジトリに保存されます。
コンテナ化されたマイクロサービス用のカスタムイメージは、Azure Container Registry (ACR) に保存されます。
ベースイメージはDocker Hubに保存されます。カスタムイメージは、ベースイメージが更新されるたびに自動的に再構築される必要があります。
Azure Cosmos DB for NoSQL は、ドキュメント、メタデータ、およびベクトル埋め込みを保存します。
Azure Functions は、NoSQL でホストされているドキュメントの Azure Cosmos DB のベクトル埋め込みを生成し、Service Bus にメッセージを送信して検索インデックスの更新をトリガーします。
Azure Container Apps (ACA) アプリは、NoSQL ドキュメント用の Azure Cosmos DB 全体にわたるセマンティック検索を提供するバックエンド API サービスをホストします。API サービスは、Service Bus メッセージを処理し、検索インデックスを更新します。
Azure Kubernetes Service (AKS) は、既存の Azure Cosmos DB for NoSQL ドキュメントに対して、ベクトル埋め込みの再生成をバッチ処理します (埋め込みモデルが変更されるたびに)。
エクストラネットに接続されたコンテナ化されたWebhookにより、ビジネスパートナーは文書を送信して、内部のAIワークフローで処理し、意味検索や取得を行うことができる。
監視
Azure リソースによって生成されたテレメトリは、Azure Monitor に送信されます。
Log Analyticsワークスペースは、ACAアプリのログ、AKSコンテナのログ、およびAzure Functionsアプリのログを収集するために使用されます。
Azure Functionsの監視は現在、Azure Application Insights SDKの計測機能を使用して実装されています。
ビジネス要件
新規または更新されたAzure Cosmos DB for NoSQLでホストされるドキュメントの埋め込みは、自動的に生成される必要があります。
バックエンドAPIサービスは、営業時間中に自動的にスケールする必要があります。
バックエンドAPIのコールドスタート時の遅延は最小限に抑える必要がある。
機密情報はコンテナイメージの外に保存する必要があります。
開発者は、Azure Functions ホストとアプリ間でテレメトリを関連付けることができなければなりません。
すべてのトレースは、OpenTelemetry SDKの計測機能を使用して実装する必要があります。
開発努力は最小限に抑えなければならない。
技術要件
コンテナイメージは、コードの更新がメインブランチにマージされる前に、自動的に構築され、検証される必要があります。
イメージのビルド自動化はAzure Container Registry内で実行される必要があり、ローカルの開発者マシンや外部のビルドサービスへの依存を排除する必要があります。
イメージビルドが開発者のローカルマシンに依存する状態は解消されなければならない。
ACAにおけるイベント駆動型スケーリングは、Azure Service Busキュー内の保留中のメッセージ数に基づいて行われる必要があります。
Azure Cosmos DB for NoSQL の RU 消費量を最小限に抑える必要があります。
ベクトル類似性検索では、Azure Cosmos DB for NoSQLに保存されている埋め込みデータを使用する必要があります。
パートナー向けのコンテナ化されたWebhookサービスは、Azure App Service上で実行する必要があります。
機密情報は、コンテナイメージ、ソースコード管理システム、またはアプリケーション構成に直接保存してはなりません。実行時に安全にアクセスできる必要があります。

すべてのシークレットはAzure Key Vaultに一元的に保存され、実行時にはマネージドIDを通じてアクセスされる必要があります。
Azure App Service は、外部サービスに依存することなく、実行時にシークレットを提供する必要があります。
リソースとワークロードは、自動化されたバージョン管理パイプラインを通じて、Bicepテンプレートを使用してデプロイする必要があります。再現性と監査可能性を確保するため、ローカルおよびコマンドラインによるデプロイは排除する必要があります。
既知の問題
ベクトル類似性クエリ実行中にRU消費量が急増する。
ビジネス要件に従って、AKS、ACA、およびAzure Functionsアプリから収集したログを一度だけ分析する必要があります。
何を使うべきでしょうか？
A. KQLクエリ
B. Azureアクティビティログ
C. Python関数
D. Azure Monitorワークブック
正解**: A (**コメントを発表する**)**
ビジネス要件
開発者は、Azure Functions ホストとアプリ間でテレメトリを関連付けることができなければなりません。
KQLクエリは、AKS、Azure Container Apps（ACA）、およびAzure Functions間のテレメトリを関連付けるための、一度限りのクロスリソース分析に使用するのに最適なツールです。
クロスリソースクエリ（KQL（Kustoクエリ言語）を使用すると、ユニオン演算子を使用して、異なるワークスペースまたはアプリケーションのテーブルを簡単に結合できます。
相関機能: operation_Id などの標準クラウドフィールドを使用してログを結合できます。
_ResourceId、またはカスタムヘッダーを使用して、アーキテクチャ全体で単一のリクエストを追跡します。
オンデマンド分析：セットアップ時間が不要なため、Azure Monitor Log Analyticsインターフェイス内で直接、一度限りの調査を行うのに最適です。
参照：
https://oneuptime.com/blog/post/2026-02-16-how-to-perform-cross-workspace-queries-in-azure-log-analytics/view

質問**: 53**
AKSクラスター内のコンテナが繰り返し再起動する。
ノードレベルのCPUとメモリのメトリクスは正常であるにもかかわらず、Podイベントにはプローブの失敗が記録されている。
繰り返し再起動が発生する原因を診断する必要があります。
まず最初に何をすべきでしょうか？
A. より多くのレプリカに展開規模を拡大します。
B. コンテナの生存性プローブのinitialDelaySecondsを減らします。
C. ポッドをホストしているノードをドレインして再起動します。
D. ポッドのイベントとコンテナのログを調べます。
正解**: (**正解を表示します**)**
最初のトラブルシューティング手順は、Pod イベントとコンテナ ログを検査することです。Microsoft の AKS トラブルシューティング ガイダンスでは、Pod が繰り返し再起動したり、CrashLoopBackOff などの状態になったりする場合、Kubernetes イベントとアプリケーション ログが主要な診断ソースとして挙げられています。kubectl describe pod <pod-name>
> は、失敗したライブネスプローブやレディネスプローブ、再起動の理由、スケジューリングの問題、コンテナの終了情報などのイベントを公開します。kubectl logs <pod-name> はアプリケーションレベルの出力を提供しますが、kubectl logs <pod-name> --previous は、コンテナが既に再起動されている場合に特に役立ちます。これは、以前に終了したコンテナインスタンスからログを取得するためです。
シナリオでは既にノードのCPUとメモリが正常であると規定されているため、ノードのリソース負荷が高まる可能性は低い。
プローブの失敗は、プローブのパスやポートの誤り、タイムアウト、依存関係チェックの不備、アプリケーションの初期化の遅延、またはアプリケーションエラーなどが原因で発生する可能性があります。Microsoft は、インフラストラクチャを変更する前に、ヘルスプローブの構成とアプリケーションの動作を確認することを特に推奨しています。

レプリカ数を増やしても根本原因は特定できません。initialDelaySecondsを小さくすると、起動の遅いアプリケーションがかえって早く失敗する可能性があります。ノードレベルの障害を示す証拠がない限り、ノードのドレインや再起動は正当化されません。
学習ガイドの参照: AKS のトラブルシューティング # Kubernetes イベント、コンテナ ログ、ライブネス/レディネス プローブ、kubectl describe、kubectl logs --previous。

質問: **54**
あなたは、新しいAzure Service Busキューからの注文を処理するAzure Functionを設計しています。
システムがメッセージを複数回処理しないようにし、失敗したメッセージは調査のために保存する必要があります。
どの2つの行動を実行すべきでしょうか？それぞれの正解は、解決策の一部を示しています。
注：正解ごとに1ポイントが加算されます。
A. 未配達郵便物処理プロセスを含める。
B. HTTPトリガーを使用して注文を処理します。
C. キューで重複検出を有効にする。
D. サービスバスでセッションを有効にする。
正解: **A,C (**コメントを発表する**)**
重複注文をゼロに抑え、安全なエラー処理を実現することで、信頼性の高い注文処理を保証するには、重複検出、PeekLock決済、およびカスタムデッドレターリーティングを利用したAzure Service Busトリガーを備えたAzure Functionを実装する必要があります。
参考文献：
https://azureintegrations.com/2020/03/01/azure-servce-bus-best-practices/

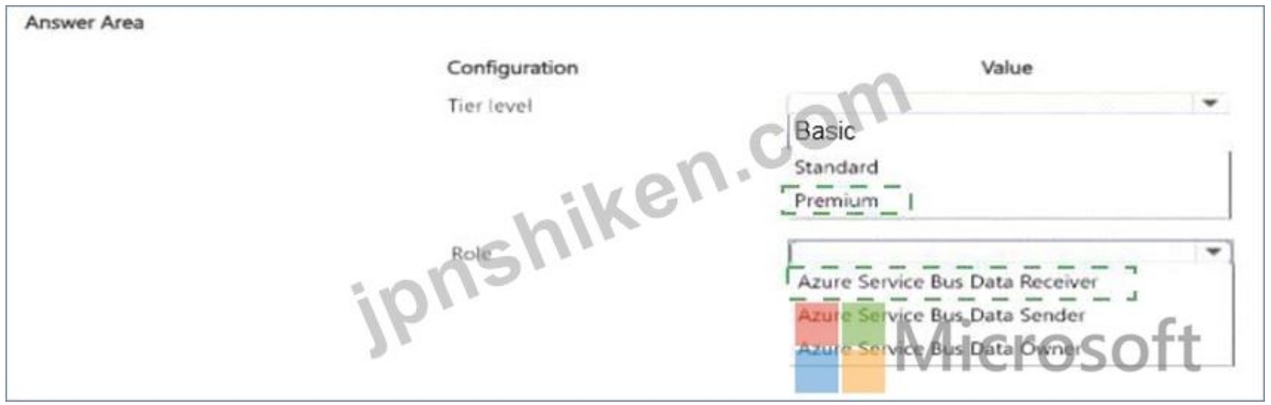
質問: **55**
外部API呼び出しに依存するAzure Service Busメッセージを処理します。
APIが一時的に利用できない場合は、配信回数を増やさずにメッセージの処理を遅延させる必要があります。APIが利用可能になったときにメッセージを処理し、同時にメッセージへのアクセスを維持する方法を見つける必要があります。どのようなメッセージアクションを実行すべきでしょうか？
A. 延期
B. 完了
C. デッドレター
D. 放棄する
正解: **(**正解を表示します**)**
詳細説明：延期は、特定の依存関係が一時的に利用できないために現在処理できないメッセージを対象としており、アプリケーションは後でシーケンス番号で取得するためにメッセージをメインキューに保持したいと考えています。完了するとメッセージは完全に削除され、デッドレターでは失敗したメッセージとして脇に移動され、放棄するとすぐにロックが解除されて再配信されます。したがって、延期は、個々のメッセージを意図的に後で処理できるように保持しつつ、処理を延期するという要件に最も適しています。
学習ガイドの関連性 :Azureサービス統合 :Service Bus、Event Grid、Azure Functionsトリガー
バインディング、およびイベント駆動型処理。
Microsoft Learn 公式資料: AI-200 学習ガイド | Service Bus メッセージの延期 | Service Bus メッセージの転送、ロック、および決済

質問: **56**
複数の Azure Service Bus キューを使用するソリューションを開発しています。Azure Service Bus 名前空間用の Azure Event Grid サブスクリプションを作成します。メッセージの処理には、サブスクライバーとして Azure Functions を使用します。
キューからAzure Event Gridへイベントが送信できることを確認する必要があります。ソリューションは最小権限の原則に従い、コストを最小限に抑える必要があります。
どの Azure Service Bus ロールとティアレベルを使用すべきですか？回答するには、回答領域で適切なオプションを選択してください。

注：正解ごとに1ポイントが加算されます。



正解:



Explanation:



検証済み回答 :ティア :プレミアム。役割 :Azure Service Bus データ レシーバー。

詳細説明 :Microsoft のドキュメントによると、Service Bus キューとトピックの Event Grid 統合には Service Bus Premium 名前空間が必要です。Azure Function サブスクライバーはキューからメッセージを受信する必要があります。この目的のための最小限の権限を持つ組み込みデータ プレーンロールは Azure Service Bus Data Receiver です。Data Sender では受信が許可されず、Data Owner は必要以上に権限が広すぎます。したがって、Premium は機能の前提条件を満たし、Data Receiver は明示された最小限の権限の要件を満たします。

学習ガイドの関連性 :Azureサービス統合 :Service Bus、Event Grid、Azure Functionsトリガー
バインディング、およびイベント駆動型処理。

Microsoft Learn 公式資料: AI-200 学習ガイド | Service Bus と Event Grid の統合 | Microsoft Entra ID を使用した Service Bus の認証

質問: **57**

ホットスポットに関する質問

PythonのWeb APIは、トレースにOpenTelemetryを使用します。call_downstream_service関数は、requestsライブラリを使用して外部HTTPリクエストを送信します。
以下のコードは、アプリケーションにおける唯一のOpenTelemetry設定です。

```
01  from opentelemetry import trace
02  tracer = trace.get_tracer(__name__)

03  def process_request():
04    with tracer.start_as_current_span("process_request"):
05        call_downstream_service()
```

以下の各記述について、正しい場合は「はい」を選択してください。そうでない場合は「いいえ」を選択してください。
注：正解ごとに1ポイントが加算されます。

Answer Area

Identifying missing context propagation for distributed trace correlation

Statement	Yes	No
The code creates a span for each request.	○	○
The code automatically propagates trace context to downstream HTTP services.	○	○
The solution requires additional instrumentation for end-to-end trace correlation.	○	○

正解:

Answer Area

Identifying missing context propagation for distributed trace correlation

Statement	Yes	No
The code creates a span for each request.	○	○
The code automatically propagates trace context to downstream HTTP services.	○	○
The solution requires additional instrumentation for end-to-end trace correlation.	○	○

Explanation:
ボックス1 :はい
このコードはprocess_request関数用のローカルスパンを正常に作成しますが、requestsライブラリによって行われる送信HTTPリクエスト用のスパンを自動的に作成または伝播することはありません。
ボックス2 :いいえ
手動トレースのみ: 現在の設定では、tracer.start_as_current_span(...) コンテキストマネージャでラップされたコードのスパンのみが作成されます。
ボックス3 :はい
計測機能が不足しています：標準のrequestsライブラリはOpenTelemetryをネイティブに認識しません。ネットワーク呼び出しにフックするには、外部の計測パッケージが必要です。
参照：

https://learn.microsoft.com/en-us/python/api/overview/azure/core-tracing-opentelemetry-readme

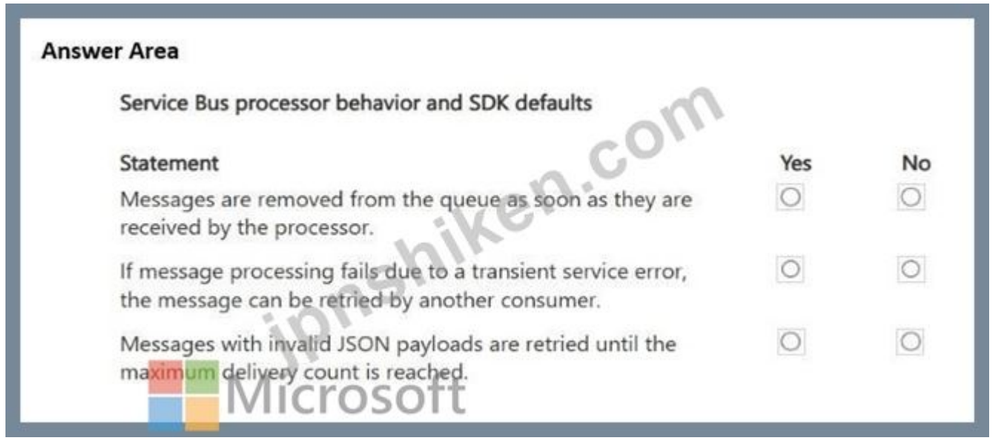
質問: **58**
ホットスポットに関する質問
あなたはバックエンドワーカーサービスのメッセージ処理コードをレビューしています。
以下のPythonコードを確認してください。このコードは、サービスバスプロセッサを初期化して起動するものです。

```
from azure.servicebus import ServiceBusClient, ServiceBusReceiveMode

01 import json
02 from azure.servicebus import ServiceBusClient, ServiceBusReceiveMode
03 from azure.identity import DefaultAzureCredential
04 credential = DefaultAzureCredential()
05 with client.get_queue_receiver(
06     queue_name="inference-requests",
07     receive_mode=ServiceBusReceiveMode.PEEK_LOCK,
08     max_wait_time=30
09 ) as receiver:
10      for msg in receiver:
11          try:
12              payload = json.loads(str(msg))
13              result = run_inference(payload)
14              save_result(result, payload.get("document_id"))
15              receiver.complete_message(msg)
16          except json.JSONDecodeError:
17              receiver.dead_letter_message(
18                  msg,
19                  reason="MalformedPayload",
20                  error_description="Message body is not valid JSON"
21              )
22          except ModelNotFoundError:
23              receiver.dead_letter_message(
24                  msg,
25                  reason="UnsupportedModel",
26                  error_description=f"Model {payload.get('model')} is not available"
27              )
28          except TransientServiceError:
29              receiver.abandon_message(msg)
```

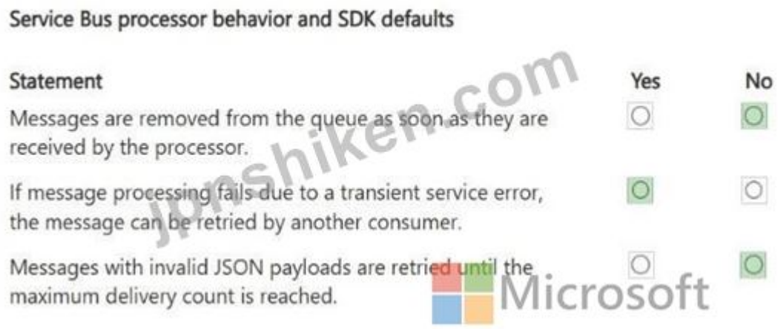
以下の各記述について、正しい場合は「はい」を選択してください。そうでない場合は「いいえ」を選択してください。

注：正解ごとに1ポイントが加算されます。



正解:

Answer Area



Explanation:

ボックス1 :いいえ

いいえ、PEEK_LOCKモードを使用している場合、メッセージは受信後すぐにキューから削除されるわけではありません。

PEEK_LOCKの仕組み

ロック :サービスバスプロセッサは、特定の期間、メッセージをワーカー専用にロックします。

不可視状態 :ロックが有効な間は、他のワーカーはメッセージを見たり処理したりすることはできません。

削除 :メッセージは、正常に処理され、明示的に完了した後にのみ、キューから完全に削除されます。

ボックス2 :はい

はい、TransientServiceError が原因でメッセージが失敗した場合、別のコンシューマーによって再試行される可能性があります。

明示的な放棄: このコードは TransientServiceError をキャッチしたときに receiver.abandon_message(msg) を呼び出します。

ロック解除 :メッセージを破棄すると、現在のワーカーが保持している PEEK_LOCK が即座に解除されます。

即時利用可能 :メッセージはアクティブキューに戻り、他の競合する消費者に即座に表示されるようになります。

ボックス3 :いいえ

いいえ、無効なJSONペイロードを含むメッセージは、最大配信回数に達するまで再試行されません。最初の試行で即座にデッドレターキュー（DLQ）に送られます。

即時デッドレター処理: コードは json.JSONDecodeError をキャッチし、明示的に receiver.dead_letter_message(msg) を呼び出します。

即時終了: この API 呼び出しは、Azure Service Bus に対して、通常の配信カウント ロジックをバイパスして、メッセージをすぐに DLQ に移動するように指示します。
再配信なし: メッセージがDLQに移動されたため、ブローカーはそれをメインキューから削除し、それ以上の処理試行を停止します。
参照：
https://learn.microsoft.com/en-us/python/api/azure-servicebus/azure.servicebus.aio.servicebusreceiver
https://learn.microsoft.com/en-us/azure/service-bus-messaging/service-bus-python-how-to-use-queues

質問: **59**
Azure Container Apps 上で動作する複数のマイクロサービスを開発しています。これらのマイクロサービスに対して、外部 HTTP 受信トラフィックが有効になっています。
デプロイ済みのマイクロサービスは、ユーザーが新機能をテストできるように更新する必要があります。以下の要件を満たす必要があります。
* 更新されたマイクロサービスがテストユーザーに提供するための単一のURLを有効にし、維持する。
* 現在のマイクロサービスバージョンに対応するマイクロサービスを更新します。
Azure Container Apps を構成する必要があります。
どの機能を設定する必要がありますか？回答するには、回答欄で適切なオプションを選択してください。
注：正解ごとに1ポイントが加算されます。



正解:



Explanation:



検証済み回答: テストユーザー用の単一URL: リビジョンラベル。現在のマイクロサービスアクティベーション動作:
復習モード。
詳細説明 :Azure Container Apps のリビジョン ラベルは、安定したラベル固有の URL を提供し、これを別のリビジョンに再割り当てできるため、一貫性のあるテスト エンドポイントに適しています。リビジョン モードは、同時にアクティブにできるリビジョンの数を制御し、現在のアプリケーション バージョンと更新されたアプリケーション バージョンがどのようにアクティブ化されるかを決定します。コンテナ イメージとレジストリはデプロイされる内容を識別しますが、要求された安定したテスト URL やアプリケーション レベルのリビジョン アクティブ化動作は提供しません。
学習ガイドの関連性 :コンテナ化された Azure ワークロード :レジストリの構築、App Service コンテナ、コンテナ アプリの改訂/スケーリング動作、および AKS デプロイの選択肢。
Microsoft Learn 公式リファレンス: AI-200 学習ガイド | Azure Container Apps の改訂履歴 | Azure Container Apps の改訂ラベル

質問: **60**
外部 API を呼び出すために、API へのアクセス トークンを提供する Azure Functions を開発しています。アクセス トークンは、mykeyvault という名前の Azure Key Vault 内の token という名前のシークレットに保存されています。Azure Functions がこのトークンにアクセスできるようにする必要があります。Azure Functions アプリ構成には、どの値を保存すればよいでしょうか？
A. KeyVault:mykeyvault;Secret:token
B. アプリ:設定:シークレット:mykeyvault:token
C. @Microsoft.KeyVault(SecretUri=https://mykeyvault.vault.azure.net/secrets/token/)
D. AZUREKVCONNSTR_ https://mykeyvault.vault.azure.net/secrets/token/
正解: **(正解を表示します)**
Azure Functions にシークレットへのアクセスを安全に提供しつつ、アプリケーション コード内でシークレットを公開しないようにするには、Key Vault の参照構文を Azure Functions アプリの構成設定に保存する必要があります。
保存する必要のある正確な値は、シークレットの最新バージョンを自動的に対象とするか、特定の固定バージョンを対象とするかによって異なります。
最新バージョンをターゲットにする（推奨）
Function Appが常に最新バージョンのトークンを取得するようにするには、以下の構文を使用してください。
@Microsoft.KeyVault(VaultName= mykeyvault;SecretName=token)
参照：
https://learn.microsoft.com/en-us/azure/key-vault/general/apps-api-keys-secrets

質問: **61**
ケーススタディ1 - ファブリカム社
背景
Fabrikam Inc.は、オンライン小売業者向けにAIを活用した需要予測および商品推奨サービスを提供するグローバルな小売分析企業です。同社は、ソリューションをMicrosoft Azure上で完全に稼働させるための近代化を進めています。
このプラットフォームは、トランザクションデータを取り込み、意味検索のための埋め込みを生成し、ベクトル類似性検索を実行し、コンテナ化されたマイクロサービスを通じて製品の推奨結果を返します。開発者はPythonとAzure SDKを使用します。運用チームは、コンテナのオーケストレーション、スケーリング、監視、およびセキュリティを管理します。
そのソリューションは、厳格な性能、拡張性、およびセキュリティ要件を満たす必要がある。
現在の環境
アプリケーションアーキテクチャ
レコメンデーションエンジンは、コンテナ化されたPythonアプリケーションとして動作する、顧客向けのHTTP APIです。このエンジンはAzure Container Apps（ACA）にデプロイされます。
埋め込みデータは、pgvectorを使用してAzure Database for PostgreSQLに保存されます。
セマンティック検索は、メタデータフィルタリングとベクトル類似性検索を組み合わせた手法を用いる。
Azure Managed Redisはキャッシュ層として使用されます。
フロントエンドおよびAPIワークロードは、Azure Container Apps（ACA）にデプロイされます。

バッチモデル再トレーニングワークロードは、Azure Kubernetes Service (AKS) で実行されます。
コンテナとCI/CD
コンテナイメージはAzure Container Registry（ACR）に保存されます。
CI/CDは、コミット時にイメージを構築するためにACRタスクを使用します。
ACA環境は改訂管理をサポートしています。
AKSワークロードは、Gitに保存されているKubernetesマニフェストファイルを使用してデプロイされます。
監視
ログはAzure Monitorに収集されます。
トラブルシューティングを行う際、チームはコンテナログとKubernetesイベントを調査します。
開発者は、レイテンシの急増を分析するためにKQLクエリを作成します。
ビジネス要件
顧客体験：予測不可能な季節的なトラフィックの急増時でも、エンドユーザーに対してシームレスで低遅延のレコメンデーション体験を維持する。
運用コスト効率：非アクティブ期間中にリソースの割り当てを解除し、スケールコストの暴走を防ぐことで、コンピューティング費用を最小限に抑えます。
データの整合性と鮮度：顧客の不満を防ぐため、商品のおすすめ情報が常に最新のカタログメタデータと価格を反映していることを確認してください。
セキュリティとコンプライアンス：長期にわたる認証情報を排除し、すべての機密情報の管理を一元化することで、ゼロトラストセキュリティモデルを遵守する。
グローバルな拡張性：既存の小売業者のクエリパフォーマンスを低下させることなく、毎日数百万件もの新しい商品埋め込みを迅速に取り込むことをサポートします。
技術要件
パフォーマンス：セマンティック検索のレイテンシは、ピーク負荷時でも200ミリ秒未満に抑える必要があります。
データベースの最適化：埋め込みにはpgvectorを使用し、メタデータフィルタリングを実装して計算オーバーヘッドを削減します。ベクトルワークロードに合わせて計算量とメモリを適切に構成し、高次元インデックスがRAMに常駐し、効率的な数学的スループットが確保されるようにします。ベクトル類似度計算は、必須のメタデータ制約を満たす製品に対してのみ実行する必要があります。
データベースのパフォーマンス：データベース接続は、接続最適化の実装を通じて、最小限の遅延で高い同時実行性をサポートする必要があります。
データロード戦略：取り込みスループットを最大化するために、セカンダリインデックスは埋め込みデータの一括ロードが完了した後にのみ適用する必要があります。
キャッシュ：Redisのキャッシュエントリは10分後に自動的に期限切れとなるように設定してください。メタデータの更新時にキャッシュエントリを無効にするリアクティブなメカニズムを実装してください。
ID: サービス間およびサービスとデータベース間の認証には、管理対象IDを使用します。
設定ファイル内に平文の認証情報を記述することは固く禁じられています。
機密情報の管理：すべての機密情報は一元的に保管する必要があります。機密情報は、一元化されたライフサイクルポリシーを使用して自動的にローテーションされる必要があります。
スケーリング：イベント駆動型スケーリングには、Kubernetesイベント駆動型オートスケーリング（KEDA）を使用します。レコメンデーションAPIはHTTPトラフィックに基づいてスケーリングする必要があり、バッチジョブはキューの長さに基づいてスケーリングし、スケール・トゥ・ゼロをサポートする必要があります。
CI/CD: すべてのイメージはAzure Container Registryに保存する必要があります。ソースコードのコミットをトリガーとしてイメージのビルドを自動化するには、ACRタスクを使用してください。
モニタリング：KQLを使用してパフォーマンステレメトリを分析し、マイクロサービスの接続障害をトラブルシューティングします。AKSおよびACAのトラブルシューティングを行う際は、ログとイベントを調べます。
ホットスポットに関する質問
Azure Database for PostgreSQLインスタンスのデータベースリソースを構成する必要があります。
ビジネス要件と技術要件を満たすために、どのように設定を完了すればよいでしょうか？回答するには、回答欄で適切なオプションを選択してください。
注：正解ごとに1ポイントが加算されます。

PostgreSQL compute, memory, and storage configuration for vector workloads

Requirement	Configuration
Meet the 200ms semantic search latency requirement.	Increase compute vCores. Increase max_connections. Increase memory allocation.
Optimize the environment for high-dimensional pgvector index residency.	Enable read replica. Increase max_connections. Increase memory allocation.
Support the continuous ingestion of transaction-based embeddings.	Enable storage autoscale. Increase compute vCores. Increase backup retention.

Microsoft
Jpnshiken.com

正解:

PostgreSQL compute, memory, and storage configuration for vector workloads

Requirement	Configuration
Meet the 200ms semantic search latency requirement.	Increase compute vCores. Increase max_connections. Increase memory allocation.
Optimize the environment for high-dimensional pgvector index residency.	Enable read replica. Increase max_connections. Increase memory allocation.
Support the continuous ingestion of transaction-based embeddings.	Enable storage autoscale. Increase compute vCores. Increase backup retention.

Microsoft
Jpnshiken.com

Explanation:

ボックス1 :コンピューティングvコア数を増やす

技術要件、パフォーマンス :セマンティック検索のレイテンシは、ピーク負荷時でも200ミリ秒未満に抑える必要があります。

ピーク負荷時のセマンティック検索のレイテンシを200ミリ秒未満に短縮するには、コンピューティングvCoreを増やすのが最善策です。

ベクトル類似性検索と意味的検索は、CPU負荷の高い処理です。ベクトル距離計算（コサイン類似度、ドット積、ユークリッド距離など）の背後にある数学は、クエリごとに実行される数学演算に大きく依存しています。システムにピーク負荷がかかると、計算用vCoreが主要なボトルネックになりやすくなります。vCoreを増やすことで並列処理能力が直接向上し、類似性計算の実行時間が劇的に短縮され、検索レイテンシを要求される200ミリ秒のSLA内に抑えることができます。

ボックス2 :メモリ割り当てを増やす

最善策は、メモリ割り当てを増やすことです。

高次元ベクトルインデックス（pgvector拡張機能で管理されるHNSWやIVFFlatなど）は、リソースを大量に消費し、RAMに大きく依存します。高速なベクトル類似性検索を維持し、高遅延のディスク操作を回避するには、ベクトルインデックス全体がメモリに収まる（RAMに常駐する）必要があります。データベースインスタンスのメモリをスケールアップすると、PostgreSQLの共有バッファとキャッシュが直接拡張され、高次元インデックスがRAMに常駐して高速な意味検索が確保されます。
ボックス3：ストレージの自動スケーリングを有効にする
トランザクションベースの埋め込みデータの継続的な取り込みをサポートするには、ストレージの自動スケーリングを有効にするのが最善策です。
継続的なデータ取り込みには動的な領域が必要: 継続的なトランザクション処理により、ベクトルデータベース (pgvector を使用した Azure Database for PostgreSQL や Azure SQL Database など) は時間の経過とともに絶えず拡張されます。
取り込みエラーの防止：ストレージ容量が制限に達すると、データベースは読み取り専用状態に切り替わります。これにより、即座にエラーが発生し、すべてのリアルタイム埋め込み書き込み操作が停止します。ストレージの自動スケーリングを有効にすると、ダウンタイムなしで環境が動的にストレージをプロビジョニングできるようになります。
参照：
https://dl.acm.org/doi/10.1145/3695053.3731013
https://learn.microsoft.com/en-us/training/paths/develop-ai-solutions-azure-database-postgresql/
https://learn.microsoft.com/en-us/azure/architecture/guide/technology-choices/vector-search

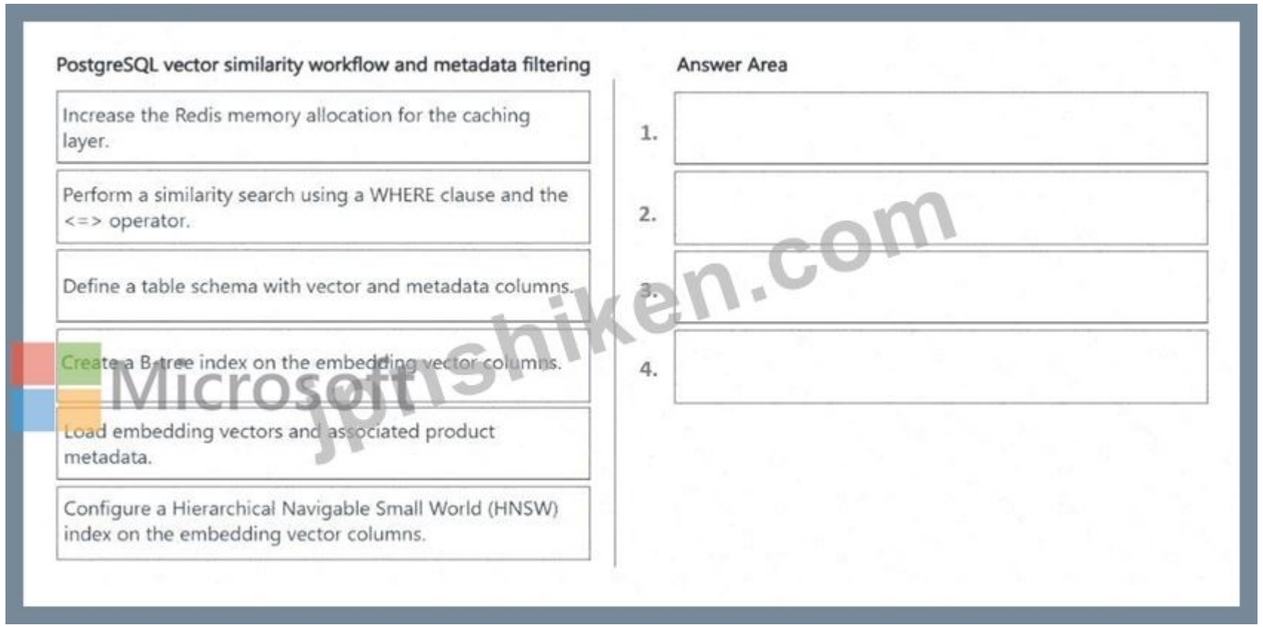
有効的な**AI-200**問題集はJPNTest.com提供され、**AI-200**試験に合格することに役に立ちます！JPNTest.comは今最新**AI-200**試験問題集を提供します。JPNTest.com AI-200試験問題集はもう更新されました。ここで**AI-200**問題集のテストエンジンを手に入れます。最新版のアクセス、https://www.jpntest.com/shiken/AI-200-mondaishu　144問、３０%ディスカウント、特別な割引コード: **JPNshiken**」

質問: **62**
Azure Cosmos DB for NoSQL コンテナー内の embedding という名前のプロパティに埋め込みデータを保存します。
入力された埋め込み表現に最も類似した5つの文書を取得する必要があります。
類似性クエリを構築する必要があります。
あなたはどうすべきでしょうか？
A. オフセットページネーションを使用してください。
B. 強い一貫性を徹底する。
C. 埋め込み値でグループ化します。
D. 結果の数を制限する。
正解: **(**正解を表示します**)**
ベクトル検索を使用してAzure Cosmos DB for NoSQLから最も類似性の高い5つのドキュメントを取得するには、結果の数を制限する必要があります。
VectorDistance() 関数を使用してベクトル類似性検索を実行する場合、クエリ内で出力サイズを直接制限する必要があります。
参照：
https://stemhash.com/semantic-search-with-llms/

質問: **63**
あなたのソリューションは、大規模な Excel ベースのレポート (例:
西地域における第3四半期の売上高はいくらでしたか？」という質問に対して、どの回答が最も適切でしょうか？
A. Excelデータを非構造化テキストとして保存し、ベクトル検索用に埋め込む
B. データを構造化テーブル/SQLソースに変換し、関数呼び出しエージェントを使用してクエリを実行します。
C. モデルのパラメータ知識に依存する

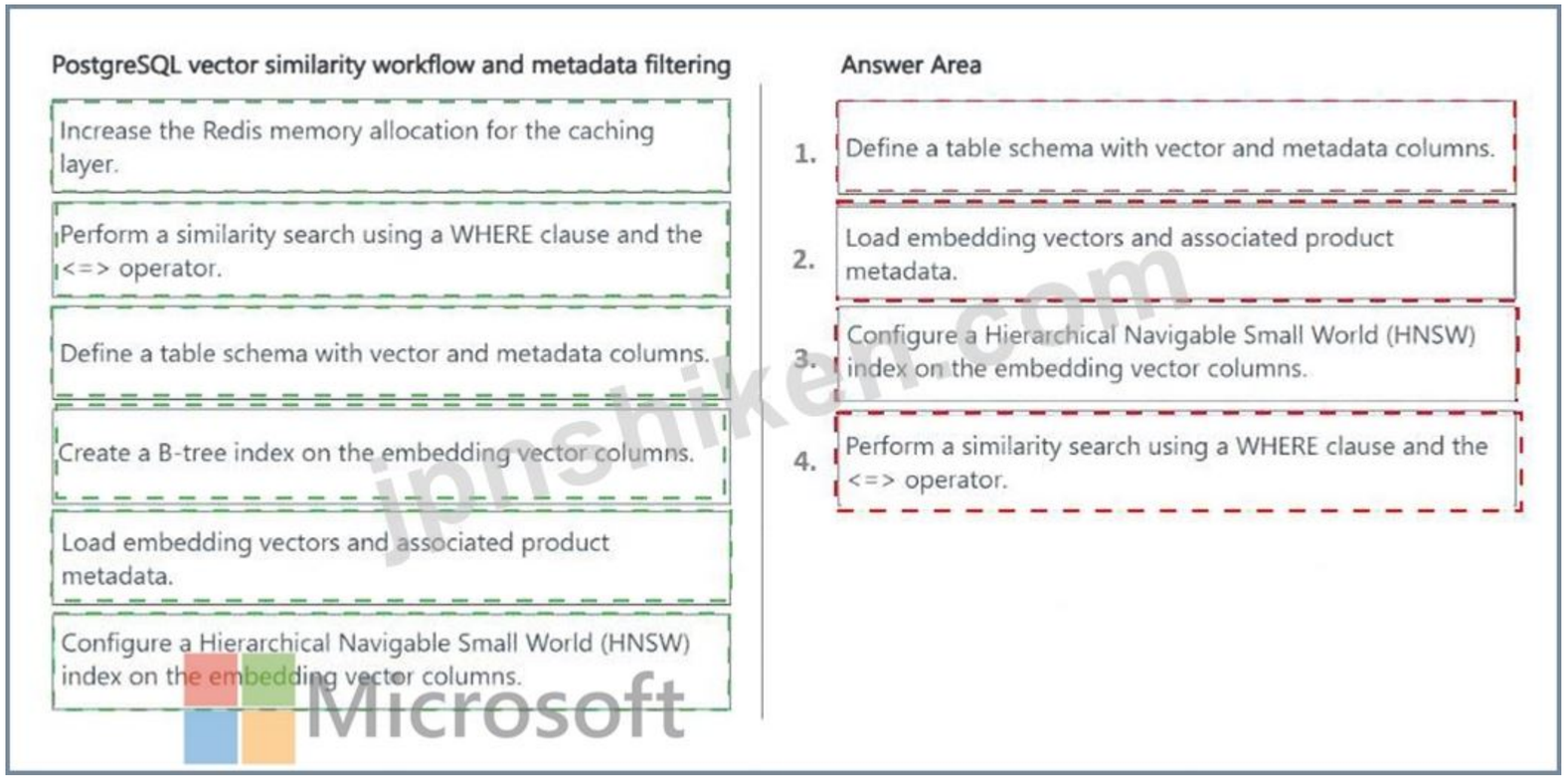
D. Azure AI Visionを使用してスプレッドシートを画像として読み込む
正解: (正解を表示します)
正確な数値／表形式の検索は、ベクトル類似性検索や数値の自由記述埋め込みではうまく処理できません。データを構造化し（例：データベースに）、関数呼び出しエージェントが正確なクエリを発行するようにすれば、正確で検証可能な回答が得られます。

質問: **64**
Fabrikam Inc.の技術要件およびパフォーマンス要件を満たすために、レコメンデーションエンジン用のセマンティック検索ワークフローを実装する必要があります。
どの4つの行動を順番に実行すべきでしょうか？回答するには、行動リストから適切な行動を回答欄に移動させ、正しい順序に並べ替えてください。



Microsoft
jpnshiken.com

正解:

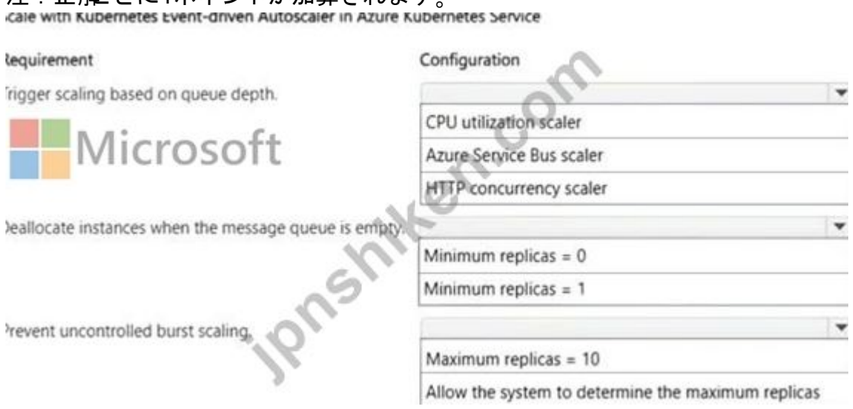


Explanation:
* ベクトル列とメタデータ列を含むテーブルスキーマを定義します。
* 埋め込みベクトルと関連する製品メタデータを読み込みます。
* 埋め込みベクトル列に階層型ナビゲーション可能なスモールワールド（HNSW）インデックスを設定します。
* WHERE句と< = >演算子を使用して類似性検索を実行します。
ワークフローではまず、pgvector埋め込み列とフィルタリングに必要な製品メタデータの両方を含むPostgreSQLスキーマを確立する必要があります。次に、埋め込みとその関連メタデータを一括ロードします。この順序は重要です。Microsoftは、ベクターインデックスを作成する前にデータをロードすることを推奨しており、その後でインデックスを作成することで、より高速なデータ取り込みと最適なインデックスレイアウトを実現できます。
読み込み後、埋め込み列にHNSWインデックスを設定します。HNSWはpgvectorでサポートされている近似最近傍インデックスであり、低遅延のベクトル検索において優れた速度と再現率のトレードオフを提供します。Microsoftは、Azure Database for PostgreSQLにおける効率的なコサイン距離検索のためのHNSWについて具体的に説明しています。
最後に、検索クエリを実行して、WHERE句を通じて必須の製品メタデータ制約を適用し、コサイン類似度演算子< = >を使用してベクトル類似度を評価します。これにより、Fabrikamの要件である、対象となる製品のみの類似度計算が満たされます。
BツリーインデックスはANNベクトルインデックスではなく、Redisのメモリを増やしてもPostgreSQLのセマンティック検索は実現されません。
学習ガイドの参照: Azure Database for PostgreSQL # pgvector、HNSW インデックス作成、バルクロード最適化、ベクトル類似性演算子、メタデータフィルタリングによる検索。

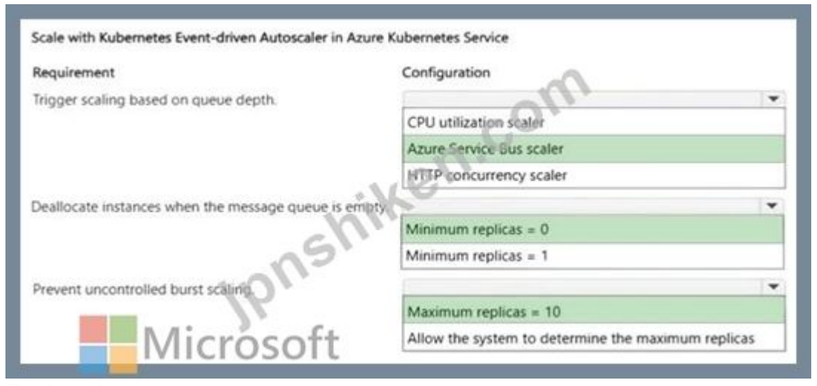
質問: **65**
ケーススタディ1 - ファブリカム社
背景

Fabrikam Inc.は、オンライン小売業者向けにAIを活用した需要予測および商品推奨サービスを提供するグローバルな小売分析企業です。同社は、ソリューションをMicrosoft Azure上で完全に稼働させるための近代化を進めています。
このプラットフォームは、トランザクションデータを取り込み、意味検索のための埋め込みを生成し、ベクトル類似性検索を実行し、コンテナ化されたマイクロサービスを通じて製品の推奨結果を返します。開発者はPythonとAzure SDKを使用します。運用チームは、コンテナのオーケストレーション、スケーリング、監視、およびセキュリティを管理します。
そのソリューションは、厳格な性能、拡張性、およびセキュリティ要件を満たす必要がある。
現在の環境
アプリケーションアーキテクチャ
レコメンデーションエンジンは、コンテナ化されたPythonアプリケーションとして動作する、顧客向けのHTTP APIです。このエンジンはAzure Container Apps（ACA）にデプロイされます。
埋め込みデータは、pgvectorを使用してAzure Database for PostgreSQLに保存されます。
セマンティック検索は、メタデータフィルタリングとベクトル類似性検索を組み合わせた手法を用いる。
Azure Managed Redisはキャッシュ層として使用されます。
フロントエンドおよびAPIワークロードは、Azure Container Apps（ACA）にデプロイされます。
バッチモデル再トレーニングワークロードは、Azure Kubernetes Service (AKS) で実行されます。
コンテナとCI/CD
コンテナイメージはAzure Container Registry（ACR）に保存されます。
CI/CDは、コミット時にイメージを構築するためにACRタスクを使用します。
ACA環境は改訂管理をサポートしています。
AKSワークロードは、Gitに保存されているKubernetesマニフェストファイルを使用してデプロイされます。
監視
ログはAzure Monitorに収集されます。
トラブルシューティングを行う際、チームはコンテナログとKubernetesイベントを調査します。
開発者は、レイテンシの急増を分析するためにKQLクエリを作成します。
ビジネス要件顧客体験：予測不可能な季節的なトラフィックの急増
時でも、エンドユーザーに対してシームレスで低遅延のレコメンデーション体験を維持する。
運用コスト効率：非アクティブ期間中にリソースの割り当てを解除し、スケールコストの暴走を防ぐことで、コンピューティング費用を最小限に抑えます。
データの整合性と鮮度：顧客の不満を防ぐため、商品のおすすめ情報が常に最新のカタログメタデータと価格を反映していることを確認してください。
セキュリティとコンプライアンス：長期にわたる認証情報を排除し、すべての機密情報の管理を一元化することで、ゼロトラストセキュリティモデルを遵守する。
グローバルな拡張性：既存の小売業者のクエリパフォーマンスを低下させることなく、毎日数百万件もの新しい商品埋め込みを迅速に取り込むことをサポートします。
技術要件
パフォーマンス :セマンティック検索のレイテンシは、ピーク負荷時でも200ミリ秒未満に抑える必要があります。
データベースの最適化：埋め込みにはpgvectorを使用し、メタデータフィルタリングを実装して計算オーバーヘッドを削減します。ベクトルワークロードに合わせて計算量とメモリを適切に構成し、高次元インデックスがRAMに常駐し、効率的な数学的スループットが確保されるようにします。ベクトル類似度計算は、必須のメタデータ制約を満たす製品に対してのみ実行する必要があります。
データベースのパフォーマンス :データベース接続は、接続最適化の実装を通じて、最小限の遅延で高い同時実行性をサポートする必要があります。
データロード戦略：取り込みスループットを最大化するために、セカンダリインデックスは埋め込みデータの一括ロードが完了した後にのみ適用する必要があります。
キャッシュ :Redisのキャッシュエントリは10分後に自動的に期限切れとなるように設定してください。メタデータの更新時にキャッシュエントリを無効にするリアクティブなメカニズムを実装してください。
ID: サービス間およびサービスとデータベース間の認証には、管理対象IDを使用します。
設定ファイル内に平文の認証情報を記述することは固く禁じられています。
機密情報の管理 :すべての機密情報は一元的に保管する必要があります。機密情報は、一元化されたライフサイクルポリシーを使用して自動的にローテーションされる必要があります。

スケーリング :イベント駆動型スケーリングには、Kubernetesイベント駆動型オートスケーリング (KEDA) を使用します。レコメンデーションAPIはHTTPトラフィックに基づいてスケーリングする必要があり、バッチジョブはキューの長さに基づいてスケーリングし、スケール・トゥ・ゼロをサポートする必要があります。
CI/CD: すべてのイメージはAzure Container Registryに保存する必要があります。ソースコードのコミットをトリガーとしてイメージのビルドを自動化するには、ACRタスクを使用してください。
モニタリング :KQLを使用してパフォーマンステレメトリを分析し、マイクロサービスの接続障害をトラブルシューティングします。AKSおよびACAのトラブルシューティングを行う際は、ログとイベントを調べます。
ホットスポットに関する質問
バッチ再学習ワークロードをデプロイする必要があります。
スケーリング設定はどのように完了すればよいですか？回答するには、回答欄で適切なオプションを選択してください。
注：正解ごとに1ポイントが加算されます。



正解:



Explanation:
ボックス1 :Azure Service Busスケーラー
この要件を満たすには、Azure Service Bus スケーラーをセットアップする必要があります。
キュー深度メトリック: キュー深度のみに基づいてスケーリングをトリガーするには、オーケストレーション レイヤー (Azure Container Apps や KEDA を使用する AKS など) に、メッセージ ブローカーとネイティブに通信し、アクティブなメッセージ数などのメトリックを監視できるスケーラーが必要です。
スケール・トゥ・ゼロのサポート：標準のリソーススケーラーとは異なり、Azure Service Bus スケーラーを使用すると、イベント駆動型のバッチワークロードは、メッセージが到着したときにコンシューマーを起動し、キューが完全に空になったときにインスタンスをゼロまでスケールダウンできます。
ボックス2：最小レプリカ数 = 0。
最小レプリカ数を0に設定する必要があります。

最小レプリカ数を0に設定すると、キューにメッセージが残っていない場合にシステムがインスタンス数をゼロまで縮小し、リソースを完全に解放してアイドル状態のコンピューティングコストを排除できます。
ボックス3：最大複製数⇒0
バッチ再トレーニングワークロード中に制御不能なバーストスケーリングを防ぐには、最大レプリカ数を10に設定する必要があります。
上限の強制 :システムが最大レプリカ数を決定できるようにすると、基盤となるオートスケーラー（KEDAやAzure Kubernetes ServiceのHorizontal Pod Autoscalerなど）は、無限にスケールアウトしたり、プラットフォームのデフォルトの大きな制限までスケールアウトしたりする自由度を得てしまいます。これは、大量のバッチキューが処理される際に、制御不能なバーストスケーリングが発生するリスクを直接的に高めます。
リソースとコストの保護：上限値（最大レプリカ数＝10など）をハードコーディングすることで、バッチワークロードが割り当てられたクラスタリソースを消費しないようにし、顧客向けレコメンデーションAPIのリソース不足を防ぎ、運用コストを予測可能なものにします。
参照：
https://azure.github.io/aca-dotnet-workshop/aca/09-aca-autoscale-keda/
https://kserve.github.io/website/docs/reference/crd-api

質問: **66**
Azure Container Apps 上で動作する複数のマイクロサービスを開発しています。これらのマイクロサービスに対して、外部 HTTP 受信トラフィックが有効になっています。
デプロイ済みのマイクロサービスは、ユーザーが新機能をテストできるように更新する必要があります。以下の要件を満たす必要があります。
* 更新されたマイクロサービスがテストユーザーに提供するための単一のURLを有効にし、維持する。
* 現在のマイクロサービスバージョンに対応するマイクロサービスを更新します。
Azure Container Apps を構成する必要があります。
どの機能を設定する必要がありますか？回答するには、回答欄で適切なオプションを選択してください。
注：正解ごとに1ポイントが加算されます。



正解:

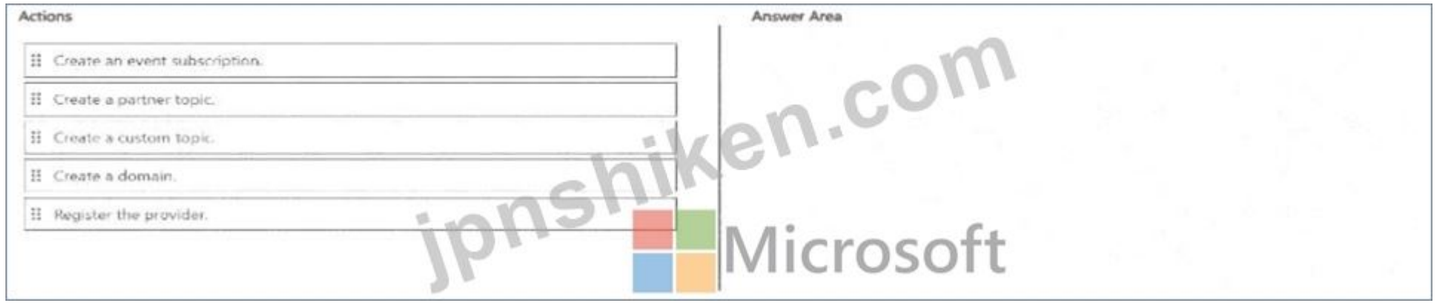


Explanation:

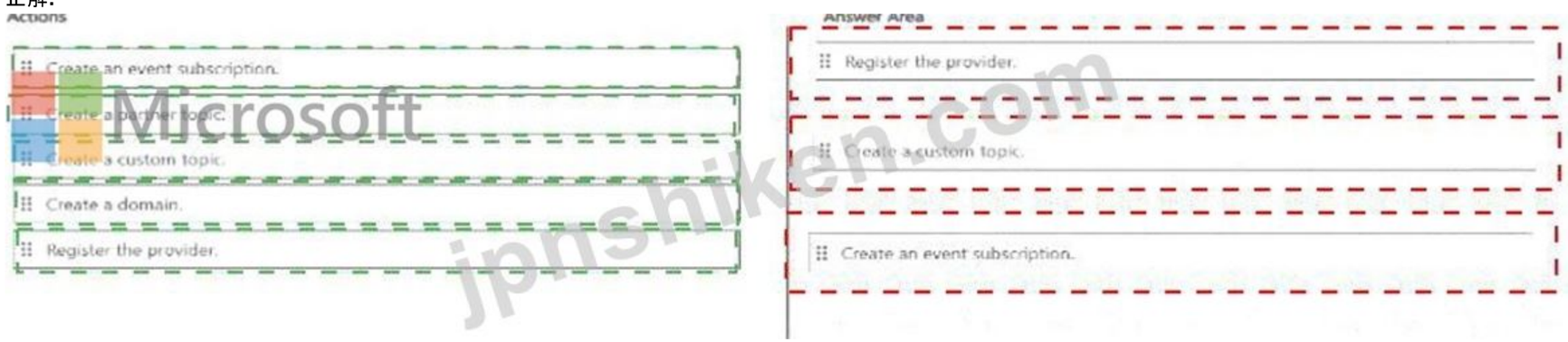


検証済み回答: テストユーザー用の単一URL: リビジョンラベル。現在のマイクロサービスアクティベーション動作:
復習モード。
詳細説明 :Azure Container Apps のリビジョン ラベルは、安定したラベル固有の URL を提供し、これを別のリビジョンに再割り当てできるため、一貫性のあるテスト エンドポイントに適しています。リビジョン モードは、同時にアクティブにできるリビジョンの数を制御し、現在のアプリケーション バージョンと更新されたアプリケーション バージョンがどのようにアクティブ化されるかを決定します。コンテナ イメージとレジストリはデプロイされる内容を識別しますが、要求された安定したテスト URL やアプリケーション レベルのリビジョン アクティブ化動作は提供しません。
学習ガイドの関連性 :コンテナ化された Azure ワークロード :レジストリの構築、App Service コンテナ、コンテナ アプリの改訂/スケーリング動作、および AKS デプロイの選択肢。
Microsoft Learn 公式リファレンス: AI-200 学習ガイド | Azure Container Apps の改訂履歴 | Azure Container Apps の改訂ラベル

質問: **67**
新しくプロビジョニングされた Azure サブスクリプションがあります。AI 推論イベント用のカスタム Event Grid ワークフローを設計しています。
信頼性の高いイベントを下流のプロセッサにルーティングするには、イベントグリッドのコンポーネントを実装する必要があります。
どの3つの行動を順番に実行すべきでしょうか？回答するには、行動リストから適切な行動を回答欄に移動させ、正しい順序に並べ替えてください。



正解:



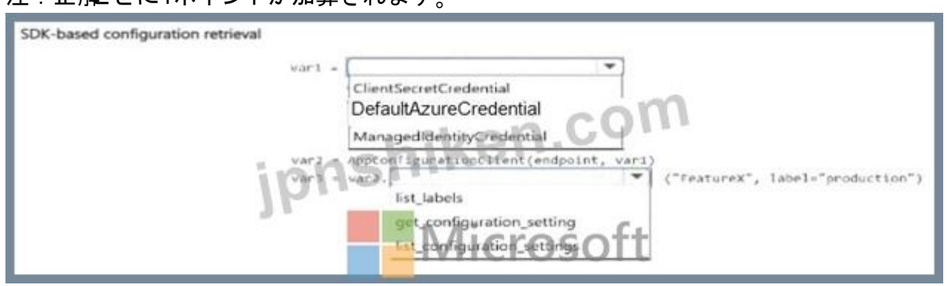
Explanation:



検証済みの回答 :1) Event Gridリソースプロバイダーを登録します。2) カスタムトピックを作成します。3) イベントサブスクリプションを作成します。
詳細説明 :Event Gridリソースを作成するには、新しくプロビジョニングされたサブスクリプションにEvent Gridリソースプロバイダが利用可能である必要があります。パブリッシャーは、イベントイングレスリソースとしてカスタムトピックを必要とし、そのトピック上のイベントサブスクリプションによってダウンストリームプロセッサへのルーティングが定義されます。イベントサブスクリプションには、イベントタイプやデータフィールドなどのフィルタを含めることができ、信頼性の高いイベントのみがプロセッサに到達します。上記のカスタムワークフローでは、パートナートピックまたはドメインを作成する必要はありません。
学習ガイドの関連性 :Azureサービス統合 :Service Bus、Event Grid、Azure Functionsトリガー
バインディング、およびイベント駆動型処理。
Microsoft Learn 公式資料: AI-200 学習ガイド | イベント グリッドのカスタム トピックまたはドメインを作成する

質問: **68**
Azure AI Foundry に GPT-40 モデルをデプロイしました。ユーザーから、同じプロンプトに対して一貫性のない補完結果が報告されています。プロンプトを変更せずに、出力結果をより決定論的にする必要があります。どうすればよいですか？
A. max_tokensパラメータを増やす
B. 温度を0などの低い値に設定する
C. top_pパラメータを増やす
D. コンテンツフィルタリングを有効にする
正解: (正解を表示します)
温度はトークン選択におけるランダム性を制御します。温度を0に近い値に設定すると、モデルは常に最も確率の高い次のトークンを選択するようになり、同一のプロンプトに対してより決定論的で再現性の高い出力が得られます。

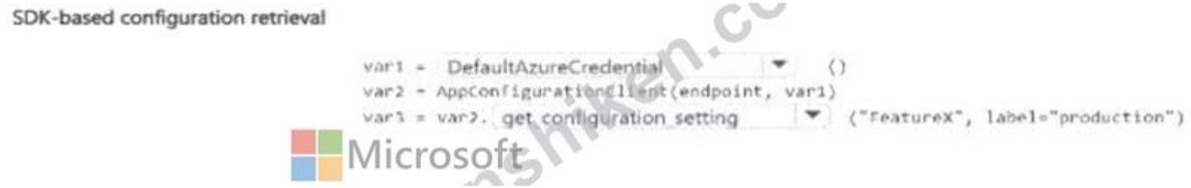
質問: **69**
Python SDK を使用して、Azure App Configuration リソースに保存されているランタイム設定の値を取得するコンテナ化された AI アプリケーションを開発しています。このアプリケーションは、マネージド ID のセキュリティ コンテキストで Azure 上で実行されます。
アプリケーションは、コードの変更なしにローカル開発環境で動作する必要があります。
取得処理を実装するコードを完成させる必要があります。
コードをどのように完成させればよいでしょうか？回答するには、回答欄で適切な選択肢を選んでください。
注：正解ごとに1ポイントが加算されます。



正解:



Explanation:



検証済みの回答: var1 = DefaultAzureCredential(); 取得呼び出し = get_configuration_setting(...)。

詳細説明 :DefaultAzureCredential は、ローカル開発環境と Azure ホスト環境で同じアプリケーション コードをサポートするように設計されています。ローカルでは開発者の資格情報を使用できますが、Azure ではワークロードのマネージド ID を使用できます。アプリ構成クライアントの get_configuration_setting メソッドは、表示されているコードに一致する特定のキーとラベルのペアを取得します。ManagedIdentityCredential を直接使用しても、コードや環境固有の処理を行わない限り、同じローカル開発資格情報チェーンは提供されません。

学習ガイドの関連性 :セキュリティと運用 :Key Vault、アプリ構成、マネージドID、OpenTelemetry、Azure Monitor、およびKQLベースのトラブルシューティング。

Microsoft Learn 公式リファレンス: AI-200 学習ガイド | Azure アプリ構成 Python クライアント | Azure リソースのマネージド ID

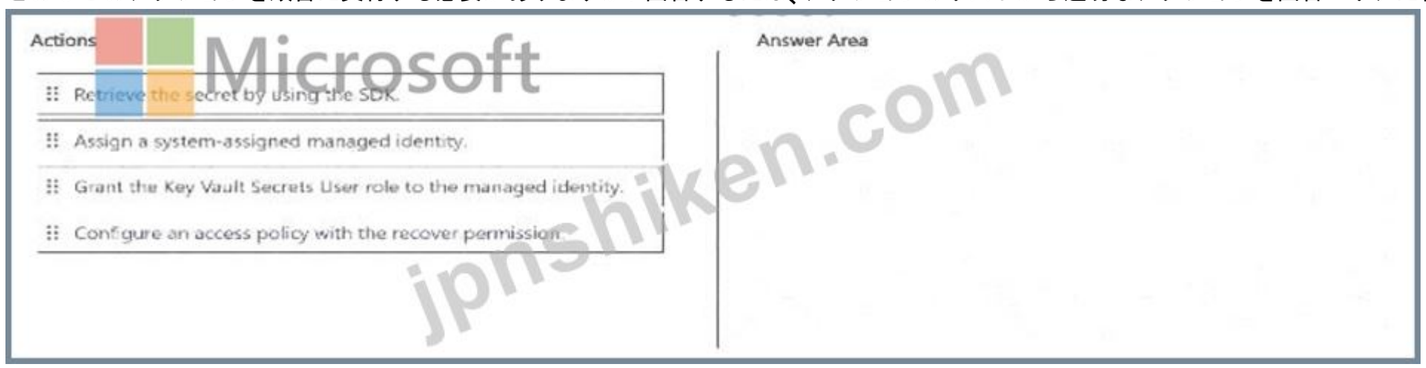
質問: 70

Azure Container Apps に AI サービスをデプロイしています。

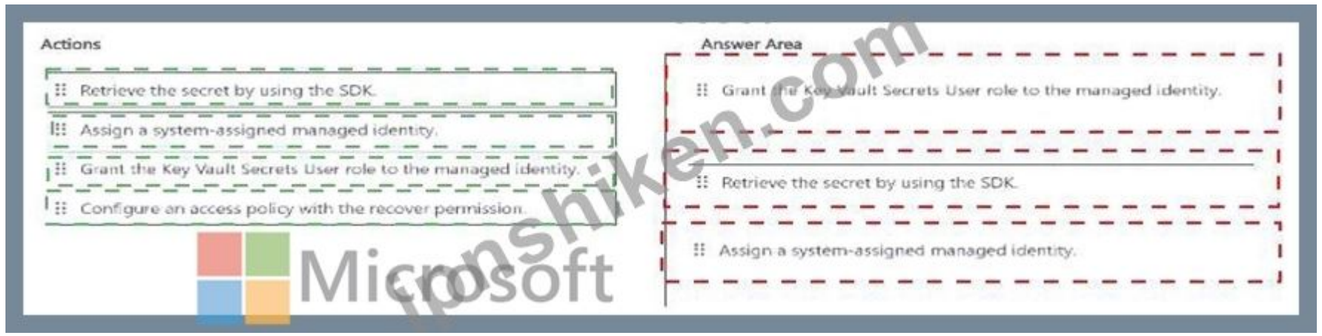
サービスは、マネージドIDを使用してKey Vaultからシークレットを安全に取得する必要があります。

安全なアクセスを設定する必要があります。

どの 3 つのアクションを順番に実行する必要がありますか? 回答するには、アクションのリストから適切なアクションを回答エリアに移動して、正しい順序に並べ替えてください。



正解:



Explanation:



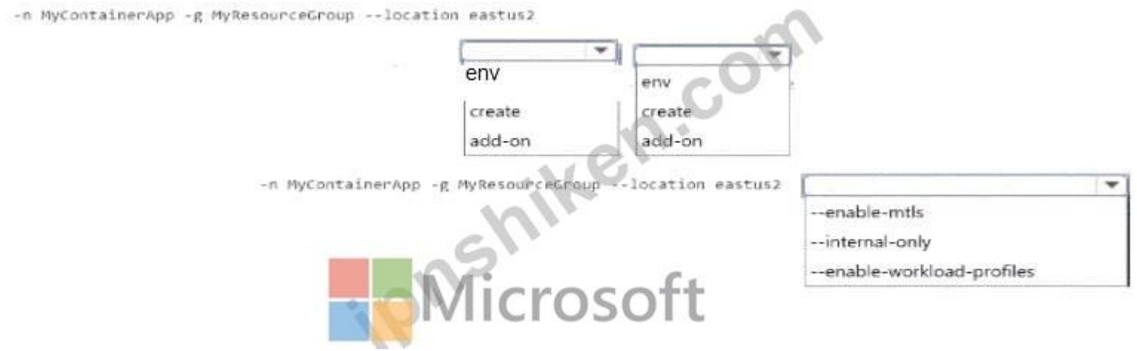
質問: **71**
Azure SQL Databaseに接続する本番環境用のAzure Functionsアプリをデプロイします。
ソリューションは以下の機能を提供する必要があります。
* ソースコード管理において機密情報が漏洩するのを防ぐ。
* 関数アプリを再デプロイすることなく、シークレットのローテーションをサポートします。
認証情報の更新中は、システム停止時間を避けてください。
安全で保守性の高いシークレット管理を設定する必要があります。どのような設定を行うべきでしょうか？
A. スタートアップクラスにハードコードされた接続文字列
B. パラメータファイルはソース管理に保存されています
C. Key Vault参照を含むアプリケーション設定
D. local.settings.json 内の環境変数
正解: **C** (コメントを発表する)

質問: **72**
あなたは、Azure Container Apps を使用するマイクロサービスベースのアプリケーションを開発しています。このアプリケーションは、注文処理、在庫管理、レポート生成などのタスクを処理する複数のコンテナ化されたサービスで構成されています。あなたは、注文処理アプリケーションの新しいリビジョンをデプロイします。
注文処理はWebリクエストによって開始される必要があり、受信したWebリクエストに基づいて常に利用可能でなければならない。
レプリカが受信リクエストを処理できる状態になっていることを検証する必要があります。
何を実装すべきでしょうか？
A. HTTPライブネスプローブ
B. HTTP起動プローブ
C. TCP準備完了プローブ

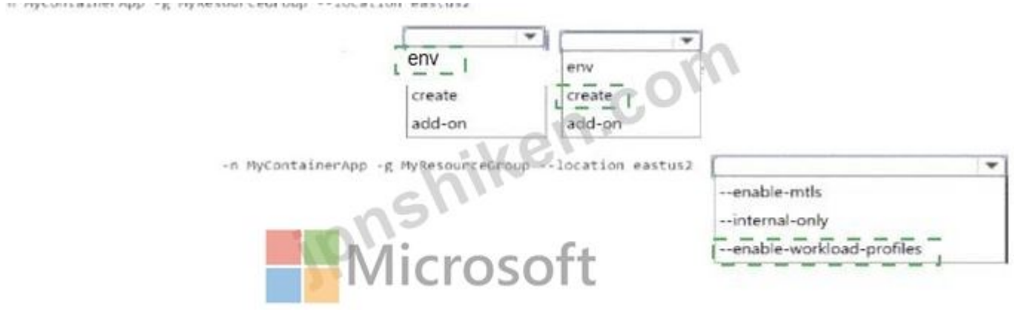
D. HTTP準備状況プローブ
E. TCPライブネスプローブ
正解: **(**正解を表示します**)**
詳細説明：準備プローブは、実行中のレプリカがトラフィックを受信する準備ができているかどうかを判定します。Webトリガー型の注文処理サービスの場合、HTTP準備プローブはアプリケーションエンドポイントをテストし、レプリカがリクエストを処理できるようになるまでルーティングプールから除外することができます。ライブネスプローブはプロセスを再起動する必要があるかどうかを検出し、スタートアッププローブは初期化中に起動が遅いコンテナを保護します。TCP準備プローブはソケットの受け入れをテストできますが、ここで説明するHTTPサービスには、HTTP準備プローブの方がアプリケーションをより認識できる選択肢です。
学習ガイドの関連性 :コンテナ化された Azure ワークロード :レジストリの構築、App Service コンテナ、コンテナ アプリの改訂/スケーリング動作、および AKS デプロイの選択肢。
Microsoft Learn 公式資料: AI-200 学習ガイド | Azure Container Apps のヘルス プローブ

質問: **73**
あなたは、Azure Key Vaultから接続文字列とAPIキーを取得する、AIを活用したAPIを開発しています。
以下のセキュリティ機能を提供するソリューションを構成する必要があります。
* APIは、アプリケーション構成ファイルに認証情報を保存することなく、Key Vaultに対して認証を行う必要があります。
* APIで使用されるIDは、シークレットを読み取るために必要な最小限の権限のみを持つ必要があります。
* 設定は、IDまたは認証情報が侵害された場合の被害範囲を最小限に抑える必要があります。
APIへの安全なアクセス戦略を実装する必要があります。
どの2つの行動を実行すべきでしょうか？それぞれの正解は解決策の一部を示しています。2つ選択してください。
注：正解ごとに1ポイントが加算されます。
A. Azure App Configuration に秘密の値を保存します。
B. サブスクリプションの範囲でキーコンテナー管理者ロールを割り当てます。
C. キーボールトシークレットユーザーロールをボールトスコープで付与します。
D. システム割り当ての管理IDを使用します。
正解: **C,D (**コメントを発表する**)**
詳細説明 :APIはシステム割り当ての管理IDを使用するため、アプリケーション認証情報は開発チームによって保存またはローテーションされません。そのIDには、シークレット値を読み取るために必要な権限のみが付与されます。つまり、キーコンテナースコープのKey Vault Secrets Userの権限は、サブスクリプションスコープのKey Vault Administratorの権限よりも大幅に限定されます。アプリ構成にシークレット値を保存すると、シークレットを保護するという要件に違反し、リスクが増大します。したがって、修正後のペアは、認証情報不要の認証と最小権限の認可を、限定された影響範囲で実現します。
学習ガイドの関連性 :セキュリティと運用 :Key Vault、アプリ構成、マネージドID、OpenTelemetry、Azure Monitor、およびKQLベースのトラブルシューティング。
Microsoft Learn 公式リファレンス: AI-200 学習ガイド | Azure Key Vault RBAC ガイド | Azure リソースのマネージド ID

質問: **74**
あなたは、Azure Container Apps を使用するマイクロサービスベースのアプリケーションを開発しています。このアプリケーションは、注文処理、在庫管理、レポート生成などのタスクを処理する、複数のコンテナ化されたサービスで構成されています。
コンテナアプリは安全に保護する必要があります。すべてのアプリは同じ仮想ネットワーク内に存在し、同じDapr構成を共有し、同じログ記録場所を共有する必要があります。
アプリケーションは、コンテナに利用可能なメモリ量とコンピューティングリソース量の設定をサポートする必要があります。
Azure Container App を構成する必要があります
CLIコマンドはどのように入力すればよいですか？回答するには、回答欄で適切なオプションを選択してください。
注：正解ごとに1ポイントが加算されます。



正解:



Explanation:



検証済みの回答: `az containerapp env create ... --enable-workload-profiles`。

詳細説明 :コンテナアプリの共有境界は、管理対象環境です。1つの環境内のアプリは、その環境の仮想ネットワーク統合、可観測性宛先、およびDapr関連の環境構成を共有できます。ワークロードプロファイルを有効にすると、その環境内のワークロードに対して選択可能なコンピューティングプロファイルとリソースサイジングオプションが提供されます。そのため、CLIはコンテナアプリ環境を作成し、ワークロードプロファイルを有効にする必要があります。個々のコンテナアプリを単独で作成しても、シナリオで必要とされる共通の環境レベルのネットワーク、ログ記録、およびリソースプロファイル境界は確立されません。

学習ガイドの関連性 :コンテナ化された Azure ワークロード :レジストリの構築、App Service コンテナ、コンテナ アプリの改訂/スケーリング動作、および AKS デプロイの選択肢。

Microsoft Learn 公式資料: AI-200 学習ガイド | Azure Container Apps 環境

質問**: 75**

ある企業がAzure Cosmos DB for NoSQLアカウントを保有している。このアカウントはセッションの一貫性を保つように構成されている。

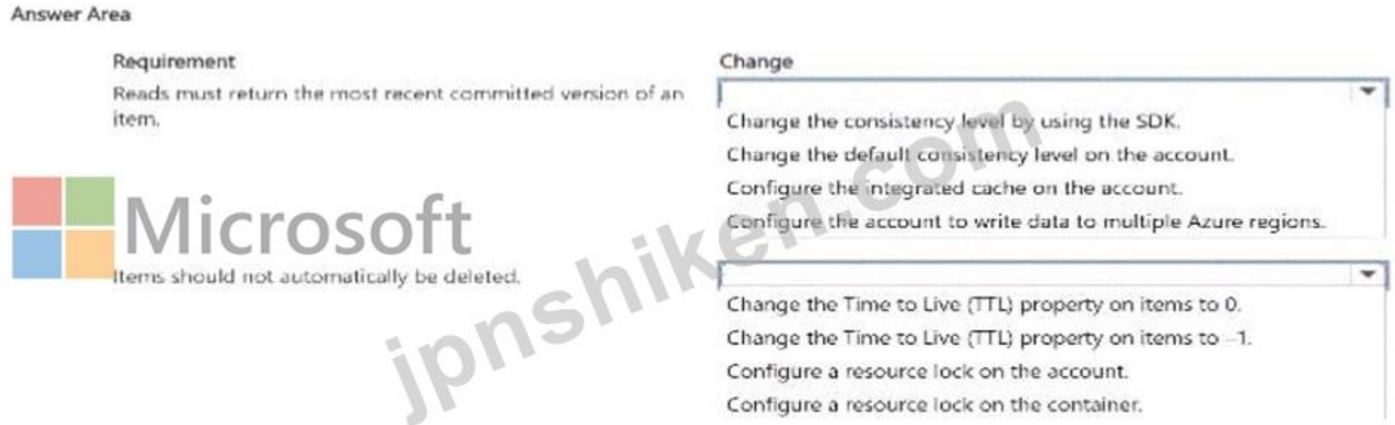
データは単一のAzureリージョンに書き込まれ、3つのAzureリージョンから読み取ることができます。

SDKを使用してAzure Cosmos DB for NoSQLコンテナデータにアクセスするアプリケーションには、以下の要件があります。

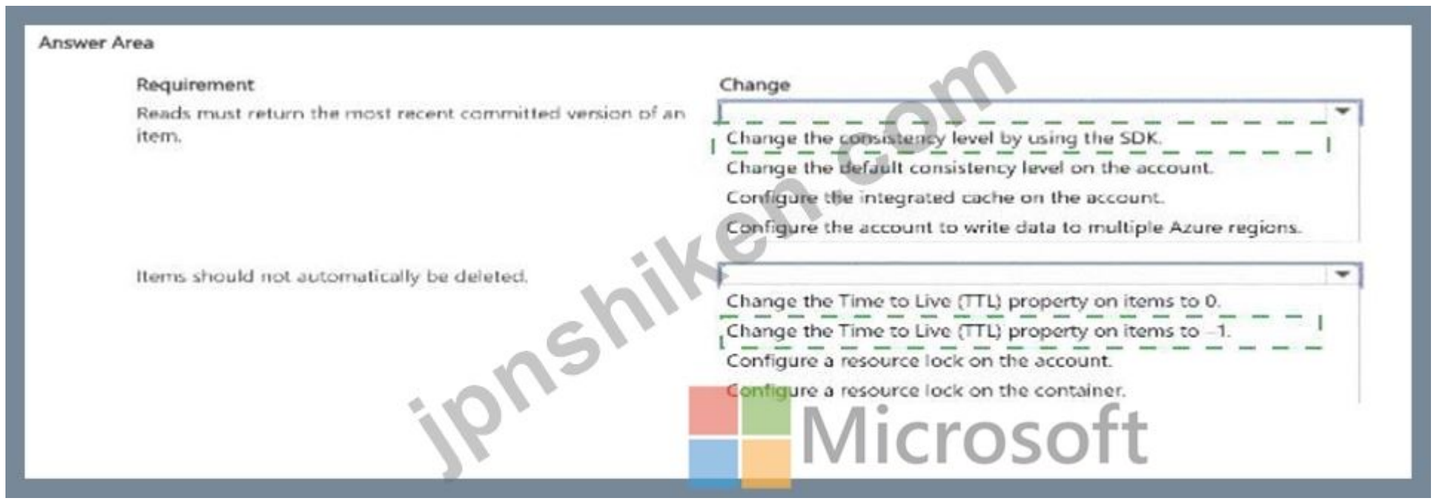
* アプリケーションからのロードは、任意の Azure リージョンからアイテムの最新のコミット済みバージョンを返す必要があります。

コンテナ内のアイテムは自動的に削除されるべきではありません。

Azure Cosmos DB for NoSQL アカウントに変更を適用する必要があります。どうすればよいですか？回答するには、回答欄で適切なオプションを選択してください。



正解:



Explanation:



質問: **76**

-

本番環境へのデプロイに向けてコンテナイメージを準備しています。

コンテナイメージの構築およびデプロイプロセスでは、以下の事項を確実に満たす必要があります。

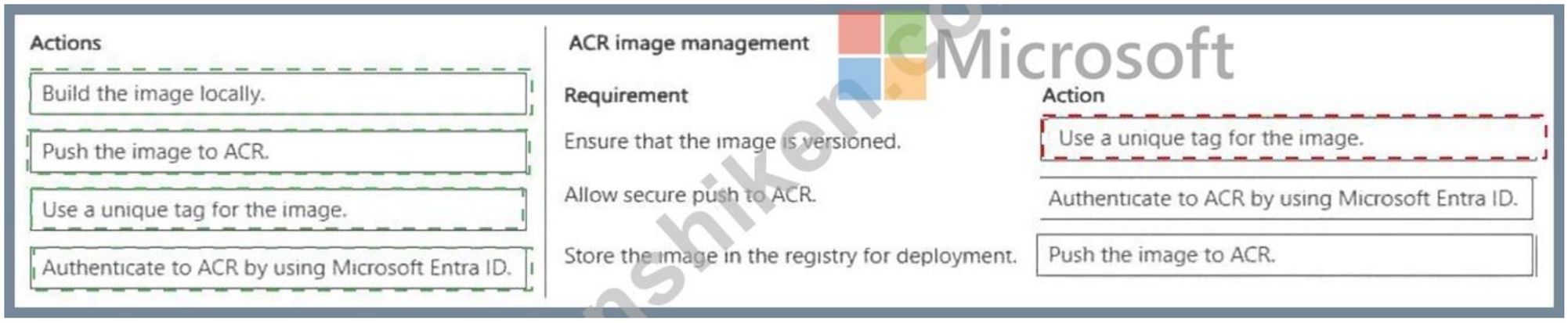
※この画像は固有のバージョンを持っています。

*Azure Container Registry (ACR) にイメージをプッシュする際には、セキュア認証が使用されます。

*イメージは展開のためにACRに保存されます。
コンテナイメージのビルドおよびプッシュプロセスが要件を満たしていることを確認する必要があります。
各要件に対して、どの操作を実行すべきでしょうか？回答するには、適切な操作を該当する要件に移動してください。各操作は、1回、複数回、またはまったく実行しない場合があります。コンテンツを表示するには、ペイン間の分割バーを移動したり、スクロールしたりする必要がある場合があります。
注：正解ごとに1ポイントが加算されます。



正解:

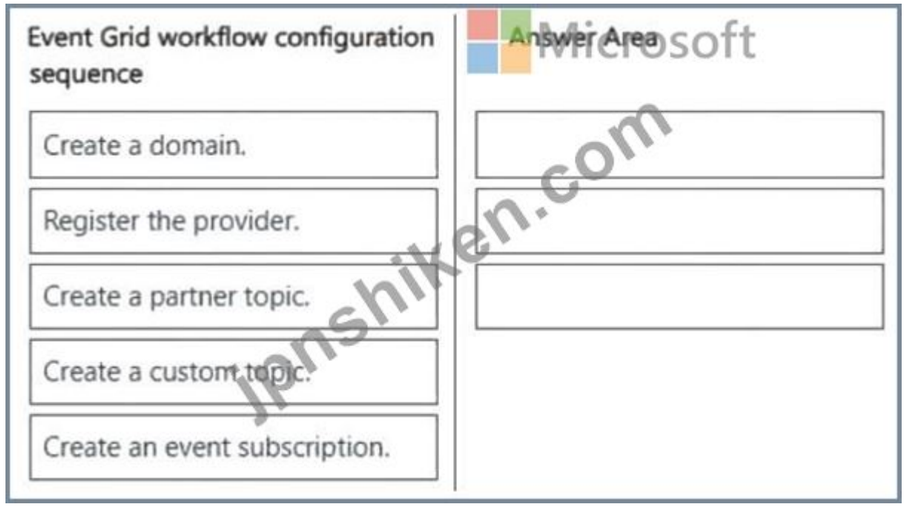


Explanation:
* 画像のバージョン管理を確実に行う：画像に一意のタグを使用する。
* ACRへのセキュアなプッシュを許可する :Microsoft Entra IDを使用してACRに認証します。
* 展開のためにイメージをレジストリに保存します: イメージを ACR にプッシュします。
本番環境へのデプロイでは、特定のビルドを識別して一貫してデプロイできるように、各イメージに一意のタグを付与する必要があります。Microsoft は、デプロイ時に一意のタグを使用することを推奨しています。これは、プッシュされた各イメージに、latest などの変更可能なタグを再利用するのではなく、固有のバージョン参照が付与されるためです。一般的な一意のタグ付け方式としては、ビルド ID、タイムスタンプ、ソース管理識別子などがあります。
イメージをプッシュする前に、Microsoft Entra ID を使用して Azure Container Registry に認証してください。Microsoft は、長期間有効なレジストリ管理者資格情報に依存するのではなく、az acr login などの Azure ID ベースの認証を使用することを推奨しています。また、適切な Azure RBAC/データプレーンのアクセス許可によって、ID がアーティファクトをプッシュできるようにする必要があります。
最後に、`docker push <registry>.azurecr.io/<repository>:<tag>`などのプッシュ操作を使用して、タグ付けされたイメージとそのレイヤーをACRにアップロードします。プッシュが正常に完了すると、イメージはレジストリに保存され、下流のデプロイサービスから参照できるようになります。
イメージをローカルでビルドする機能は、明示された要件には対応していません。一部のワークフローではビルドが必要になる場合がありますが、ビルド自体がACRにおける固有のバージョン管理、認証、またはストレージを提供するものではありません。

学習ガイドの参照: Azure Container Registry # イメージのタグ付けとバージョン管理、Microsoft Entra 認証、コンテナ イメージのプッシュ、リポジトリとアーティファクト。

有効的な**AI-200**問題集はJPNTest.com提供され、**AI-200**試験に合格することに役に立ちます！JPNTest.comは今最新**AI-200**試験問題集を提供します。JPNTest.com AI-200試験問題集はもう更新されました。ここで**AI-200**問題集のテストエンジンを手に入れます。最新版のアクセス、https://www.jpntest.com/shiken/AI-200-mondaishu　**144**問、３０%ディスカウント、特別な割引コード: **JPNshiken**」

質問: **77**
ドラッグアンドドロップ問題
新しくプロビジョニングされた Azure サブスクリプションがあります。AI 推論イベント用のカスタム Event Grid ワークフローを設計しています。
信頼性の高いイベントを下流のプロセッサにルーティングするには、イベントグリッドのコンポーネントを実装する必要があります。
どの3つの行動を順番に実行すべきでしょうか？回答するには、行動リストから適切な行動を回答欄に移動させ、正しい順序に並べ替えてください。



正解:



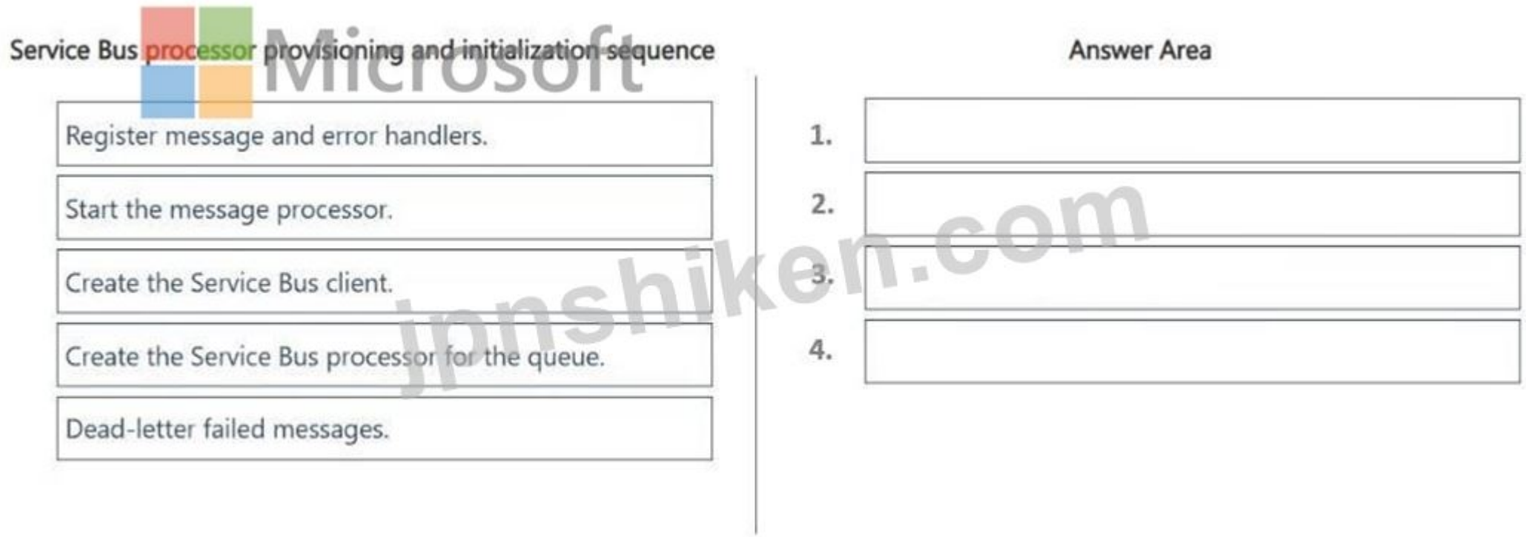
Explanation:
ステップ1 :プロバイダーを登録する
これは新しくプロビジョニングされた Azure サブスクリプションであるため、Event Grid コンポーネントをデプロイする前に、Microsoft.EventGrid リソース プロバイダーを登録する必要があります。

ステップ2 :カスタムトピックを作成する
アプリケーションから発生するAI推論イベントをルーティングするため、これらのイベントが送信されるエンドポイントとして機能するカスタムトピックを作成する必要があります。
ステップ3 :イベント購読を作成する
ルーティングルールを定義するには、イベントサブスクリプションを作成する必要があります。ここでフィルタリングを設定し、信頼性の高いイベントのみがダウンストリームプロセッサに送信されるようにします。
参照 :
https://www.c-sharpcorner.com/article/building-event-driven-ai-workflows-with-azure-event-grid-and-net/

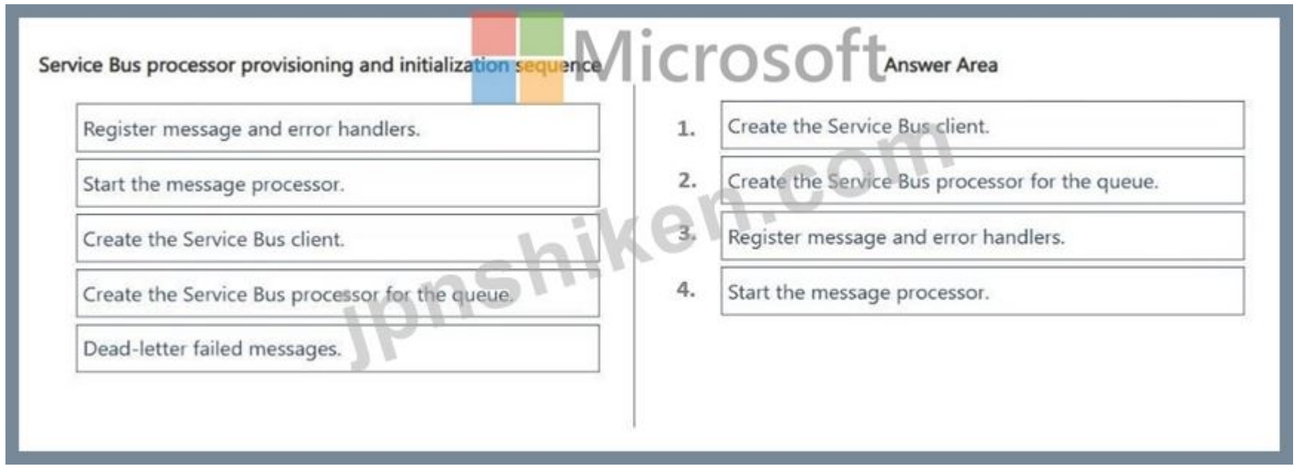
質問: **78**
サポートコール中に、顧客との会話を英語からスペイン語のテキストにリアルタイムで翻訳し、遅延を最小限に抑える必要があります。どのAzure AIサービスを使用すべきでしょうか？
A. Azure AI Translator (テキスト翻訳REST API)
B. Azure AI Speech（音声翻訳）
C. Azure AI 言語 (テキスト分析)
D. Azure AI ドキュメントインテリジェンス
正解: **B (**コメントを発表する**)**
Azure AI Speech の音声翻訳機能は、テキストのみの Translator サービスとは異なり、低遅延のライブオーディオシナリオ向けに特別に構築された単一のストリーミングパイプライン内で、リアルタイムの音声テキスト変換と翻訳を実行します。

質問: **79**
Azure App Service API アプリ（api1 という名前）にコンテナをデプロイする予定です。api1 のソースコードは GitHub リポジトリにホストされています。コンテナは実行時に API キーを使用してバックエンド サービスに接続します。
コンテナは、ソースリポジトリやGitコミット履歴にAPIキーを公開することなく、実行時にAPIキーを取得できる必要があります。APIキーはアプリケーション構成に直接保存してはなりません。APIキーはAzure Key Vaultから安全にアクセスできる必要があります。
API キーが Azure Key Vault に安全に保存され、ソース管理や Git コミット履歴に公開されることなく、実行時にコンテナから利用できる状態になっていることを確認する必要があります。
解決策 :Azureポータルで設定したApp Serviceアプリケーション設定としてAPIキーを保存します。
その解決策は目標を達成しているか？
A. いいえ
B. はい
正解: **(**正解を表示します**)**

質問: **80**
ドラッグアンドドロップ問題
AIバッチジョブ用のサービスバスプロセッサをプロビジョニングおよび構成しています。
プロセッサは既存のキューに接続し、メッセージおよびエラー処理のためのハンドラを登録した後、メッセージの受信を開始する必要があります。
メッセージおよびエラー処理のために、サービスバスプロセッサをプロビジョニングおよび構成する必要があります。
どの4つの行動を順番に実行すべきでしょうか？回答するには、行動リストから適切な行動を回答欄に移動させ、正しい順序に並べ替えてください。



正解:



Explanation:

ステップ1 :サービスバスクライアントを作成する

これにより、接続文字列またはAzure ID認証情報を使用して、Azure Service Bus名前空間への初期接続が確立されます。

ステップ2 :キュー用のサービスバスプロセッサを作成する

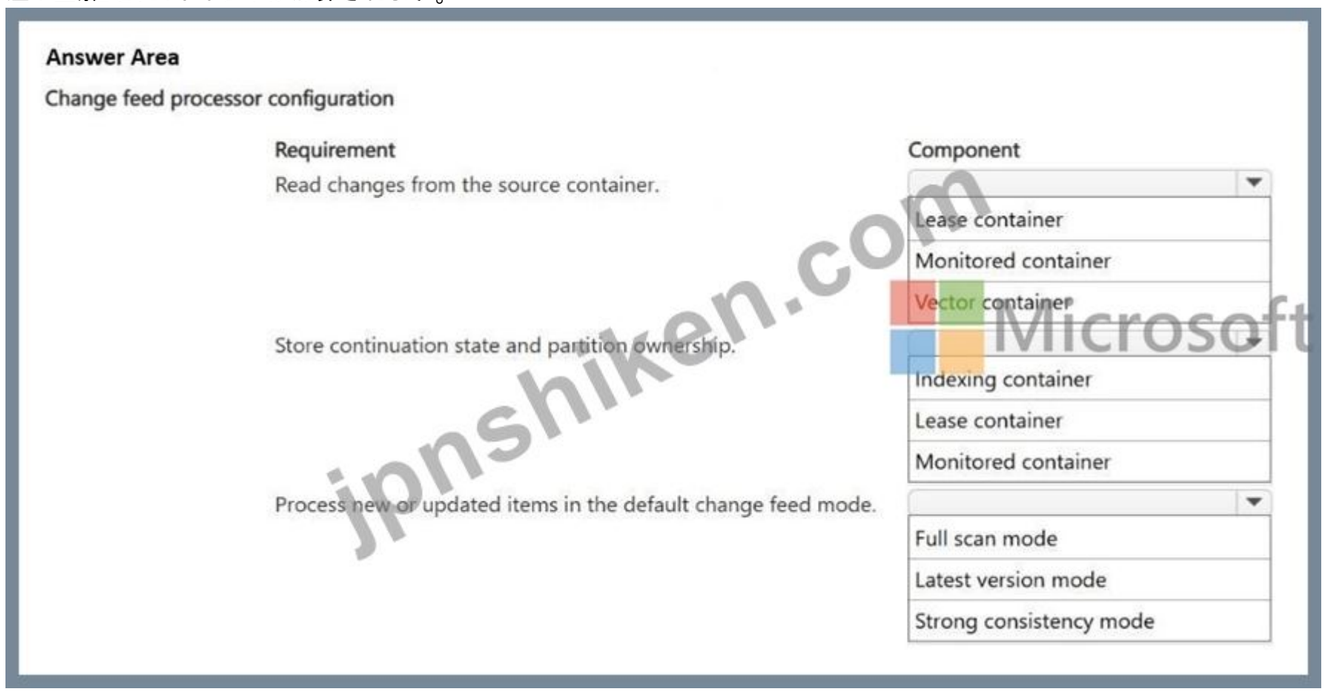
クライアントを取得したら、そのファクトリ メソッドを呼び出して、キューをターゲットとする専用のプロセッサをインスタンス化します。ステップ 3: メッセージ ハンドラとエラー ハンドラを登録します。プロセッサには、処理プロパティとエラー プロパティに割り当てられたイベント ハンドラが必要で、メッセージの到着時や例外発生時に何が起こるかを明示的に指示します。ステップ 4: メッセージ プロセッサを開始します。プロセッサは、受信を開始するメソッドを呼び出すまでアイドル状態のままです。このメソッドは、メッセージをリッスンして取得する内部ループを開始します。

参照 :

https://learn.microsoft.com/en-us/azure/service-bus-messaging/service-bus-dotnet-get-started-with-queues?tabs=passwordless

質問: **81**

ホットスポットに関する質問
あなたは、Azure Cosmos DB for NoSQL 内の新規または更新されたドキュメントに反応するイベント駆動型システムを開発しています。
アプリケーションは、変更フィードプロセッサを使用して、コンテナに新規または更新されたドキュメントが出現するたびに処理する必要があります。また、アプリケーションは実行間でプロセッサの状態を維持する必要があります。
ステートフルな変更フィード処理に必要なコンポーネントを設定する必要があります。
どのコンポーネントを設定する必要がありますか？回答するには、回答欄で適切なオプションを選択してください。
注：正解ごとに1ポイントが加算されます。



正解:

Answer Area

Change feed processor configuration

Requirement	Component
Read changes from the source container.	Lease container **Monitored container** Vector container
Store continuation state and partition ownership.	Indexing container **Lease container** Monitored container
Process new or updated items in the default change feed mode.	Full scan mode **Latest version mode** Strong consistency mode

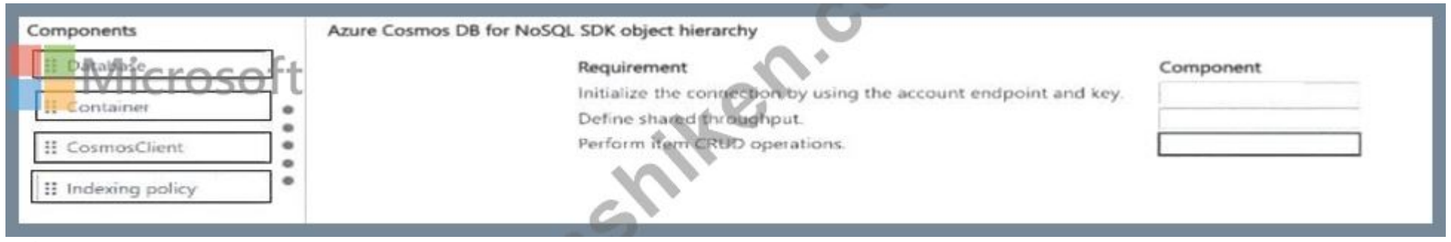
jpnshiken.com

Explanation:
ボックス1：監視対象コンテナ。
ソースから新規または更新されたドキュメントを読み取るには、監視対象コンテナを設定する必要があります。変更フィードプロセッサは、このコンテナの変更を監視し、変更されたドキュメントのソート済みリストを順次取得します。
ボックス2 :リースコンテナ
リースコンテナは、状態ストレージとして機能する独立したコンテナです。複数のワーカーインスタンスにわたって処理チェックポイントを保持するため、システムは中断したところから処理を再開できます。
ボックス3：最新バージョンモード
新規または更新された文書を処理するには、変更フィードの最新バージョンモードを使用する必要があります。
最新バージョンモードは、デフォルトの変更フィードモードです。このモードでは、変更された各項目（挿入および更新）の最新バージョンのみが、変更された順序どおりに提供されます。項目が短時間に複数回更新された場合、プロセッサは最新の更新後の最終状態のみを受け取ります。
参照：
https://learn.microsoft.com/en-us/azure/cosmos-db/change-feed-processor

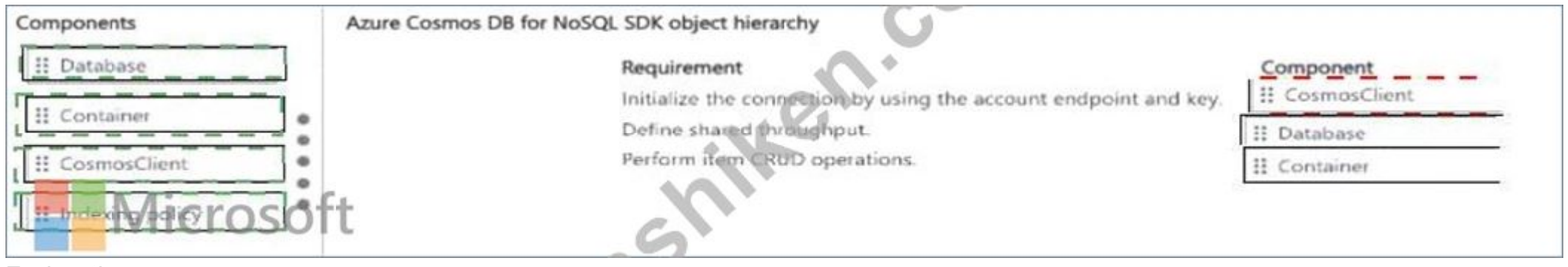
質問**: 82**
あなたは、セマンティック検索用のAzure Database for PostgreSQLテーブルを設計しています。クエリでは、created_at列に基づいて頻繁にフィルタリングが行われます。
スキーマは、ベクトル類似性検索と信頼性の高い日付フィルタリングをサポートする必要があります。
テーブルが要件を満たしていることを確認する必要があります。
どの2つの行動をとるべきでしょうか？それぞれの正解は、解決策の一部を示しています。

注：正解ごとに1ポイントが加算されます。
A. created_at には型付きタイムスタンプ列を使用します。
B. created_at を自由形式の文字列として保存します。
C. 埋め込みデータをvarchar型の列に格納します。
D. 埋め込みをpgベクトル列に格納します。
正解: **A,D (**コメントを発表する**)**

質問: **83**
あなたは、アプリケーションデータを保存するためにAzure Cosmos DB for NoSQLを使用する.NETアプリケーションを開発しています。このアプリケーションは、Azure Cosmos DB for NoSQL SDKを使用してデータベースアカウントとやり取りします。アプリケーションは次のタスクを実行する必要があります。
アカウントのエンドポイントとキーを使用して接続を初期化します。
* 共有スループットを定義します。
コンテナに格納されているアイテムに対して、作成、読み取り、更新、削除（CRUD）操作を実行します。



正解:



Explanation:



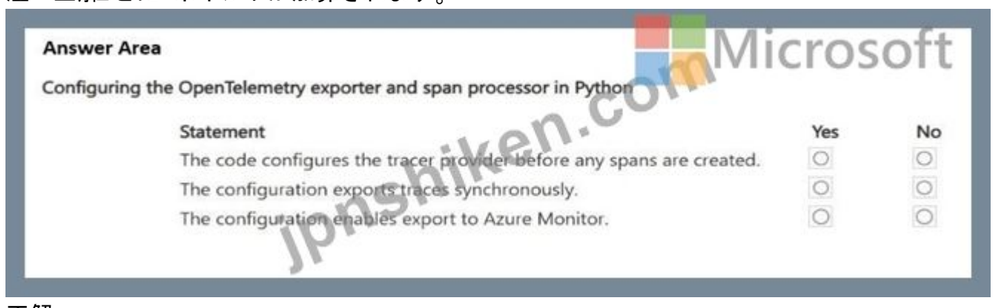
検証済み回答: 接続の初期化: CosmosClient。共有スループットの定義: Database。アイテムの CRUD の実行: Container。
詳細説明 :Cosmos DB SDKは、サービスリソース階層に従います。CosmosClientはアカウント接続を表し、アカウントエンドポイントと認証情報で初期化されます。複数のコンテナがデータベーススループットを共有する場合、共有スループットはデータベースレベルで構成されます。アイテムの作成、読み取り、更新、削除操作は、コンテナオブジェクトを介して実行されます。これは、コンテナがアイテムを保持し、パーティショニング/インデックス作成動作を定義するためです。インデックス作成ポリシーは、CRUDに使用されるSDKオブジェクトではなく、コンテナに添付される構成です。

学習ガイドの関連性 :AIデータ管理ワークロード :Cosmos DB、PostgreSQL、キャッシング、ベクトルストレージ、ベクトル検索、一貫性、および接続最適化。
Microsoft Learn 公式資料: AI-200 学習ガイド | Cosmos DB データベース、コンテナ、およびアイテム

質問: **84**
ホットスポットに関する質問
あなたは、分散トレースをAzure Monitorに送信する必要のあるアプリケーションのPythonトレース構成を確認しています。
以下のコードは、OpenTelemetryのトレースを設定します。

```
01 from opentelemetry import trace
02 from opentelemetry.sdk.trace import TracerProvider
03 from opentelemetry.sdk.trace.export import BatchSpanProcessor
04 from azure.monitor.opentelemetry.exporter import AzureMonitorTraceExporter
05
06 trace.set_tracer_provider(TracerProvider())
07 tracer = trace.get_tracer(__name__)
08
09 exporter = AzureMonitorTraceExporter(connection_string="<conn>")
10 span_processor = BatchSpanProcessor(exporter)
11 trace.get_tracer_provider().add_span_processor(span_processor)
```

以下の各記述について、正しい場合は「はい」を選択してください。そうでない場合は「いいえ」を選択してください。
注：正解ごとに1ポイントが加算されます。



正解:



Explanation:
ボックス1 :はい
はい、このコードはスパンが作成される前にトレーサープロバイダを設定します。

トレーサープロバイダの初期化シーケンスは構造的には正しいものの、プロセッサとエクスポーターが実行フローのかなり遅い段階でアタッチされるため、トレーサーインスタンスが取得された時点ではまだスパンをキャプチャできません。
ボックス2 :いいえ
いいえ、この設定ではトレースは同期的にエクスポートされません。
このコードはBatchSpanProcessorを利用しており、完了したスパンをバッチ処理し、バックグラウンドスレッド上で非同期的にエクスポートすることで、テレメトリ収集がメインアプリケーションコードをブロックするのを防ぎます。
ボックス3 :はい
はい、この構成により、分散トレースをAzure Monitorに正常にエクスポートできます。
提供されたコードは、ネイティブのOpenTelemetry SDKコンポーネントを公式のAzure Monitorエクスポーターライブラリと正しく初期化および接続します。
参照：
https://learn.microsoft.com/en-us/python/api/overview/azure/monitor-opentelemetry-exporter-readme

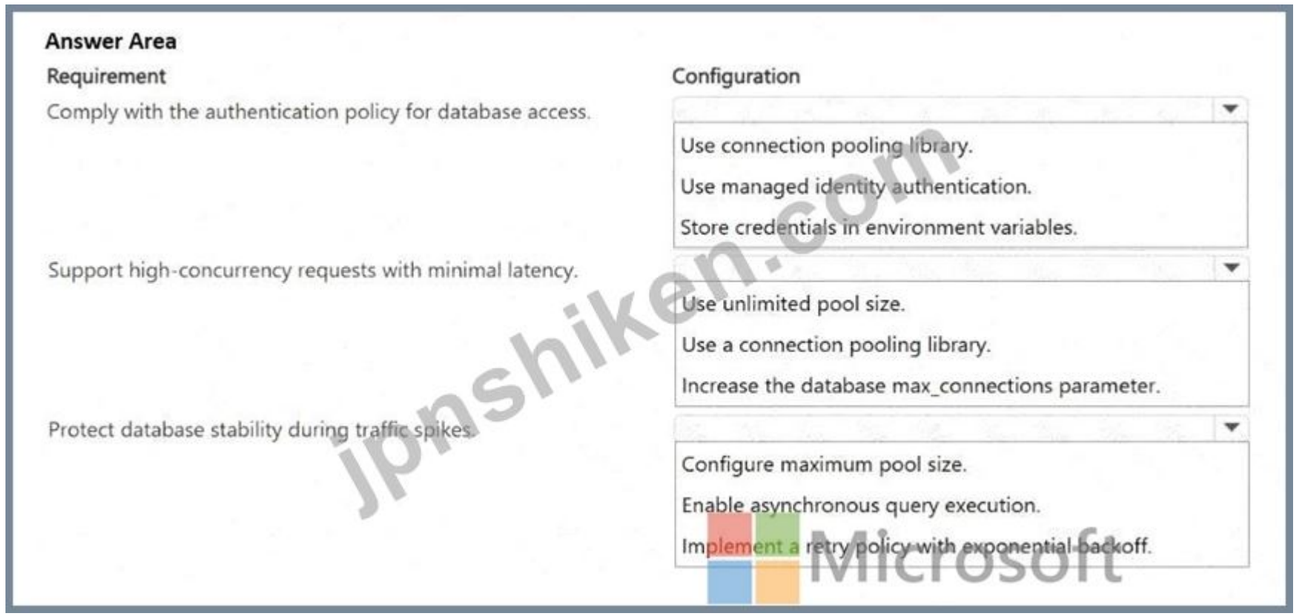
質問: **85**
Azure Storage Blobへのアップロードは、バックエンド処理をトリガーする必要があります。
解決策は以下の条件を満たす必要があります。
世論調査は避ける。
遅延を最小限に抑える。
アップロードが発生した際に自動的にトリガーされます。
ファイルがアップロードされたらすぐに処理が開始されるようにする必要があります。
どのトリガーを実装すべきでしょうか？
A. タイマー
B. イベントグリッド
C. キューストレージ
D. HTTP
正解: **B** (コメントを発表する)
Azure Event Gridは、最小限の遅延でリアルタイムのイベント通知をプッシュし、ポーリングを完全に回避できるため、最適なトリガータイプです。
プッシュ型モデル :ブロブが作成されると同時にイベントを即座に送信します。
最低遅延：通知をミリ秒単位で配信します。
ポーリングによるオーバーヘッドがない :タイマーやキューによるポーリングと比較して、リソースを節約できます。
仕組み
ユーザーがファイルを Azure Blob Storage にアップロードします。Blob Storage は Microsoft.Storage.BlobCreated イベントを発生させます。
Event Grid はこのイベントをバックエンドのエンドポイント (Azure Function や Webhook など) に直接プッシュします。処理はすぐに開始されます。
参照：
https://damle-amit075.medium.com/azure-databricks-auto-loader-plugging-the-gap-for-batch-loads-72585b0f158e

質問: **86**
あなたは、セマンティック検索結果をRedisにキャッシュするAI検索APIを開発しています。
検索結果は10分間キャッシュに保持する必要があります。基となるデータが変更された場合、キャッシュされたエントリは返してはなりません。
データの一貫性を確保するためには、キャッシュアサイド戦略を実装する必要があります。
どの2つの行動を実行すべきですか？それぞれの正解は解決策の一部を示しています。2つ選択してください。注：正解ごとに1ポイントが加算されます。

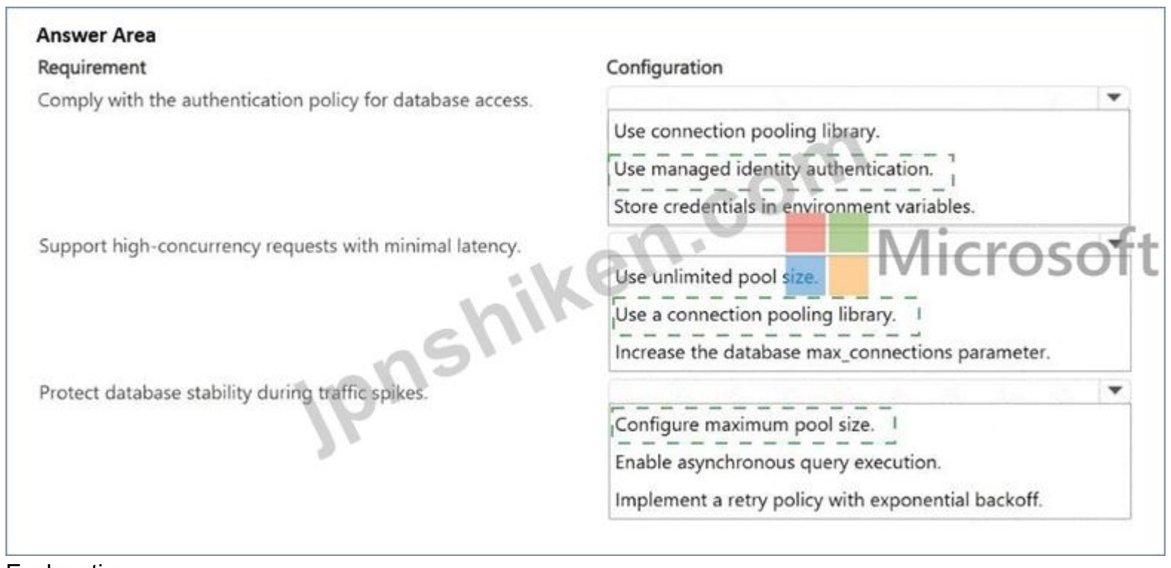
A. 各キーに10分間の有効期限を設定します。
B. キーアクセスに基づいてスライディング有効期限を実装します。
C. キースペースイベントのキャッシュ通知を設定します。
D. ソースデータが変更された際に、関連するキャッシュキーを削除します。
正解: **A,D** (コメントを発表する)
詳細説明 :キャッシュアサイド設計では、有効期間の制限と明示的な無効化の両方が必要です。10分間のTTLにより、キャッシュされた検索結果が永久に保持されることを防ぎ、ソースデータが変更されたときに関連キーを削除することで、10分間の間に既知の古いエントリが返されるのを防ぎます。スライディング有効期限は、アクセスに基づいて有効期間を延長し、古いデータを保持することができます。キースペース通知はRedisキーイベントを報告しますが、権威あるソースが変更されたときにエントリを無効化するアプリケーションロジックを置き換えるものではありません。
学習ガイドの関連性 :AIデータ管理ワークロード :Cosmos DB、PostgreSQL、キャッシング、ベクトルストレージ、ベクトル検索、一貫性、および接続最適化。
Microsoft Learn 公式資料: AI-200 学習ガイド | Azure Managed Redis ドキュメント

質問: **87**
あなたは、ユーザーがHTTPエンドポイント経由で送信した大容量画像のアップロードを処理するAzure Functionsアプリを設計しています。
解決策は以下の条件を満たす必要があります。
* 画像処理を最初のアップロード要求から切り離すことで、クライアントのタイムアウトを防ぎます。
* 処理失敗時の自動再試行動作をサポートします。
* バックグラウンド処理を、受信するHTTPアップロードの速度とは独立してスケーリングします。
拡張性と信頼性に優れた非同期処理ソリューションを設計する必要があります。
どの2つの行動を実行すべきでしょうか？それぞれの正解は、解決策の一部を示しています。
注：正解ごとに1ポイントが加算されます。
A. 画像処理のためのキュートリガー関数を実装する。
B. ストレージキューで再試行ポリシーを設定します。
C. HTTPトリガー関数内で画像を処理します。
D. アップロードされたメタデータをAzureストレージに永続化します。
正解: (正解を表示します)
正しいアーキテクチャでは、HTTP 取り込みパスと計算負荷の高い画像処理ワークロードの間に、非同期バッファとして Azure Storage キューを使用します。HTTP トリガー関数は、アップロードを受け付け、作業メッセージをキューに追加し、画像処理の完了を待つことなく速やかに処理を再開する必要があります。Microsoft は、HTTP トリガーの Azure Function では、基となる関数が実行を継続していても Azure Load Balancer の応答タイムアウトが発生する可能性があるため、長時間実行される HTTP 操作には非同期パターンを特に推奨しています。
キューによってトリガーされる関数は、画像処理を独立して実行します。Azure Functions は、キューに格納されたワークロードに基づいて Azure Queue Storage のトリガーにターゲットベースのスケーリングを適用するため、ワーカーの容量は HTTP 到着処理に直接依存するのではなく、バックログの増加に応じて拡張できます。
信頼性を高めるため、Azure キュー トリガー関数は再試行動作を提供します。visibilityTimeout 設定は、失敗したメッセージが再試行可能になるまでの遅延を制御し、maxDequeueCount は、メッセージが < queue-name > - ポイズン キューに転送されるまでに実行される処理試行回数を制御します。既定の最大試行回数は 5 回です。
Cは非同期要件を満たしていません。Dはアップロード状態の追跡には役立ちますが、ワークロードの分離、再試行処理、キュー駆動型スケーリングといった機能は提供しません。
学習ガイドの参照: Azure Functions # キュー ストレージのトリガーとバインディング、イベント駆動型スケーリング、再試行
/poisonキューの動作。非同期HTTP処理。

質問: **88**
コンテナ化されたレコメンデーションAPIから、安全なデータベース接続を最適化する必要があります。
アプリケーションはどのように設定すればよいですか？



正解:



Explanation:

* データベースアクセスに関する認証ポリシーを遵守する :マネージドID認証を使用する。
* 最小限の遅延で高同時実行リクエストをサポートする：接続プーリングライブラリを使用します。
* トラフィック急増時のデータベースの安定性を保護する：最大プールサイズを設定します。

推奨事項APIでは、Fabrikamが長期有効な資格情報を使用しないサービスとデータベース間の認証を明示的に要求しているため、マネージドID認証を使用する必要があります。Azure Container Appsのマネージドを使用すると、アプリケーションは実行時にMicrosoft Entraトークンを取得し、アプリケーション構成にユーザー名、パスワード、接続シークレットを保存することなく、サポートされているAzureリソースにアクセスできます。

高負荷なPostgreSQLアクセスを実現するには、接続プーリングライブラリを実装してください。Azure Database for PostgreSQLのドキュメントでは、新しいデータベース接続を繰り返し開くとバックエンドプロセスが生成され、CPUとメモリを消費するため、接続プーリングを推奨しています。既存の接続を再利用することで、接続確立のオーバーヘッドが削減され、レイテンシとスループットが向上します。Azureでは、Flexible Serverの組み込みプーリングオプションとしてPgBouncerも提供されています。
急激なトラフィック増加時にPostgreSQLを保護するには、最大プールサイズを設定してください。無制限のプールでは、アプリケーションの要求に応じて過剰な同時接続が発生し、データベースメモリが枯渇したり、競合が増加したりする可能性があります。Microsoftは、接続数の増加はパフォーマンスの大幅な低下につながる可能性があると警告しており、プーリングと組み合わせた保守的な接続制限を推奨しています。
したがって、最適な構成は、マネージドID、接続プーリング、および制限付きプールサイズを組み合わせたものである。

。
学習ガイドの参照項目 :Azure Container Apps のマネージド ID、Azure Database for PostgreSQL の接続管理、PgBouncer、接続プールのサイズ設定とリソース保護。

質問: **89**
アプリケーションは、アップロードされた製品画像を、貴社固有の40種類のカスタムカテゴリ（例：独自部品番号）のいずれかに分類する必要があります。どのカテゴリを使用すればよいでしょうか？
A. Azure AI Visionの構築済み画像解析
B. Azure AIカスタムビジョン（画像分類）
C. Azure AI ドキュメント インテリジェンス
D. Azure AI Face API
正解: **(正解を表示します)**
事前に構築されたVisionモデルは、一般的な物体やシーンは認識しますが、ビジネス固有のカテゴリは認識しません。
Custom Visionを使用すると、独自のラベル付き画像を使用して、独自の部品番号などのドメイン固有のカテゴリに分類器をトレーニングできます。

質問: **90**
ケーススタディ2 - Proseware Inc.
背景
Proseware Inc.は、企業顧客向けにAIを活用した知識管理ソリューションを開発しています。
同社は、セマンティック検索、インテリジェントな文書検索、およびリアルタイムのパートナー連携をサポートするために、プラットフォームの近代化を進めている。
エンジニアリングチームはPythonとAzure SDKを使用しています。アーキテクチャは、コンテナ化されたマイクロサービス、ベクトル検索ワークロード、およびサーバーレスバックエンド処理をサポートするように再設計されています。
計画されたアプリケーションアーキテクチャ
マイクロサービスはDockerを使用してコンテナ化されます。
コンテナ化されたマイクロサービスとAzure Functionsアプリのコードはローカルで開発されますが、GitHubリポジトリに保存されます。
コンテナ化されたマイクロサービス用のカスタムイメージは、Azure Container Registry (ACR) に保存されます。
ベースイメージはDocker Hubに保存されます。カスタムイメージは、ベースイメージが更新されるたびに自動的に再構築される必要があります。
Azure Cosmos DB for NoSQL は、ドキュメント、メタデータ、およびベクトル埋め込みを保存します。
Azure Functions は、NoSQL でホストされているドキュメントの Azure Cosmos DB のベクトル埋め込みを生成し、Service Bus にメッセージを送信して検索インデックスの更新をトリガーします。
Azure Container Apps (ACA) アプリは、NoSQL ドキュメント用の Azure Cosmos DB 全体にわたるセマンティック検索を提供するバックエンド API サービスをホストします。API サービスは、Service Bus メッセージを処理し、検索インデックスを更新します。
Azure Kubernetes Service (AKS) は、既存の Azure Cosmos DB for NoSQL ドキュメントに対して、ベクトル埋め込みの再生成をバッチ処理します (埋め込みモデルが変更されるたびに)。
エクストラネットに接続されたコンテナ化されたWebhookにより、ビジネスパートナーは文書を送信して、内部のAIワークフローで処理し、意味検索や取得を行うことができる。
監視
Azure リソースによって生成されたテレメトリは、Azure Monitor に送信されます。
Log Analyticsワークスペースは、ACAアプリのログ、AKSコンテナのログ、およびAzure Functionsアプリのログを収集するために使用されます。

Azure Functionsの監視は現在、Azure Application Insights SDKの計測機能を使用して実装されています。
ビジネス要件
新規または更新されたAzure Cosmos DB for NoSQLでホストされるドキュメントの埋め込みは、自動的に生成される必要があります。
バックエンドAPIサービスは、営業時間中に自動的にスケールする必要があります。
バックエンドAPIのコールドスタート時の遅延は最小限に抑える必要がある。
機密情報はコンテナイメージの外に保存する必要があります。
開発者は、Azure Functions ホストとアプリ間でテレメトリを関連付けることができなければなりません。
すべてのトレースは、OpenTelemetry SDKの計測機能を使用して実装する必要があります。
開発努力は最小限に抑えなければならない。
技術要件
コンテナイメージは、コードの更新がメインブランチにマージされる前に、自動的に構築され、検証される必要があります。
イメージのビルド自動化はAzure Container Registry内で実行される必要があり、ローカルの開発者マシンや外部のビルドサービスへの依存を排除する必要があります。
イメージビルドが開発者のローカルマシンに依存する状態は解消されなければならない。
ACAにおけるイベント駆動型スケーリングは、Azure Service Busキュー内の保留中のメッセージ数に基づいて行われる必要があります。
Azure Cosmos DB for NoSQL の RU 消費量を最小限に抑える必要があります。
ベクトル類似性検索では、Azure Cosmos DB for NoSQLに保存されている埋め込みデータを使用する必要があります。
パートナー向けのコンテナ化されたWebhookサービスは、Azure App Service上で実行する必要があります。
機密情報は、コンテナイメージ、ソースコード管理システム、またはアプリケーション構成に直接保存してはなりません。実行時に安全にアクセスできる必要があります。
すべてのシークレットはAzure Key Vaultに一元的に保存され、実行時にはマネージドIDを通じてアクセスされる必要があります。
Azure App Service は、外部サービスに依存することなく、実行時にシークレットを提供する必要があります。
リソースとワークロードは、自動化されたバージョン管理パイプラインを通じて、Bicepテンプレートを使用してデプロイする必要があります。再現性と監査可能性を確保するため、ローカルおよびコマンドラインによるデプロイは排除する必要があります。
既知の問題
ベクトル類似性クエリ実行中にRU消費量が急増する。
ベクトル類似性クエリに起因する既知の問題に対処する必要があります。
どの2つの行動をとるべきでしょうか？それぞれの正解は、解決策の一部を示しています。
注：正解ごとに1ポイントが加算されます。
A. コンテナのベクトルフィールドとメタデータプロパティに複合インデックスを追加します。
B. ベクトル場のインデックス精度を変更します。
C. アカウントの一貫性レベルを 強」に設定します。
D. ベクトルインデックスタイプをOatからquantizedFlatまたはdiskANNに変更します。
正解: **(**正解を表示します**)**
既知の問題 :ベクトル類似性クエリ実行中にRU消費量が急増する。
Azure Cosmos DB でベクトル類似性クエリを実行する際に発生する RU (リクエスト ユニット) 消費量の急増に対処するには、次の 2 つの手順を実行する必要があります。
ベクトル場のインデックス精度を変更する
ベクトルインデックスタイプをflatからquantizedFlatまたはdiskANNに変更します。
インデックス精度の調整 :ベクトル量子化などのパラメータ（quantizedByteSizeの設定など）を変更すると、格納されるベクトル要素のサイズと精度が変わります。精度を下げるとメモリ使用量とインデックスサイズが小さくなり、検索速度が向上し、RUの消費量も大幅に削減されますが、再現精度は若干低下します。

インデックスタイプの変更：標準のフラットインデックスは、すべてのドキュメントに対して総当たりのk近傍法（kNN）による完全検索を実行します。これには膨大な計算オーバーヘッドが必要となり、データセットが大きくなるにつれてRU（リソース使用率）が急上昇します。quantizedFlatまたはdiskANNに移行すると、圧縮技術と高度なグラフ走査アルゴリズムを活用して近似近傍法（ANN）による検索を実行するため、クエリの遅延と計算コストが大幅に削減されます。
参照：
https://learn.microsoft.com/en-us/azure/cosmos-db/gen-ai/vector-search-performance-tips

質問: **91**
断続的な接続問題が発生している本番環境の Azure OpenAI サービスについてトラブルシューティングを行っています。すべてのテレメトリデータは Log Analytics ワークスペースに送信されます。
AppRequestsテーブルを分析するKQLクエリを作成する必要があります。このクエリは、過去30分以内に発生し、かつステータスが失敗であるリクエストを抽出する必要があります。
最終出力には、各操作ごとに発生した失敗の総数を表示する必要があります。
時間範囲をフィルタリングし、障害件数を集計するために必要なKQL演算子を選択する必要があります。
どの2つの演算子を使用すべきでしょうか？正解はそれぞれ解答の一部を示しています。2つ選択してください。
注：正解ごとに1ポイントが加算されます。
A. 要約
B. 異なる
C. プロジェクト
D. どこで
正解: **A,D** (コメントを発表する)

有効的な**AI-200**問題集はJPNTest.com提供され、**AI-200**試験に合格することに役に立ちます！JPNTest.comは今最新**AI-200**試験問題集を提供します。JPNTest.com AI-200試験問題集はもう更新されました。ここで**AI-200**問題集のテストエンジンを手に入れます。最新版のアクセス、https://www.jpntest.com/shiken/AI-200-mondaishu 「144問、３０%ディスカウント、特別な割引コード: **JPNshiken**」

質問: **92**
アプリケーションは会話履歴を保存し、複数ターンのチャットボットが以前のターンを参照できるようにしますが、コンテキストウィンドウには制限があります。トークン制限内で最も関連性の高い履歴を保持する必要があります。
何を実装すべきでしょうか？
A. 履歴はクライアント側のみに保存し、モデル側には決して送信しない。
B. 2ターン後にすべての履歴を切り捨てる
C. 過去の発言を要約し、最近の発言はそのまま保持する
D. 出力専用のmax_tokensを増やす
正解: (正解を表示します)

質問: **93**
あなたはチャットボット向けのセマンティック検索機能を構築しています。ドキュメントの埋め込みデータをRedisに保存します。
Redisに接続して埋め込み値を保存する以下のPythonコードを確認してください。



以下の各記述について、正しい場合は「はい」を選択してください。そうでない場合は「いいえ」を選択してください。

注：正解ごとに1ポイントが加算されます。



正解:



Explanation:

検証済みの回答：はい、いいえ、はい。

詳細な説明：`HSET`はバイナリ埋め込みを`doc:1`をキーとするRedisハッシュのフィールドとして格納するため、最初の記述は正しいです。このコードはRediSearch/Redisクエリエンジンのベクトルインデックスを作成したり、ベクトルフィールドスキーマを定義したりしません。単にバイトを格納するだけでは類似性検索は有効にならないため、2番目の記述は誤りです。

`EXPIRE doc:1 600` は 600 秒（10 分）の有効期間を設定するため、3 番目のステートメントが真になります。

学習ガイドの関連性：AIデータ管理ワークロード：Cosmos DB、PostgreSQL、キャッシング、ベクトルストレージ、ベクトル検索、一貫性、および接続最適化。

Microsoft Learn 公式リファレンス: AI-200 学習ガイド | Azure Managed Redis でのベクトル検索

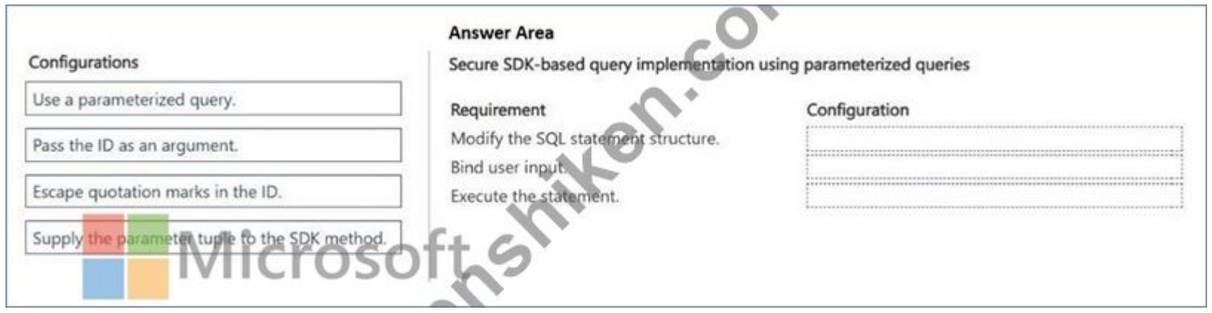
質問: **94**

ドラッグアンドドロップ問題

Python APIは、SQL文を使用してAzure Database for PostgreSQLからドキュメントを取得します。このAPIは、ユーザー入力からドキュメントIDを受け取ります。現在の実装では、ドキュメントIDをSQL文に直接挿入します。SQLインジェクションの可能性を最小限に抑えることで、SQL文の実行を安全にする必要があります。

現在の実装をどのように変更すればよいでしょうか？ 適切な設定を正しい要件に移動することで、この問いに答えることができます。各設定は、1回だけ使用することも、複数回使用することも、全く使用しないこともできます。コンテンツを表示するには、ペイン間の分割バーを移動したり、スクロールしたりする必要があるかもしれません。

注：正解ごとに1ポイントが加算されます。



正解:



Explanation:

ボックス1 :パラメータ化クエリを使用する

SQLインジェクションからSQLステートメントを保護するには、文字列の直接挿入をパラメータ化クエリ（プリペアドステートメントとも呼ばれる）に置き換える必要があります。

ボックス2 :IDを引数として渡す

現在の実装では、文書IDがSQL文に直接挿入されます。

ボックス3 :SDKメソッドにパラメータタプルを指定します

パラメータタプルの供給: ユーザー入力を別のタプルまたはリスト引数としてデータベースドライバの実行メソッドに渡します (例: cursor.execute(query, (document_id,)))。

参照：

https://www.stackhawk.com/blog/finding-and-fixing-sql-injection-vulnerabilities-in-flask-python/

質問: **95**

あなたはAzure上でサーバーレスJavaアプリケーションを開発しています。新しいAzure Functionsアプリケーションのシークレットを扱うために、新しいAzure Key Vaultを作成します。

申請は以下の要件を満たす必要があります。

Javaコードを変更することなく、Azure Key Vaultを参照できます。

受信するアプリケーションイベントの数に基づいて、Azure Functionsホストのインスタンスを動的に追加および削除します。

コールドスタートを避けるため、インスタンスは常にウォーム状態に保ってください。

VNetに接続します。

Azure Functions アプリケーションを削除する場合は、Azure Key Vault インスタンスへの認証情報も削除する必要があります。

Azure Functions アプリケーションに Azure Key Vault へのアクセス権を付与する必要があります。

どの3つの行動を順番に実行すべきでしょうか？回答するには、行動リストから適切な行動を回答欄に移動させ、正しい順序に並べ替えてください。

Actions

- Create a user-assigned managed identity for the application.
- Create the Azure Functions app with a Premium plan type.
- Create an access policy in Azure Key Vault for the application identity.
- Create an SSL certification in Azure Key Vault for the application identity.
- Create the Azure Functions app with an App Service plan type.
- Create the Azure Functions app with a Consumption plan type.
- Create a system-assigned managed identity for the application.

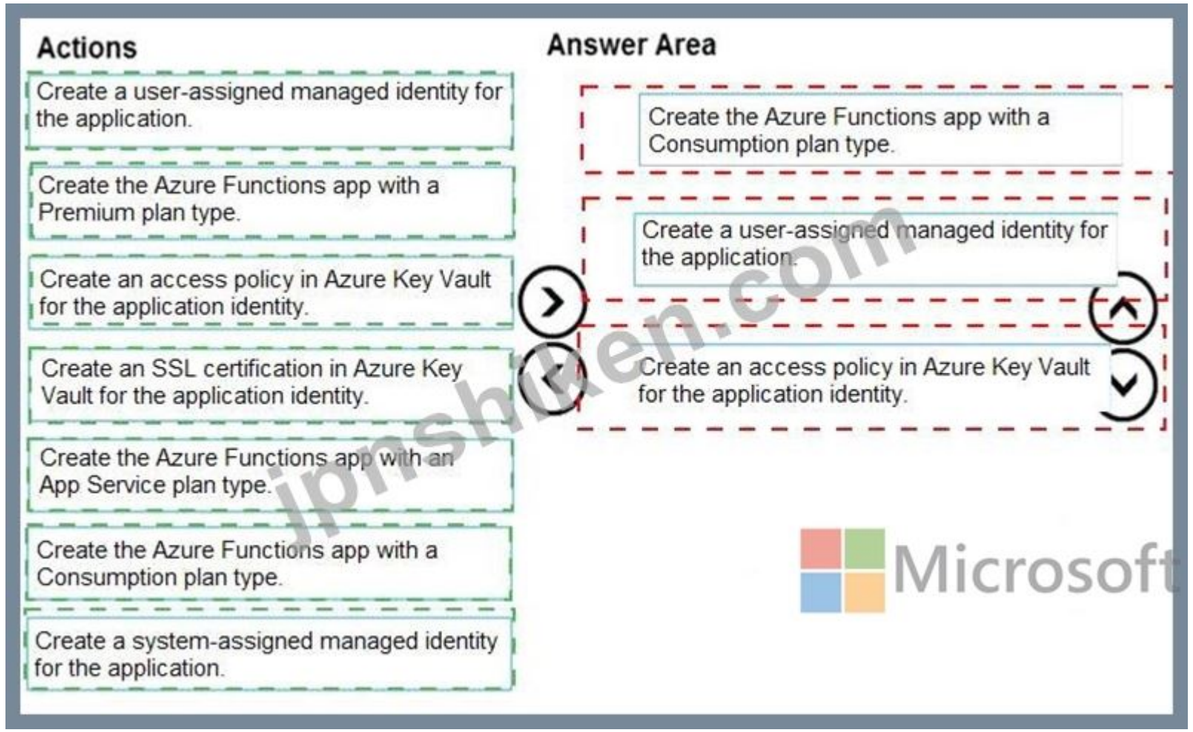
Answer Area





Microsoft
Ipnshiken.com

正解:



Explanation:



ステップ 1: 従量課金プランタイプの Azure Functions アプリを作成します。
サーバーレスには従量課金プランをご利用ください。
ステップ2 :アプリケーション用にシステム割り当て型の管理IDを作成します。
アプリケーション用に、システム割り当て型の管理IDを作成します。
Key Vault の参照は現在、システムによって割り当てられた管理対象 ID のみをサポートしています。ユーザーが割り当てた ID は使用できません。
ステップ3 :アプリケーションID用のアクセスポリシーをKey Vaultで作成します。
先ほど作成したアプリケーションIDに対して、Key Vaultでアクセス ポリシーを作成します。このポリシーで 「シークレットの取得」権限を有効にします。 「承認済みアプリケーション」またはapplicationIdの設定は、管理対象IDと互換性がないため、構成しないでください。
参照 :
https://docs.microsoft.com/en-us/azure/app-service/app-service-key-vault-references

質問: 96

Azure Container App をデプロイする予定です。
コンテナアプリがセッションアフィニティをサポートするように設定する必要があります。
コンテナアプリには、どのイングレスタイプとリビジョンモードを割り当てるべきですか？
A. TCPイングレスタイプとシングルリビジョンモード
B. TCPイングレスタイプと複数リビジョンモード
C. HTTPイングレスタイプと複数リビジョンモード
D. HTTPイングレスタイプとシングルリビジョンモード
正解: **(**正解を表示します**)**
Azure Container Apps のセッション アフィニティ (スティッキー セッションとも呼ばれる) は、同じクライアントからのリクエストを同じコンテナ アプリ レプリカにルーティングします。Microsoft のドキュメントによると、この機能は HTTP Cookie を使用して実装されています。
そのため、生のTCPイングレスではなく、HTTPイングレスが必要となります。
2つ目の要件は、シングルリビジョンモードです。Microsoftは、セッションアフィニティはコンテナアプリがシングルリビジョンモードでHTTPイングレスが有効になっている場合にのみサポートされると明言しています。シングルリビジョンモードでは、すべてのトラフィックが最新のアクティブなリビジョンにルーティングされるため、イングレスレイヤーはクライアントをそのリビジョン内の特定のレプリカに一貫して関連付けることができます。
複数リビジョンモードは、トラフィック分割、ブルーグリーンデプロイメント、A/Bテストなど、受信トラフィックを複数のアクティブなリビジョンに分散できるシナリオ向けに設計されています。このモデルは、ドキュメントに記載されているセッションアフィニティの制限と互換性がありません。また、スティッキーセッションの動作は一般的なTCP接続ルーティングではなくHTTP Cookieアフィニティに依存するため、TCPイングレスも正しくありません。
したがって、必要な構成は、シングルリビジョンモードのHTTPイングレスです。
学習ガイドの参照: Azure Container Apps # イングレス; スティッキーセッション/セッションアフィニティ; リビジョンモード; HTTP ルーティング。

質問: **97**
あなたは、新しい Azure Kubernetes Service クラスターにデプロイするための複数のマイクロサービスを開発しています。これらのマイクロサービスは、Azure Cosmos DB と Azure Blob Storage に保存されているデータを管理します。データは、Azure Key Vault に保存されている顧客管理キーを使用して保護されます。
すべての Azure Key Vault キーについて、キーのローテーションを自動化し、手動でのキーローテーションも許可する必要があります。キーは 3 か月ごとにローテーションする必要があります。キーの有効期限が切れる前に、有効期限切れの通知を送信する必要があります。
キーローテーションを設定し、キーの有効期限切れ通知を有効にする必要があります。
どの2つの行動をとるべきでしょうか？それぞれの正解は、解決策の一部を示しています。
注：正解ごとに1ポイントが加算されます。
A. 鍵作成時に鍵ローテーションポリシーを作成および構成します。
B. Azure Key Vaultのアラートを設定します。
C. Azure Key Vault アクセス ポリシーを作成して割り当てます
D. 新しいAzure Event Gridインスタンスを作成して構成する
正解: **(**正解を表示します**)**

質問: **98**
コストが急増するのを避けるため、Azure OpenAI を利用したアプリケーションでは、すべてのユーザーにおいて、定義されたトークン/分予算を超えないようにする必要があります。
何を設定すればよいですか？
A. コンテンツフィルタリングの厳格度レベル
B. Azure AI Foundry におけるデプロイメントレベルのレート制限 (TPM/RPM)
C. Azure AI Search におけるベクトルインデックスのサイズ制限

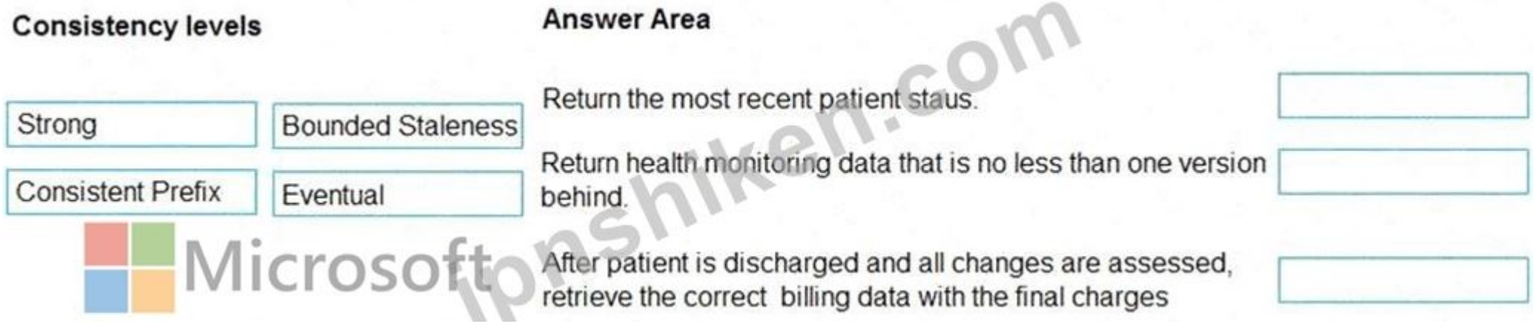
D. モデル温度
正解: (正解を表示します)
Azure OpenAI モデルのデプロイメントに対して、デプロイメントレベルのトークン/分 (TPM) およびリクエスト/分 (RPM) レート制限を設定します。Azure OpenAI のクォータは、個々のモデルデプロイメントに TPM 単位で割り当てられ、Microsoft のドキュメントによると、デプロイメントに割り当てられた TPM は、そのデプロイメントに対する推論リクエストに適用されるレート制限に直接対応します。また、モデルの TPM と RPM の比率に応じて、関連する RPM 制限も適用されます。
すべてのアプリケーションユーザーが同じデプロイメントを呼び出すため、この制限は各エンドユーザーごとに個別に適用されるのではなく、デプロイメントの境界で適用されます。リクエストトラフィックが構成済みの容量を超えると、Azure はトークンの無制限消費を許可するのではなく、スロットリングを適用します。Microsoft Foundry は、AI Gateway を介した明示的なトークン管理制御もサポートしており、モデルデプロイメントに対して TPM 制限を設定できます。制限を超えるリクエストには、HTTP 429 Too Many Requests レスポンスが返されます。
コンテンツフィルタリングは、消費量ではなく、安全性のカテゴリを制御します。Azure AI Search のベクトルインデックス制限は、生成されたトークンではなく、検索ストレージを制御します。温度はサンプリングのランダム性を変化させますが、レートやコストの制御において強制的な役割を果たしません。
学習ガイドの参照項目 :Azure OpenAI のクォータ管理、TPM/RPM のレート制限、デプロイメント容量、スロットリング、Foundry AI Gateway トークン管理。

質問: **99**
このアプリケーションは、pgvectorを使用してAzure Database for PostgreSQLに保存されている500万個の埋め込みデータに対して類似性検索を実行します。クエリは、コサイン距離によるランキングを行う前に、部門ごとにフィルタリングすることがよくあります。
ベクトル類似性クエリのP95レイテンシがSLA目標値を超えています。監視の結果、クエリ実行中にCPU使用率が継続的に高い状態であることが確認されました。
フィルタリングされたベクトル類似性クエリのP95レイテンシを削減する必要があります。
あなたはどうすべきでしょうか？
A. 埋め込み列にpgvectorインデックスを作成します。
B. 類似性を示す ORDER BY 句を削除します。
C. ステートメントのタイムアウト時間を延長します。
D. 埋め込みデータをJSON形式で保存します。
正解: (正解を表示します)

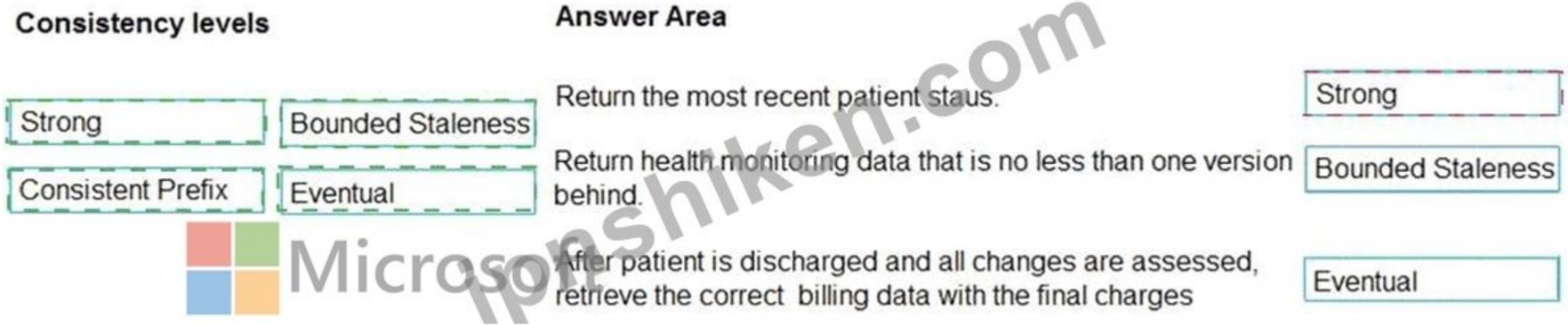
質問: **100**
パートナー向けサービス用の接続文字列は、技術要件に従って設定する必要があります。
何を使うべきでしょうか？
A. App Service 設定における Azure Key Vault の参照
B. GitHubの秘密
C. Azure Container Registry Helmチャートパッケージ
D. Dockerfileの環境変数設定
正解: (正解を表示します)
詳細説明 :App Service 設定における Azure Key Vault の参照は、コンテナ イメージ、ソース管理、およびアプリケーション構成に直接保存されるシークレットを除外するという要件を満たします。App Service は、実行時にアプリ ID を使用して参照されたシークレットを解決するため、アプリケーションはイメージに資格情報を埋め込むことなく、通常の設定として値を使用できます。GitHub シークレットは、実行時の App Service シークレット配信メカニズムではなく、ビルド/デプロイ シークレットです。Dockerfile の ENV 命令は、シークレット マテリアルをイメージ構成に配置し、要件に違反します。
学習ガイドの関連性 :セキュリティと運用 :Key Vault、アプリ構成、マネージドID、OpenTelemetry、Azure Monitor、およびKQLベースのトラブルシューティング。
Microsoft Learn 公式リファレンス: AI-200 学習ガイド | App Service および Functions での Key Vault の使用に関するリファレンス | Azure リソースのマネージド ID

質問: **101**
あなたは、病院向けに以下のユースケースをサポートするソリューションを開発しています。

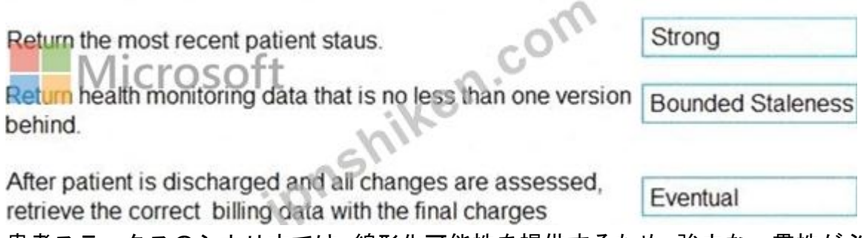
複数のユーザーが異なる場所で患者記録を更新した場合でも、最新の患者ステータスの詳細を取得する必要があります。
取得した患者の健康状態モニタリングデータは、最新バージョンまたは以前のバージョンである必要があります。
患者が退院し、すべての料金が査定された後、患者の請求記録には最終的な料金が記載されます。
Cosmos DB NoSQLデータベースをプロビジョニングし、データベースアカウントのデフォルトの整合性レベルを 強力」に設定します。インデックスモードの値を 一貫性」に設定します。
レイテンシを最小限に抑え、ソリューションの可用性への影響を最小限にする必要があります。シナリオに必要な整合性保証を満たすには、クエリレベルでデフォルトの整合性レベルをオーバーライドする必要があります。
どの整合性レベルを適用すべきでしょうか？回答するには、適切な整合性レベルを該当する要件にドラッグしてください。各整合性レベルは、1回、複数回、または全く使用しない場合があります。コンテンツを表示するには、ペイン間の分割バーをドラッグするか、スクロールする必要がある場合があります。
注：正解ごとに1ポイントが加算されます。



正解:



Explanation:



患者ステータスのシナリオでは、線形化可能性を提供するため、強力な一貫性が必要です。つまり、読み取り時に最新のコミット済み値が返されます。これは、更新が異なる場所から発生する可能性があり、アプリケーションが古い患者ステータスを返さなければならない場合に適切な保証となります。Microsoft は、Azure Cosmos DB の一貫性レベルとして 強力」を定義しており、弱いモデルと比較してレイテンシが大きくなり可用性が低下するという代償を伴うものの、最も強力な鮮度保証を提供すると述べています。

健康状態監視データの場合、境界付き陳腐化（Bounded Staleness）が意図された整合性レベルです。これは、最大バージョン数（K）または時間間隔（T）を使用して、読み取りが書き込みからどれだけ遅れるかを明示的に制限します。概念的には、これはデータが最新であるか、または限られたバージョン数だけ遅れている必要があるという要件に合致しています。
確定済みの請求データについては、結果整合性が最も低いレイテンシと最高の可用性を提供します。退院処理が完了し、それ以降の請求書き込みが行われなくなると、レプリカは最終的に同じ最終値に収束します。Microsoftは特に、それ以降の書き込みがない場合、より低い整合性レベルでも最終的には同じ収束データが得られることを指摘しています。
Azure Cosmos DB では、アカウントの「強いデフォルト」よりも一貫性を緩和するリクエストも許可されており、これはまさにここで説明されている最適化パターンです。
学習ガイドの参照: Azure Cosmos DB for NoSQL # 強力な一貫性、境界付き陳腐化、結果的一貫性、リクエストごとの一貫性オーバーライド、レイテンシ/可用性のトレードオフ。

質問: **102**
ホットスポットに関する質問
App ServiceにLinuxコンテナイメージをデプロイします。
コンテナは実行時に以下の環境変数を必要とします。
- MODEL_VERSIONという名前の非機密構成値
- 安全に保管しなければならないデータベースパスワード
これらの環境変数をランタイム時に提供するように、App Serviceを設定する必要があります。
どの設定を使用すべきですか？回答するには、回答欄で適切なオプションを選択してください。
注：正解ごとに1ポイントが加算されます。

Answer Area

Container configuration settings

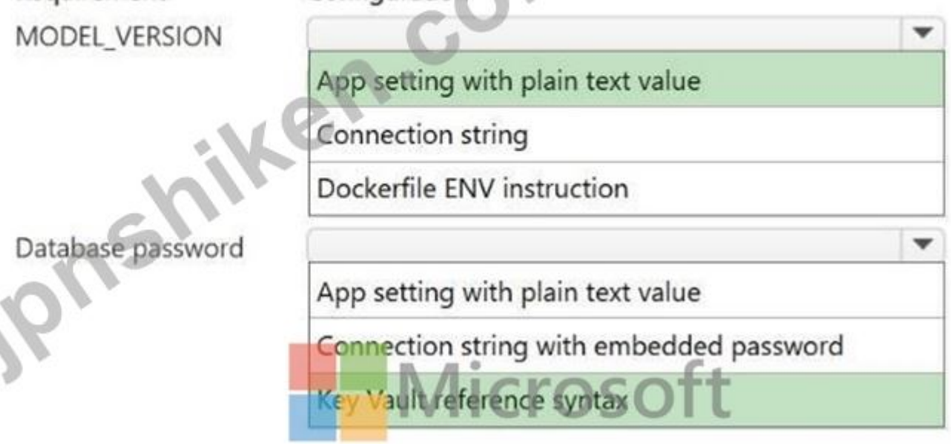
Requirement	Configuration
MODEL_VERSION	App setting with plain text value Connection string Dockerfile ENV instruction
Database password	App setting with plain text value Connection string with embedded password Key Vault reference syntax

Microsoft
jpnshiken.com

正解:

Answer Area

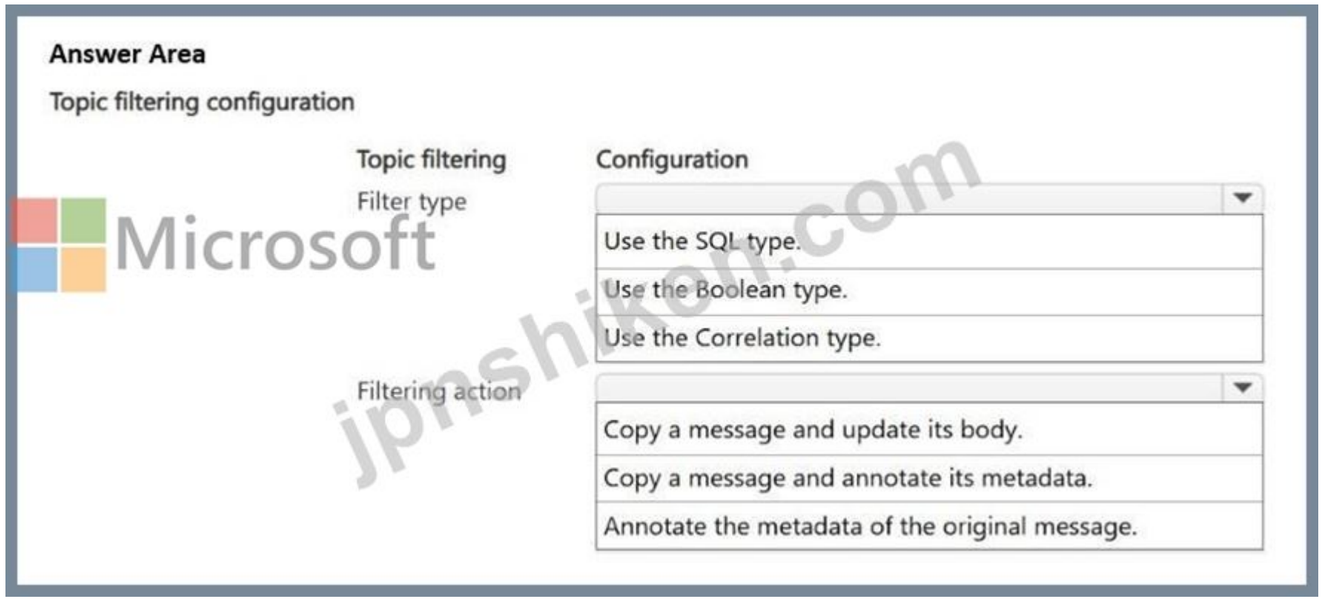
Container configuration settings



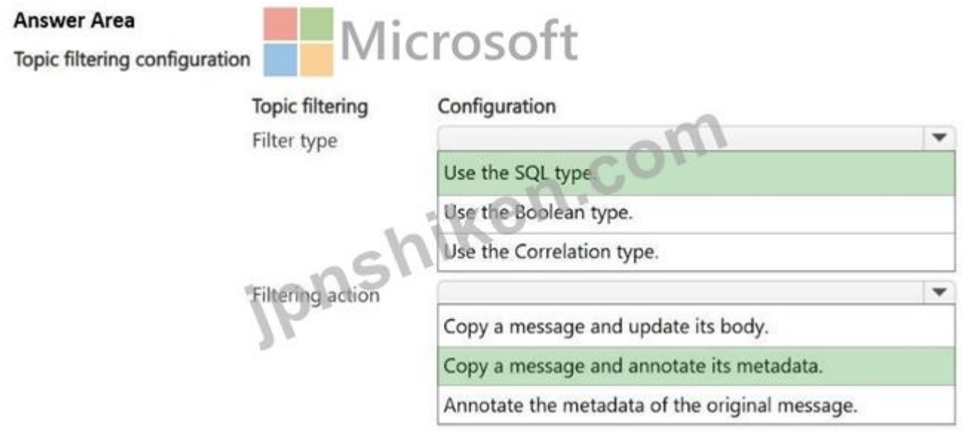
jpnshiken.com
Microsoft

Explanation:
ボックス1 :プレーンテキスト値を含むアプリ設定
Azure App Service にデプロイされた Linux カスタム コンテナーに、MODEL_VERSION という名前の機密性の低い環境変数を提供するには、それをアプリケーション設定 (App Setting) として構成する必要がありま
す。Azure App Service では、アプリケーション設定で定義されたカスタム構成は、実行時に環境変数として Linux コンテナーに自動的に挿入されます。
ボックス2 :Key Vaultの参照構文
App Serviceアプリケーション設定内で、データベースパスワードをAzure Key Vault参照として構成する必要があります。これにより、秘密情報がアプリ構成に平文で保存されるのを防ぎ、実行時に環境変数として安全にコンテナに挿入されます。
参照：
https://learn.microsoft.com/en-us/azure/app-service/tutorial-custom-container?tabs=azure-cli&pivots=container-linux

質問**: 103**
ホットスポットに関する質問
Azure Service Bus の名前空間に、Topic1 という名前のトピックが含まれています。
あなたは、トピック1に対してSub1という名前のサブスクリプションを作成する予定です。Sub1では、トピック1からのメッセージをシステムプロパティに基づいてフィルタリングし、フィルタリングされた各メッセージに注釈を付けるアクションを適用する予定です。
フィルタリングを設定する必要があります。
フィルタリングの設定はどのようにすればよいですか？回答するには、回答欄で適切なオプションを選択してください。
注：正解ごとに1ポイントが加算されます。



正解:



Explanation:

ボックス1 :SQL型を使用します。

この目的を達成するには、SQL フィルタ (SqlFilter とも呼ばれます) タイプを使用する必要があります。ボックス 2: メッセージをコピーしてメタデータに注釈を付けます。

適切なフィルタリング方法は、メッセージをコピーしてそのメタデータに注釈を付けることです。

アクションが適用されると（新しいプロパティの設定や、SQL フィルター アクションによる既存のプロパティの変更など）、Azure Service Bus はサブスクリプションの受信メッセージのコピーを作成し、そのコピーのメタデータに注釈を付けます。

参照：

https://turbo360.com/blog/azure-service-bus-topic-filter

質問**: 104**

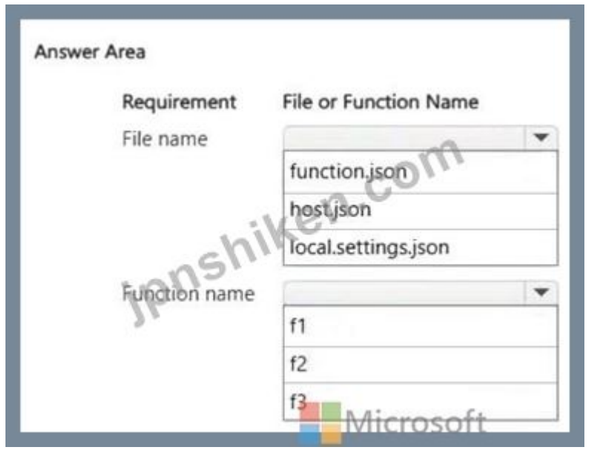
あなたは、音声および視覚アラームの開始と停止をトリガーする単一の信号サーバーを備えた危険通知システムを構築しています。

アラームを発行するために Azure Service Bus を実装すると、各アラーム コントローラーはトランザクションの一部として Azure Service Bus を使用してアラーム シグナルを受信します。アラーム イベントは監査目的で記録する必要があります。各トランザクション レコードには、アクティブ化されたアラーム タイプに関する情報を含める必要があります。
返信履歴監査ソリューションを導入する必要があります
どの2つの行動をとるべきでしょうか？それぞれの正解は、解決策の一部を示しています。
注：正解ごとに1ポイントが加算されます。
A. ハザードメッセージのSessionIdプロパティの値をReplyToSessionIdプロパティに割り当てます。
B. ハザードメッセージの MessageId プロパティの値を SequenceNumber プロパティに割り当てます。
C. ハザードメッセージのSequenceNumberプロパティの値をDeliveryCountプロパティに割り当てます。
D. ハザードメッセージのSessionIdプロパティの値をSequenceNumberプロパティに割り当てます。
E. ハザードメッセージ MessageId p ' プロパティの値を DeliveryCount プロパティに割り当てます。
F. ハザードメッセージの MessageId プロパティの値を CorrelationId プロパティに割り当てます
正解: **(**正解を表示します**)**
詳細説明：多重化されたリクエスト/レスポンスパターンでは、リクエストの ReplyToSessionId は、レスポンスにどの SessionId を配置するかをレスポンダーに指示します。そのため、リクエストの SessionId を ReplyToSessionId に渡すことで、レスポンスセッションのルーティングコンテキストが提供されます。相関関係の場合、標準のリクエスト/レスポンスパターンでは、元のリクエストの MessageId がレスポンスの CorrelationId にコピーされます。SequenceNumber と DeliveryCount はブローカーが管理する配信メタデータであり、書き込み可能なアプリケーション相関フィールドではないため、提案されている監査リンクメカニズムとして使用することはできません。
学習ガイドの関連性 :Azureサービス統合 :Service Bus、Event Grid、Azure Functionsトリガー
バインディング、およびイベント駆動型処理。
Microsoft Learn 公式資料: AI-200 学習ガイド | サービスバスのメッセージ、ルーティング、および相関関係 | サービスバスのメッセージセッション

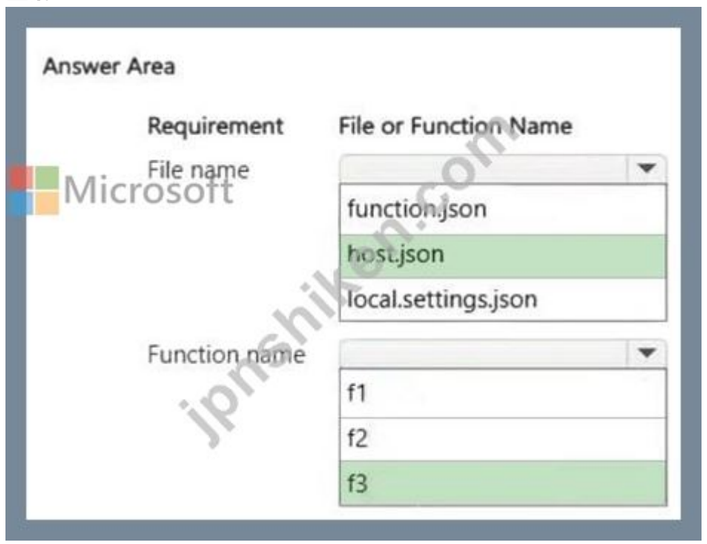
質問: **105**
ホットスポットに関する質問
企業向けの従量課金ホスティングプランを使用している Azure Functions アプリがあります。このアプリには、次の関数が含まれています。

Function Name	Trigger type
f1	HTTP
f2	Timer
f3	Azure Queues

Microsoft jpnshiken.com
アプリで動的な同時実行機能を有効にする予定です。会社側は、各関数の同時実行レベルを個別に管理することを求めています。
アプリを動的同時実行に対応させるように設定する必要があります。
どのファイル名または関数名を使用すべきですか？回答するには、回答欄で適切な値を選択してください。
注：正解ごとに1ポイントが加算されます。



正解:



Explanation:

ボックス1 :host.json

アプリケーションの動的な同時実行性を設定するには、host.json 設定ファイルを使用します。

host.json: 適切な拡張機能/拡張機能ロック構成の下で、アプリレベルの動的同時実行設定 "(dynamicConcurrencyEnabled": true) を有効にするグローバルメタデータファイル。

ボックス2 :f3

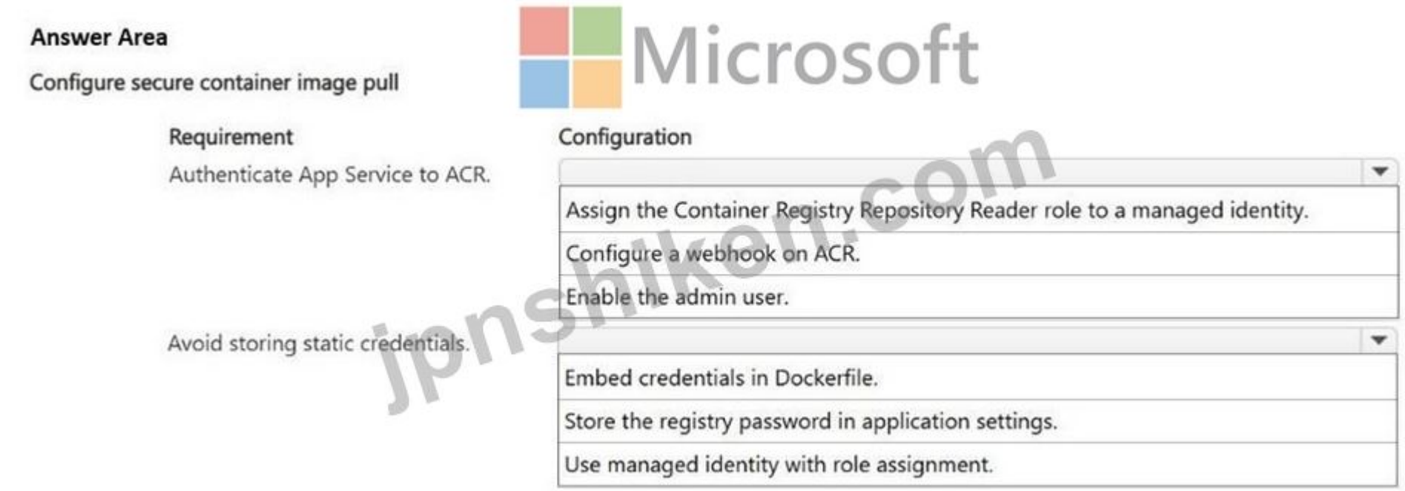
host.json ファイルを Azure Queue トリガー関数 (f3) に適用してください。これは、HTTP トリガーやタイマー トリガーではなく、ランタイム駆動型のストレージ トリガーおよびキュー トリガーの場合、動的同時実行が拡張機能ごとにネイティブにサポートおよび管理されるためです。

Azure Queue トリガー関数: Azure Functions の動的同時実行機能は、Azure Queue、Blob、Service Bus ストレージ トリガーなどのランタイム駆動のバックグラウンド バインディングを特に対象としており、関数ごとに処理制限を個別に管理し、自動的に調整することができます。

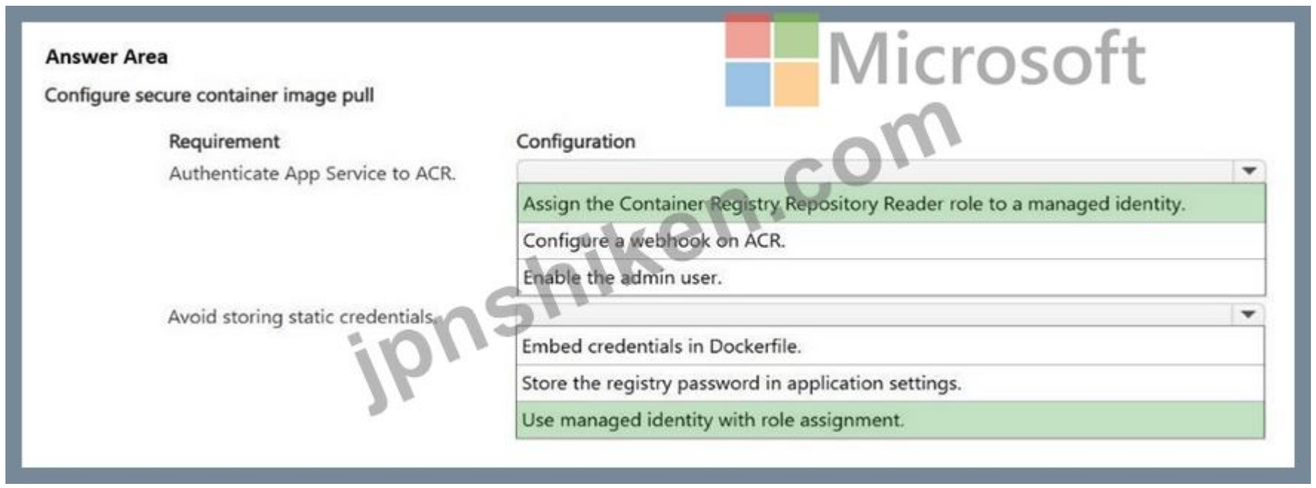
参照 :

https://learn.microsoft.com/en-us/azure/azure-functions/functions-concurrency

質問: **106**
ホットスポットに関する質問
Azure Container Registry (ACR) から App Service にプライベート コンテナ イメージをデプロイします。
App Service はイメージを取得するために ACR に対して認証を行う必要があります。
ソリューションは、静的なレジストリ認証情報を保存してはなりません。
セキュアイメージプル認証を設定する必要があります。
どの設定を使用すべきですか？回答するには、回答欄で適切なオプションを選択してください。
注：正解ごとに1ポイントが加算されます。



正解:

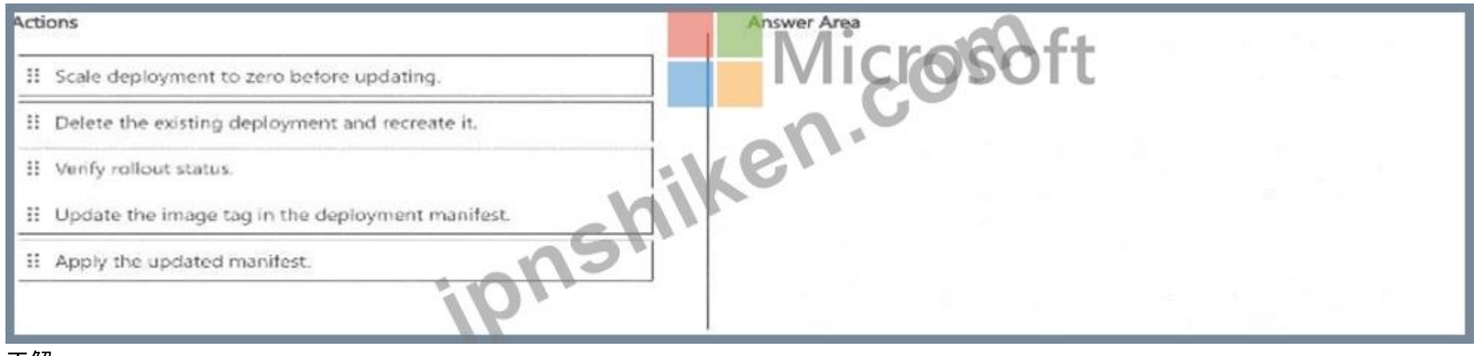


Explanation:
ボックス1：管理対象IDにコンテナレジストリリーダーの役割を割り当てます。

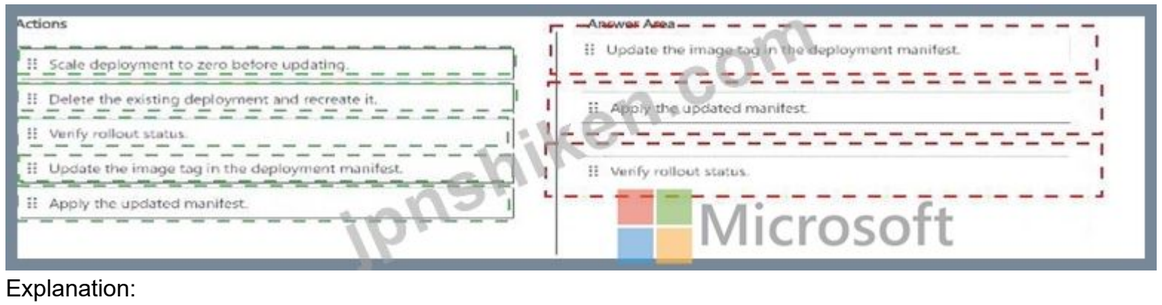
パスワードを管理せずに、Azure Container Registry (ACR) から Azure App Service にプライベート コンテナー イメージを安全にデプロイするには、App Service でマネージド ID を構成し、ACR で AcrPull ロールを付与する必要があります。
ボックス2 :ロール割り当てでマネージドIDを使用する。
要件を満たすには、マネージドIDとAzureロールベースアクセス制御（RBAC）のロール割り当てを組み合わせて使用する必要があります。
参照：
https://medium.com/@miketobicarter/deploying-a-net-app-to-azure-using-docker-acr-app-service-and-github-actions-step-by-step-3ed6ae8a4d0b

有効的な**AI-200**問題集はJPNTest.com提供され、**AI-200**試験に合格することに役に立ちます！JPNTest.comは今最新**AI-200**試験問題集を提供します。JPNTest.com AI-200試験問題集はもう更新されました。ここで**AI-200**問題集のテストエンジンを手に入れます。最新版のアクセス、https://www.jpntest.com/shiken/AI-200-mondaishu　**144**問、３０%ディスカウント、特別な割引コード: **JPNshiken**」

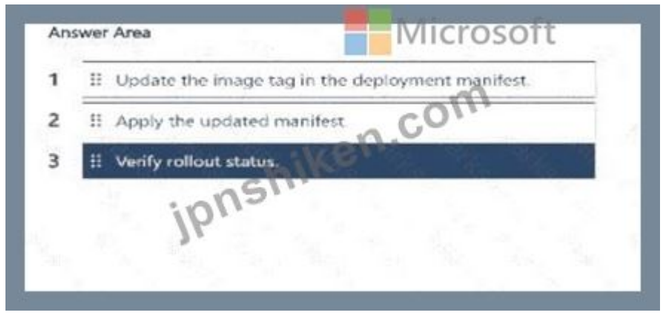
質問: **107**
既存のAKSクラスターと、Azure Container Registryに保存されているコンテナイメージがあります。
トラフィックを中断することなくローリングアップデートをサポートするデプロイメントマニフェストを使用して、新しいバージョンをデプロイする必要があります。
マニフェストファイルを使用して、ローリングアップデートを実行する必要があります。
どの3つの行動を順番に実行すべきでしょうか？回答するには、行動リストから適切な行動を回答欄に移動させ、正しい順序に並べ替えてください。



正解:



Explanation:



質問: **108**

Azure Cosmos DB をデータストアとして使用する Azure Web アプリがあるとします。次の PowerShell スクリプトを実行して、Cosmos DB コンテナーを作成します。

```
$resourceGroupName = "testResourceGroup"
$accountName = "testCosmosAccount"
$databaseName = "testDatabase"
$containerName = "testContainer"
$partitionKeyPath = "/EmployeeId"
$autoscaleMaxThroughput = 5000

New-AzCosmosDBSqlContainer `
  -ResourceGroupName $resourceGroupName `
  -AccountName $accountName `
  -DatabaseName $databaseName `
  -Name $containerName `
  -PartitionKeyKind Hash `
  -PartitionKeyPath $partitionKeyPath `
  -AutoscaleMaxThroughput $autoscaleMaxThroughput
```

You create the following queries that target the container:

```
SELECT * FROM c WHERE c.EmployeeId > '12345'
```

以下の各記述について、正しい場合は「はい」を選択してください。そうでない場合は「いいえ」を選択してください。

注：正解ごとに1ポイントが加算されます。

Answer Area

Statements	Yes	No
The minimum throughput for the container is 400 RU/s.	○	○
The first query statement is an in-partition query.	○	○
The second query statement is a cross-partition query.	○	○

正解:



Explanation:



この質問は、AI-200の目標「Azure Cosmos DB for NoSQLを使用してAIソリューションを開発する」に直接対応しており、クエリの実行、クエリのパフォーマンスの最適化、およびリクエストユニット（RU）消費量の最適化が含まれます。

ステートメント 1: いいえ - 最小スループットは 400 RU/s ではありません。

PowerShell コマンドは、以下の内容でコンテナをプロビジョニングします。

-AutoscaleMaxThroughput 5000

Azure Cosmos DB の自動スケーリングは、構成された最大スループットの約 10% ～ 100% の範囲で動作します。Microsoft のドキュメントには、5,000 RU/s でプロビジョニングされた自動スケーリング コンテナーが 500 RU/s から 5,000 RU/s の間でスケーリングされる例が具体的に示されています。したがって、このコンテナーの場合、最小動作自動スケーリング スループットは 400 RU/s ではなく 500 RU/s です。

したがって：

「コンテナの最小スループットは400 RU/秒です。」# いいえ

ステートメント 2: いいえ - 最初のクエリはパーティション内クエリではありません。

コンテナは以下を使用します。

/従業員ID

パーティションキーとして。

最初のクエリは次のとおりです。

SELECT * FROM c WHERE c.EmployeeId > ' 12345 '

クエリはパーティションキーを参照していますが、等価述語ではなく範囲述語（>）を使用しています。

Microsoft は、パーティション キーに対する範囲フィルタは単一の物理パーティションにスコープされないことを明示的に述べています。

パーティション内クエリとして認められるためには、フィルタは適用可能なパーティションを識別する必要があり、通常は次のような等価述語を使用します。

WHERE c.EmployeeId = '12345'

Microsoft のドキュメントには、基本的に同じ例が示されています。DeviceId でパーティション分割されたコンテナに対して DeviceId > ... を使用するクエリは、パーティション内クエリではありません。

したがって：

「最初のクエリ文はパーティション内クエリです。」# いいえ

ステートメント 3: はい - 2 番目のクエリはクロスパーティション クエリです。

2つ目のクエリは次のとおりです。

SELECT * FROM c WHERE c.UserId = ' 12345 '

コンテナーのパーティションキーは /EmployeeId であり、/UserId ではありません。クエリにパーティションキーに関するフィルタが含まれていないため、Azure Cosmos DB はクエリを単一の論理パーティションにルーティングできません。クエリは該当する物理パーティション全体に分散され、結果が結合されます。
マイクロソフトはこの動作について次のように直接説明しています。クエリにパーティションキーに対するフィルタが含まれていない場合、クエリはパーティションをまたいで実行される必要があります。
したがって：
「2番目のクエリ文はクロスパーティションクエリです。」# はい

質問: **109**
アップロードされた画像に成人向けコンテンツや暴力的なコンテンツが含まれているかどうかを検出してから、公開フォーラムへの投稿を許可する必要があります。どのようなツールを使用すべきでしょうか？
A. Azure AI Custom Vision 物体検出
B. Azure AIコンテンツセーフティ画像モデレーション
C. Azure AI Face API
D. Azure AI ドキュメントインテリジェンス
正解: **(正解を表示します)**
Azure AI Content Safetyの画像モデレーション機能は、性的、暴力的、自傷行為、ヘイトコンテンツなどのカテゴリを画像から検出するために特別に設計されており、対応可能な深刻度スコアを返します。

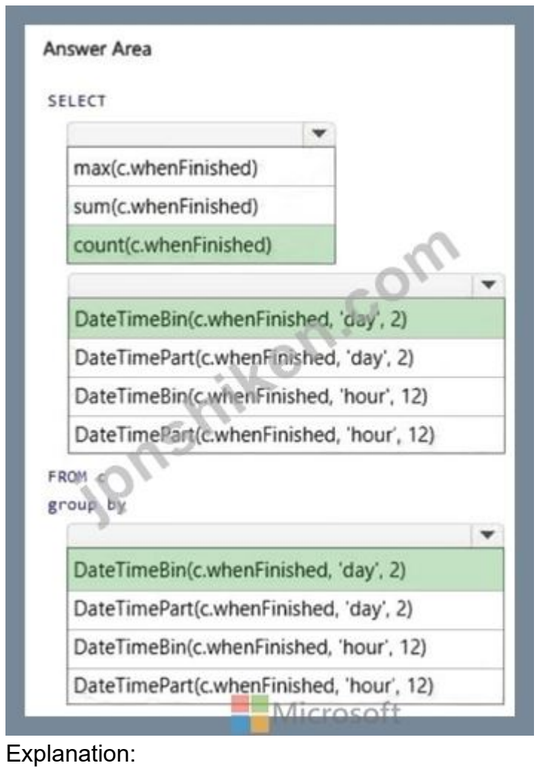
質問: **110**
ホットスポットに関する質問
あなたは、ユーザーの入力に基づいてAIが生成した画像を販売するアプリケーションを開発しています。最近、2日ごとに異なる広告を表示するマーケティングキャンペーンを開始しました。
販売データはAzure Cosmos DBに保存され、各販売の日付は「whenFinished」という名前のプロパティに保存されます。
マーケティング部門は、個々の広告ごとの販売数を表示するビューを必要としている。
ビューに対してクエリを実装する必要があります。
質問にはどのように回答すればよいですか？回答するには、回答欄で適切なオプションを選択してください。
注：正解ごとに1ポイントが加算されます。

Answer Area

```
SELECT
    [ max(c.whenFinished) | sum(c.whenFinished) | count(c.whenFinished) ]
    [ DateTimeBin(c.whenFinished, 'day', 2) | DateTimePart(c.whenFinished, 'day', 2) | DateTimeBin(c.whenFinished, 'hour', 12) | DateTimePart(c.whenFinished, 'hour', 12) ]
FROM c
group by
    [ DateTimeBin(c.whenFinished, 'day', 2) | DateTimePart(c.whenFinished, 'day', 2) | DateTimeBin(c.whenFinished, 'hour', 12) | DateTimePart(c.whenFinished, 'hour', 12) ]
```

jpnshiken.com

正解:

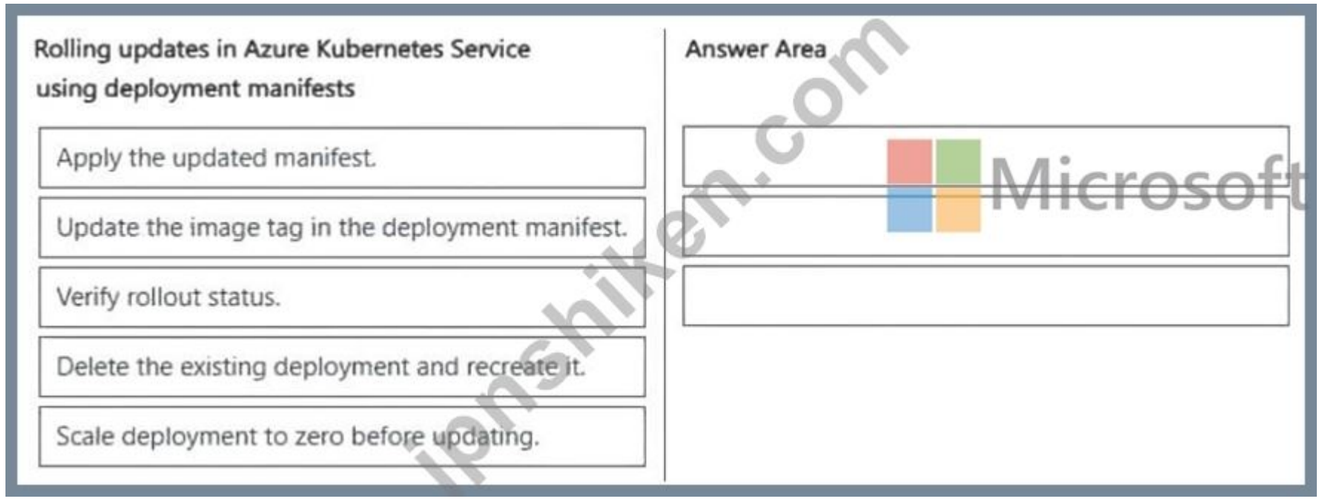


Explanation:
ボックス1 :count(c.whenFinished)
販売数を取得するには、カウント関数を使用する必要があります。
ボックス2 :DateTimeBin(c.whenFinished, 'day', 2)
ボックス3 :DateTimeBin(c.whenFinished, 'day', 2)
参照：
https://learn.microsoft.com/en-us/cosmos-db/query/datetimebin

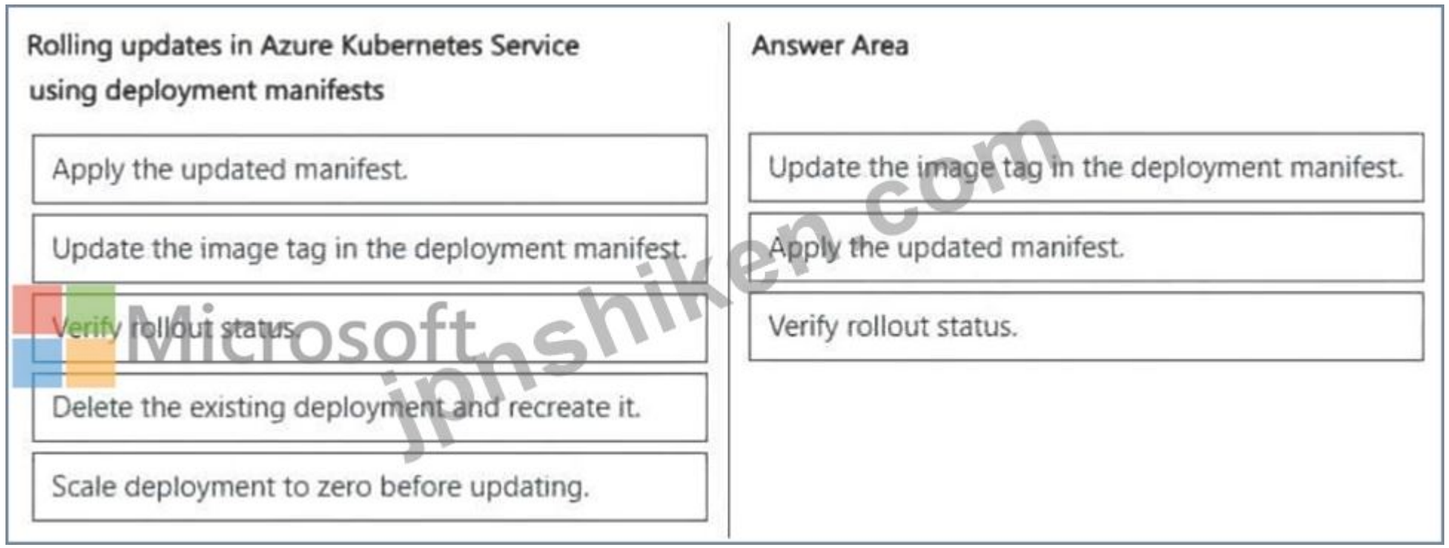
質問**: 111**
コンテナのソースコードはGitリポジトリに保存されています。
コンテナレジストリは、開発者がGitリポジトリにコードをコミットするたびに、新しいコンテナイメージを自動的に構築して保存する必要があります。
外部のビルドインフラストラクチャの使用は最小限に抑える必要があります。
ビルドプロセスをネイティブかつ自動的に管理するには、Azure Container Registry (ACR) を構成する必要があります。
どの2つのACRコンポーネントを使用すべきでしょうか？それぞれの正解は、解決策の一部を示しています。
注：正解ごとに1ポイントが加算されます。
A. Webhook
B. クイックタスク
C. アーティファクトキャッシュルール
D. ソーストリガー型タスク
正解**: (**正解を表示します**)**

ソリューションに含めるべき2つのコンポーネントは、ソーストリガー型タスクとWebhookです。
ソーストリガー型タスク :このコンポーネントは、Azure 内でネイティブに要件を満たします。Azure Container Registry (ACR) タスクは、Git リポジトリ (GitHub や Azure Repos など) を監視できます。
開発者がコードをコミットすると、外部のCI/CDビルドインフラストラクチャを必要とせずに、Azure内でネイティブビルドが自動的にトリガーされます。
Webhook :これは、GitリポジトリとACRを接続する基盤となるメカニズムです。コミットが発生すると、GitリポジトリはWebhook通知をAzure Container Registryに送信し、ソーストリガー型タスクに新しいイメージのビルドを開始するよう指示します。
参照 :
https://learn.microsoft.com/en-us/azure/container-registry/container-registry-tutorial-build-task

質問: **112**
ドラッグアンドドロップ問題
既存のAKSクラスターと、Azure Container Registryに保存されているコンテナイメージがあります。
トラフィックを中断することなく、新しいバージョンをデプロイする必要があります。
マニフェストファイルを使用して、ローリングアップデートを実行する必要があります。
どの3つの行動を順番に実行すべきでしょうか？回答するには、行動リストから適切な行動を回答欄に移動させ、正しい順序に並べ替えてください。



正解:



Explanation:
ステップ1 :デプロイメントマニフェスト内のイメージタグを更新する
これにより、定義ファイルが変更され、Azure Container Registryに保存されている新しいバージョンのイメージを指すようになります。
ステップ2：更新されたマニフェストを適用する
`kubectl apply -f <manifest.yaml>`を実行すると、Kubernetesはネイティブにダウンタイムゼロのローリングアップデート戦略を開始し、古いPodを新しいPodに段階的に置き換えます。
ステップ3：展開状況を確認する
`kubectl rollout status deployment/<deployment-name>` を実行すると、新しいバージョンが失敗なく正常にデプロイされることを追跡および確認できます。
参照：
https://learn.microsoft.com/en-us/azure/aks/tutorial-kubernetes-deploy-application

質問: **113**
ホットスポットに関する質問
Azure Monitor のリクエスト テーブルを使用して、レイテンシの問題をトラブルシューティングしています。
クラウドロール名ごとにグループ化された平均リクエスト時間を計算し、結果を過去30分に限定する必要があります。
どのコード要素を選択すべきですか？回答するには、回答欄で適切なオプションを選択してください。
注：正解ごとに1ポイントが加算されます。

KQL operator selection for filtering and aggregation

```
requests
| [ distinct | render | where ] timestamp > ago(30m)
| [ extend | project | summarize ] avg(duration) by cloud_RoleName
```

jpnshiken.com Microsoft

正解:

KQL operator selection for filtering and aggregation

```
requests
| [ distinct | render | where ] timestamp > ago(30m)
| [ extend | project | summarize ] avg(duration) by cloud_RoleName
```

jpnshiken.com Microsoft

Explanation:

ボックス1：

Kustoクエリ言語（KQL）では、where演算子はテーブルの行をフィルタリングし、特定の述語または条件を満たすレコードのみを含めます。

ボックス2：要約

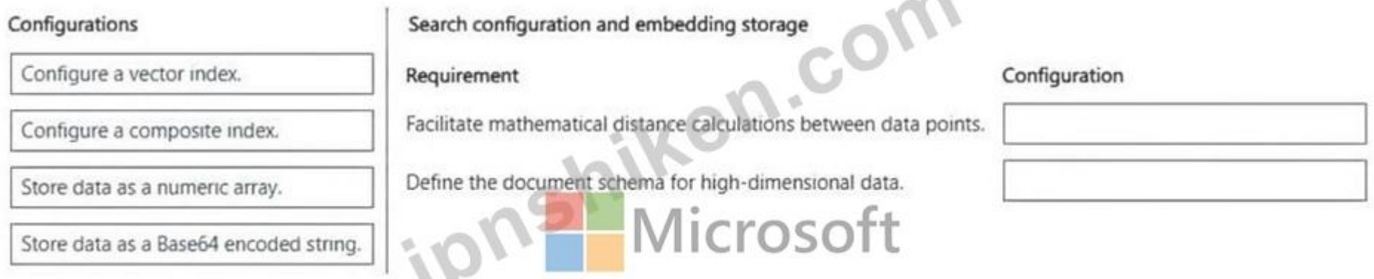
summarize は集計（件数 平均値、パーセンタイル）に使用されます。

参照：

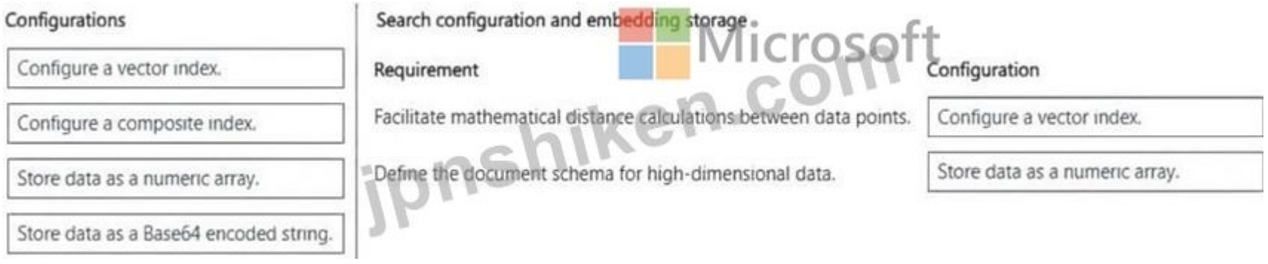
https://world.optimizely.com/blogs/K-Khan-/Dates/2025/12/troubleshooting-with-azure-application-insights-using-kql

質問: **114**
HTTP トリガー型の Azure Function は、Azure Blob Storage に保存されている JSON 構成ファイルを読み込む必要があります。
設定ファイルの内容を取得する必要があります。
何を使うべきでしょうか？
A. 出力バインディング
B. 入力バインディング
C. タイマートリガー
D. HTTPトリガー
正解: **(正解を表示します)**
HTTP トリガー関数から Azure Blob Storage に保存されている構成ファイルを読み取るには、Azure Blob Storage 入力バインディングを使用する必要があります。
入力バインディングを使用すると、SDKクライアントを使用して明示的に接続、認証、およびデータのダウンロードを行うコードを記述することなく、外部データを関数にネイティブに接続できます。
戦略概要
トリガー :HTTPトリガーを使用して、Webリクエストを介して関数を実行します。
入力 :ストレージアカウント内の特定のJSONパスを指すBlob入力バインディングを使用します。
結果 :Azure Functions ランタイムはファイルの内容を自動的に取得し、文字列、バイト配列、または解析済みオブジェクトパラメーターとして関数コードに直接提供します。
参照 :
https://learn.microsoft.com/en-us/azure/azure-functions/functions-triggers-bindings

質問: **115**
ドラッグアンドドロップ問題
Azure Cosmos DB for NoSQL API のネイティブなベクトル検索機能を使用して、検索拡張生成 (RAG) システムを実装しています。
技術記事を格納する「Documents」という名前のコンテナがあります。各記事には「embedding」という名前のプロパティが含まれています。
システムが、ユーザーの検索クエリと保存されている記事との間で効率的な類似性検索を実行できることを確認する必要があります。
セマンティック検索をサポートするには、データベースリソースを設定する必要があります。
どの構成を使用すべきでしょうか？回答するには、適切な構成を正しい要件に移動してください。各構成は、1回、複数回、またはまったく使用しない場合があります。コンテンツを表示するには、ペイン間の分割バーを移動したり、スクロールしたりする必要がある場合があります。
注：正解ごとに1ポイントが加算されます。



ionshiken.com
Microsoft
正解:



Explanation:
ボックス1 :ベクトルインデックスを設定します。
Azure Cosmos DB for NoSQL のネイティブなベクトル検索機能を使用して意味的検索をサポートし、数学的な距離計算を実行するには、ベクトルインデックスを構成する必要があります。
Azure Cosmos DB for NoSQL は、ベクトルデータを整理するために、DiskANN や QuantizedFlat などの特殊なインデックス構造を使用します。コンテナ構成内でベクトルインデックスポリシーを定義することは、ドキュメントの完全なインデックスなしスキャンを必要とせずに、VectorDistance などの関数を使用して高速な類似性検索を可能にするための必須要件です。
ボックス2 :データを数値配列として保存する
Azure Cosmos DB for NoSQL では、埋め込みプロパティを JSON ドキュメント内に数値配列 (float32 のような浮動小数点数の配列) として保存する必要があります。
参照 :
https://learn.microsoft.com/en-us/azure/cosmos-db/gen-ai/agentic-retrieval

質問**: 116**
Sub1 という名前の Azure サブスクリプションがあり、その中に RG1 という名前のリソース グループと SB1 という名前のサービス バス キューが含まれています。
Azure Event Grid プッシュイベント サブスクリプションを実装し、RG1 でリソースが作成、変更、または削除されるたびに SB1 にイベントを配信するように計画しています。開発と構成の作業を最小限に抑える必要があります。
計画している実装のために、イベントグリッドのトピックを作成する必要があります。
どのようなイベントテーマを作成すべきでしょうか？
A. 名前の契約
B. システム
C. カスタム
D. イベントドメイン
正解**: (**正解を表示します**)**

有効的な**AI-200**問題集はJPNTest.com提供され、**AI-200**試験に合格することに役に立ちます！JPNTest.comは今最新**AI-200**試験問題集を提供します。JPNTest.com AI-200試験問題集はもう更新されました。ここで**AI-200**問題集のテストエンジンを手に入れます。最新版のアクセス、https://www.jpntest.com/shiken/AI-200-mondaishu　**144**問、３０%ディスカウント、特別な割引コード**: JPNshiken**」