- **Vendor: Microsoft**
- **Exam Code: AZ-203**
- **Exam Name: Developing Solutions for Microsoft Azure**
- **New Updated Questions from Braindump2go (Updated in May/2020)**

QUESTION 108
Hotspot Questions
You have an app that stores player scores for an online game. The app stores data in Azure tables using a class named PlayerScore as the table entity. The table is populated with 100,000 records.
You are reviewing the following section of code that is intended to retrieve 20 records where the player score exceeds 15,000. (Line numbers are included for reference only.)

```
1 public void GetScore(string playerId, int score, string gameName)
2 {
3    Table Query<DynamicTableEntity> query = new TableQuery<DynamicTableEntity>().Select(new string[] { "Score" })
       .Where(TableQuery.GenerateFilterConditionForInt("Score", QueryComparisons.GretaerThanOrEqual, 15000)).Take
(20);
4    EntityResolver<KeyValuePair<string, int?>> resolver =
      (partitionKey, rowKey, ts, props, etag) => new KeyValuePair<string, int?>(rowKey, props["Score"].Int32Value);
5    foreach (var scoreItem in scoreTable.ExecuteQuery (query, resolver, null, null))
6    {
      Console.WriteLine($"{scoreItem.Key} {scoreItem.Value}");
7    }
8   }

9  public class PlayerScore : TableEntity
10 {
11   public PlayerScore(string gameId, string playerId, int score, long timePlayed)
12   {
13     PartitionKey = gameId;
14     RowKey = playerId;
15     Score = score;
16     TimePlayed = timePlayed;
17   }
18   public int Score { get; set; }
19   public long TimePlayed { get; set; }
20   }
```

You have the following code. (Line numbers are included for reference only.)

```
01 public void SaveScore(string gameId, string playerId, int score, long timePlayed)
02 {
03   CloudStorageAccount storageAccount = CloudStorageAccount.Parse(connectionString);
04   CloudTableClient tableClient = storageAccount.CreateCloudTableClinet();
05   CloudTable table = tableClient.GetTableReference("scoreTable");
06   table.CreateIfNotExists();
07   var scoreRecord = new PlayerScore(gameId, playerId, score, timePlayed);
08   TableOperation insertOperation = TableOperation.Insert(scoreRecord);
09   table.Execute(insertOperation);
10  }
11  public class PlayerScore : TableEntity
12 {
13   public PlayerScore(string gameId, string playerId, int score, long timePlayed)
14   {
15     this.PartitionKey = gameId;
16     this.RowKey = playerId;
17   Score = score;
18   TimePlayed = timePlayed;
19   }
20   public int Score { get; set; }
21   public long TimePlayed { get; set; }
22 }
```

You store customer information in an Azure Cosmos database. The following data already exists in the database:

PartitionKey	RowKey	Email
Harp	Walter	wharp@contoso.com
Smith	Steve	ssmith@contoso.com
Smith	Jeff	jsmith@contoso.com

You develop the following code. (Line numbers are included for reference only.)

```
01  CloudTableClient tableClient = account.CreateCloudTableClient();
02  CloudTable table = tableClient.GetTableReference("people");
03  TableQuery<CustomerEntity> query = new TableQuery<CustomerEntity>()
04   .Where(TableQuery.CombineFilters(
05  TableQuery.Generate.And, TableQuery.GenerateFilterCondition(Email, QueryComparisons.Equal, "Smith")
06  TableOperators.And, TableQuery.GenerateFilterCondition(Email, QueryComparisons.Equal,
"ssmith@contoso.com")
07  ));
08  await table.ExecuteQuerySegmentedAsync<CustomerEntity>(query, null);
```

For each of the following statements, select Yes if the statement is true. Otherwise, select No.
NOTE: Each correct selection is worth one point.

Answer Area

	Yes	No
The code queries the Azure table and retrieves the TimePlayed property from the table	○	○
The code will display a maximum of twenty records.	○	○
All records will be sent to the client. The client will display records for scores greater than or equal to 15,000.	○	○
The scoreItem.Key property of the KeyValuePairs that ExecuteQuery returns will contain a value for PlayerID.	○	○

Answer:

Answer Area

	Yes	No
The code queries the Azure table and retrieves the TimePlayed property from the table	○	○
The code will display a maximum of twenty records.	○	○
All records will be sent to the client. The client will display records for scores greater than or equal to 15,000.	○	○
The scoreItem.Key property of the KeyValuePairs that ExecuteQuery returns will contain a value for PlayerID.	○	○

Explanation:
Box 1: No
Box 2: Yes
The TableQuery.Take method defines the upper bound for the number of entities the query returns.
Example:
query.Take(10);
Box 3: Yes
Box 4: Yes
References:
https://www.vkinfotek.com/azureqa/how-do-i-query-azure-table-storage-using-tablequery-class.html

QUESTION 109
Hotspot Question
You are working for a company that designs mobile applications. They maintain a server where player records are assigned to their different games. The tracking system is new and in development.
The application uses Entity Framework to connect to an Azure Database. The database holds a Player table and Game table.
When adding a player, the code should insert a new player record, and add a relationship between an existing game record and the new player record.
The application will call CreatePlayerWithGame with the correct gameIdand the playerId to start the process. (Line numbers are included for reference only.)

```
01. namespace ContosoCradt
02. {
03.    public class PlayerDbContext : DbContext
04.    {
05.  public PlayerDbContext() : base ("name-dBConnString") { }
06.  public DbSet<Player> Players { get ; set ; }
07.  public DbSet<Game> Games { get ; set ]
08.  protected override void DmModelCreating(DBModelBuilder modelBuilder)
09.  {
10.   modelBuilder.Entity<Player>().MesMany(x => x.Games). WithMany (x => x Players);
11.  }
12.  }
13.   internal series class dbConfiguration : DbMigrationConfiguration<PlayerDbContext>
14.  {
15.  public dbConfiguration() . (AutomaticMigrationsEnabled = true ; )
16.  {
17.    public class mp
18.    {
19.  public void CreatePlayerWithGame(int playerId, int gameId) => AddPlayer(playerId, GetGame[gameId]);
20.  public game GetGame(int gameId)
21.  {
22.    using (var db = new PlayerDbContext())
23.  {
24.    return db.Games.FirstOrDefault(x => x.GameId == gameId);
25.  }
26.   }
27.   public Player AddPlayer (int playerId, Game game)
28.   {
29.  using (var db = new PlayerDbContext())
30.  {
31.    var player = new Player
32.     {
33.      PlayerId = playerId,
34.      Games = new List <Game> {game },
35.  });
36.  db.Players.Add(player);
37.  db.SaveChanges();
38.  return player;
39.    }
40. }
41. public class Player
42. {
43.  public int PlayerId { get ; set; }
44.  public string PlayerName { get ; set; }
45.  public virtual List<Game> Games { get ; set; }
46.  }
47.  public class Game
48.  {
49.  public int GameIs { get ; set }
50.  public string Title { get ; set; }
51.  public string Platform { get ; set; ]
52.  public virtual List<Player> Players { get ; set; }
53.     }
54.  }
```

For each of the following statements, select Yes if the statement is true. Otherwise, select No.
NOTE: Each correct selection is worth one point.

Answer Area

	Yes	No
The code will successfully insert a player record.	○	○
The code has a bug and will insert an additional copy of the Game record with a new Id.	○	○
The code has a bug and will insert the wrong gameId value.	○	○
There is a valid many-to-many relationship between Players and Games.	○	○

Answer:

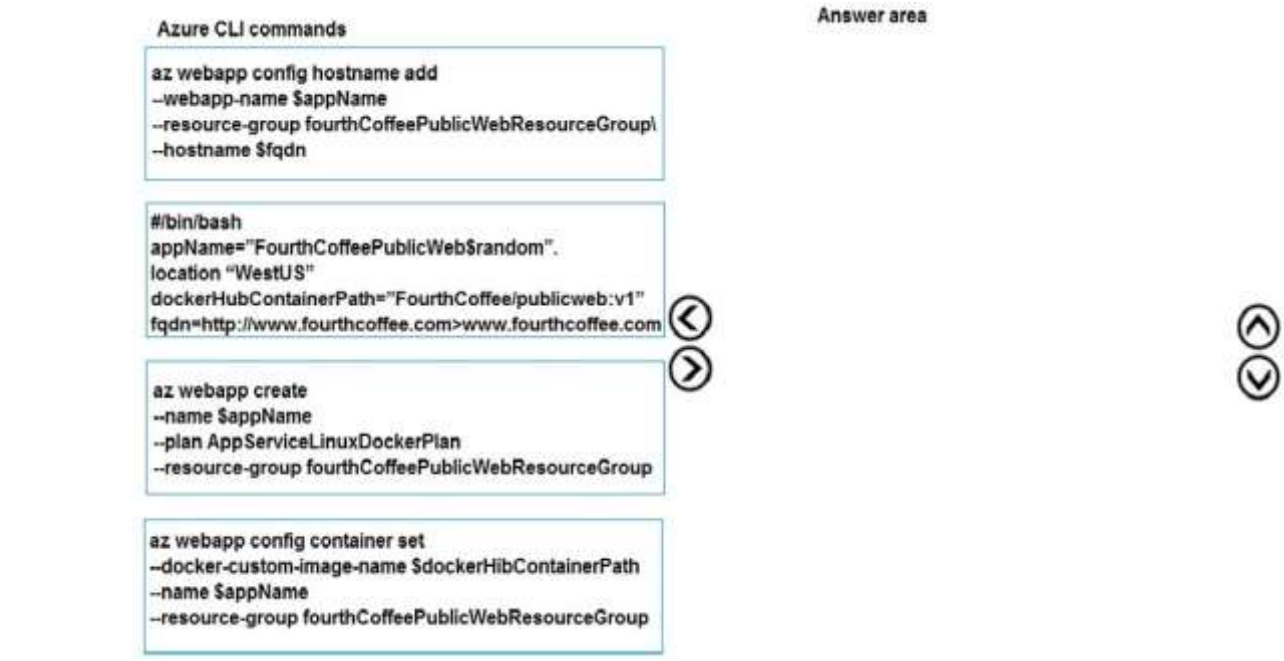
Answer Area

	Yes	No
The code will successfully insert a player record.	○	○
The code has a bug and will insert an additional copy of the Game record with a new Id.	○	○
The code has a bug and will insert the wrong gameId value.	○	○
There is a valid many-to-many relationship between Players and Games.	○	○

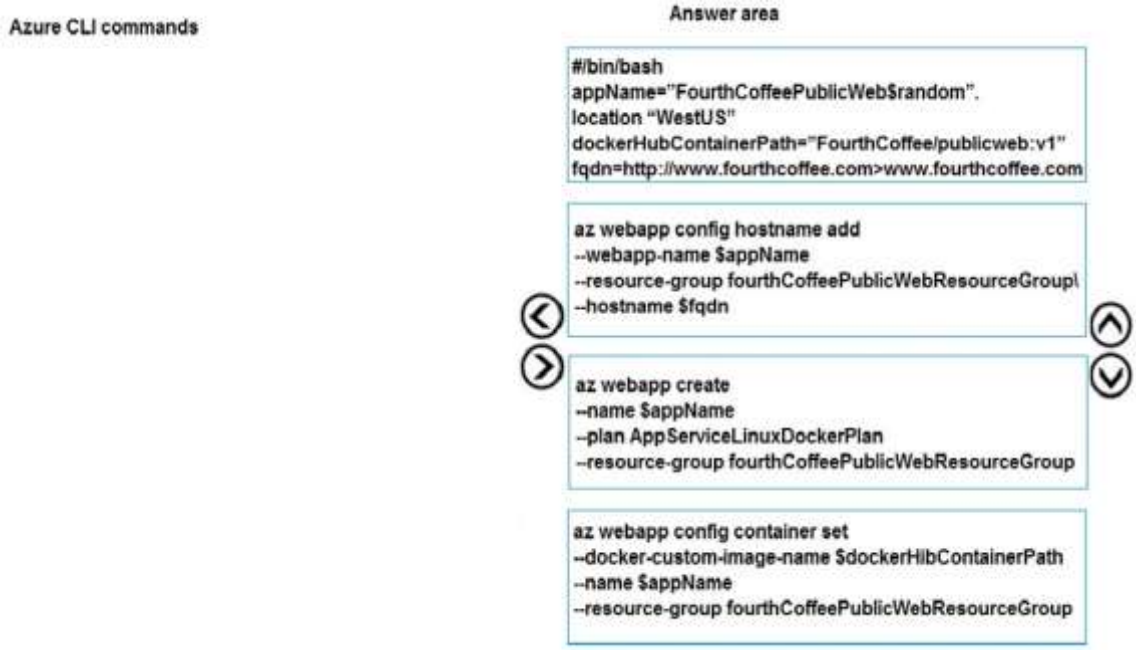
Explanation:
Many-to-many relationships without an entity class to represent the join table are not yet supported. However, you can represent a many-to-many relationship by including an entity class for the join table and mapping two separate one-to-many relationships.

```
protected override void OnModelCreating(ModelBuilder modelBuilder) {
modelBuilder.Entity<PostTag>()
.HasKey(t => new { t.PostId, t.TagId });
modelBuilder.Entity<PostTag>()
.HasOne(pt => pt.Post)
.WithMany(p => p.PostTags)
.HasForeignKey(pt => pt.PostId);
modelBuilder.Entity<PostTag>()
.HasOne(pt => pt.Tag)
.WithMany(t => t.PostTags)
.HasForeignKey(pt => pt.TagId);
}
}
```

QUESTION 110
Drag and Drop Question
Fourth Coffee has an ASP.NET Core web app that runs in Docker. The app is mapped to the www.fourthcoffee.com domain.
Fourth Coffee is migrating this application to Azure.
You need to provision an App Service Web App to host this docker image and map the custom domain to the App Service web app.
A resource group named FourthCoffeePublicWebResourceGroup has been created in the WestUS region that contains an App Service Plan named AppServiceLinuxDockerPlan.
Which order should the CLI commands be used to develop the solution? To answer, move all of the Azure CLI command from the list of commands to the answer area and arrange them in the correct order.



Answer:



Explanation:

Step 1: #bin/bash
The appName is used when the webapp-name is created in step 2.
Step 2: az webapp config hostname add
The webapp-name is used when the webapp is created in step 3.
Step 3: az webapp create
Create a web app. In the Cloud Shell, create a web app in the myAppServicePlan App Service plan with the az webapp create command.
Step : az webapp confing container set
In Create a web app, you specified an image on Docker Hub in the az webapp create command. This is good enough for a public image. To use a private image, you need to configure your Docker account ID and password in your Azure web app.
In the Cloud Shell, follow the az webapp create command with az webapp config container set.
References:
https://docs.microsoft.com/en-us/azure/app-service/containers/tutorial-custom-docker-image

QUESTION 111
Hotspot Question
A company develops a series of mobile games. All games use a single leaderboard service.
You have the following requirements:
- Code should be scalable and allow for growth.
- Each record must consist of a playedId, gameId, score, and time played.
- When users reach a new high score, the system will save the new score using the SaveScore function below.
- Each game is assigned and Id based on the series title.

You have the following code. (Line numbers are included for reference only.)

```
01 public void SaveScore(string gameId, string playerId, int score, long timePlayed)
02 {
03   CloudStorageAccount storageAccount = CloudStorageAccount.Parse(connectionString);
04   CloudTableClient tableClient = storageAccount.CreateCloudTableClinet();
05   CloudTable table = tableClient.GetTableReference("scoreTable");
06   table.CreateIfNotExists();
07   var scoreRecord = new PlayerScore(gameId, playerId, score, timePlayed);
08   TableOperation insertOperation = TableOperation.Insert(scoreRecord);
09   table.Execute(insertOperation);
10  }
11  public class PlayerScore : TableEntity
12 {
13   public PlayerScore(string gameId, string playerId, int score, long timePlayed)
14   {
15     this.PartitionKey = gameId;
16     this.RowKey = playerId;
17   Score = score;
18   TimePlayed = timePlayed;
19   }
20   public int Score { get; set; }
21   public long TimePlayed { get; set; }
22 }
```

You store customer information in an Azure Cosmos database. The following data already exists in the database:

PartitionKey	RowKey	Email
Harp	Walter	wharp@contoso.com
Smith	Steve	ssmith@contoso.com
Smith	Jeff	jsmith@contoso.com

You develop the following code. (Line numbers are included for reference only.)

```
01   CloudTableClient tableClient = account.CreateCloudTableClient();
02   CloudTable table = tableClient.GetTableReference("people");
03   TableQuery<CustomerEntity> query = new TableQuery<CustomerEntity>()
04    .Where(TableQuery.CombineFilters(
05   TableQuery.Generate.And, TableQuery.GenerateFilterCondition(Email, QueryComparisons.Equal, "Smith")
06   TableOperators.And, TableQuery.GenerateFilterCondition(Email, QueryComparisons.Equal,
"ssmith@contoso.com")
07   ));
08   await table.ExecuteQuerySegmentedAsync<CustomerEntity>(query, null);
```

For each of the following statements, select Yes if the statement is true. Otherwise, select No.
NOTE: Each correct selection is worth one point.

Answer Area

	Yes	No
The code will work with Cosmos DB.	○	○
The save score function will update and replace a record if one already exists with the same playerId and gameId.	○	○
The data for the game will be automatically partitioned.	○	○
This code will store the values for the gameId and playerId parameters in the database.	○	○

Answer:

Answer Area

	Yes	No
The code will work with Cosmos DB.	○	○
The save score function will update and replace a record if one already exists with the same playerId and gameId.	○	○
The data for the game will be automatically partitioned.	○	○
This code will store the values for the gameId and playerId parameters in the database.	○	○

Explanation:
Box 1: Yes
Code for CosmosDB, example:
// Parse the connection string and return a reference to the storage account. CloudStorageAccount storageAccount = CloudStorageAccount.Parse(CloudConfigurationManager.GetSetting("StorageConnectionString")); // Create the table client.
CloudTableClient tableClient = storageAccount.CreateCloudTableClient(); // Retrieve a reference to the table.
CloudTable table = tableClient.GetTableReference("people"); // Create the TableOperation object that inserts the customer entity. TableOperation insertOperation = TableOperation.Insert(customer1);
Box 2: No
A new record will always be added as TableOperation.Insert is used, instead of TableOperation.InsertOrReplace.
Box 3: No
No partition key is used.
Box 4: Yes
References:
https://docs.microsoft.com/en-us/azure/cosmos-db/table-storage-how-to-use-dotnet

QUESTION 112
Case Study 5 - Wide World Importers
Background
Wide World Importers is moving all their datacenters to Azure. The company has developed several applications and services to support supply chain operations and would like to leverage serverless computing where possible.
Current environment
Windows Server 2016 virtual machine
This virtual machine (VM) runs Biz Talk Server 2016. The VM runs the following workflows:
- Ocean Transport – This workflow gathers and validates container information including container contents and arrival notices at various shipping ports.
- Inland Transport – This workflow gathers and validates trucking information including fuel usage, number of stops, and routes.

The VM supports the following REST API calls:

- Container API – This API provides container information including weight, contents, and other attributes.
- Location API – This API provides location information regarding shipping ports of call and truck stops.
- Shipping REST API – This API provides shipping information for use and display on the shipping website.

Shipping Data

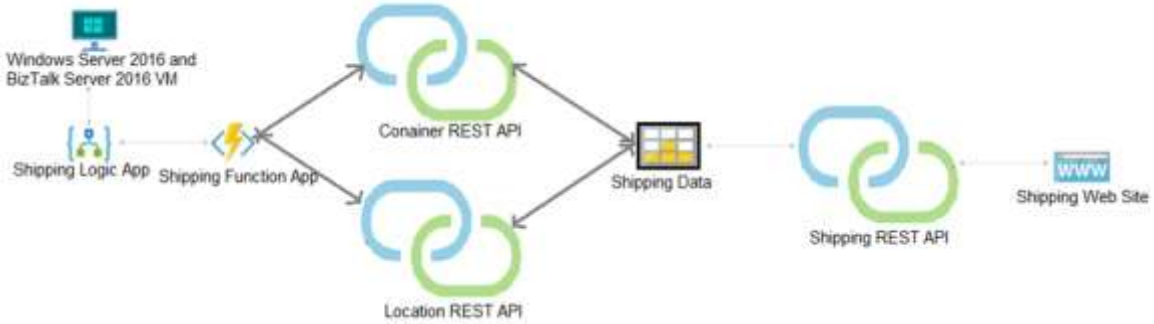
The application uses MongoDB JSON document storage database for all container and transport information.

Shipping Web Site

The site displays shipping container tracking information and container contents. The site is located at http://shipping.wideworldimporters.com

Proposed solution

The on-premises shipping application must be moved to Azure. The VM has been migrated to a new Standard_D16s_v3 Azure VM by using Azure Site Recovery and must remain running in Azure to complete the BizTalk component migrations. You create a Standard_D16s_v3 Azure VM to host BizTalk Server. The Azure architecture diagram for the proposed solution is shown below:



Shipping Logic App

The Shipping Logic app must meet the following requirements:

- Support the ocean transport and inland transport workflows by using a Logic App.
- Support industry-standard protocol X12 message format for various messages including vessel content details and arrival notices.
- Secure resources to the corporate VNet and use dedicated storage resources with a fixed costing model.
- Maintain on-premises connectivity to support legacy applications and final BizTalk migrations.

Shipping Function app

Implement secure function endpoints by using app-level security and include Azure Active Directory (Azure AD).

REST APIs

The REST API's that support the solution must meet the following requirements:

- Secure resources to the corporate VNet.
- Allow deployment to a testing location within Azure while not incurring additional costs.
- Automatically scale to double capacity during peak shipping times while not causing application downtime.
- Minimize costs when selecting an Azure payment model.

Shipping data

Data migration from on-premises to Azure must minimize costs and downtime.

Shipping website

Use Azure Content Delivery Network (CDN) and ensure maximum performance for dynamic content while minimizing latency and costs.

Issues

Windows Server 2016 VM

The VM shows high network latency, jitter, and high CPU utilization. The VM is critical and has not been backed up in the past. The VM must enable a quick restore from a 7-day snapshot to include in-place restore of disks in case of failure.

Shipping website and REST APIs

The following error message displays while you are testing the website:

```
Failed to load http://test-shippingapi.wideworldimporters.com/: No 'Access-Control-Allow-Origin'
header is present on the requested resource. Origin 'http://testwideworldimporters.com/' is
therefore not allowed access.
```

You need to support the requirements for the Shipping Logic App.

What should you use?

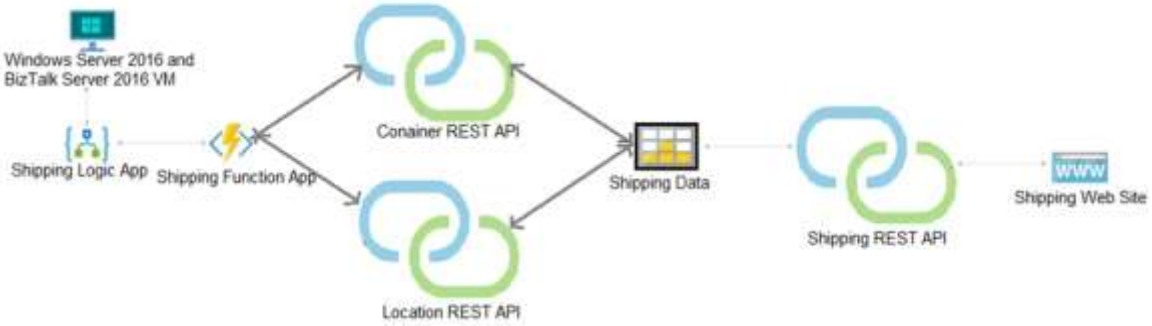
A. Azure Active Directory Application Proxy
B. Point-to-Site (P2S) VPN connection
C. Site-to-Site (S2S) VPN connection
D. On-premises Data Gateway

Answer: D
Explanation:
Before you can connect to on-premises data sources from Azure Logic Apps, download and install the onpremises data gateway on a local computer. The gateway works as a bridge that provides quick data transfer and encryption between data sources on premises (not in the cloud) and your logic apps.
The gateway supports BizTalk Server 2016.
Note: Microsoft have now fully incorporated the Azure BizTalk Services capabilities into Logic Apps and Azure App Service Hybrid Connections.
Logic Apps Enterprise Integration pack bring some of the enterprise B2B capabilities like AS2 and X12, EDI standards support
Scenario: The Shipping Logic app must meet the following requirements:
Support the ocean transport and inland transport workflows by using a Logic App.
Support industry standard protocol X12 message format for various messages including vessel content details and arrival notices.
Secure resources to the corporate VNet and use dedicated storage resources with a fixed costing model.
Maintain on-premises connectivity to support legacy applications and final BizTalk migrations.
References:
https://docs.microsoft.com/en-us/azure/logic-apps/logic-apps-gateway-install

QUESTION 113
Case Study 5 - Wide World Importers
Background
Wide World Importers is moving all their datacenters to Azure. The company has developed several applications and services to support supply chain operations and would like to leverage serverless computing where possible.
Current environment
Windows Server 2016 virtual machine
This virtual machine (VM) runs Biz Talk Server 2016. The VM runs the following workflows:
- Ocean Transport – This workflow gathers and validates container information including container contents and arrival notices at various shipping ports.
- Inland Transport – This workflow gathers and validates trucking information including fuel usage, number of stops, and routes.

The VM supports the following REST API calls:
- Container API – This API provides container information including weight, contents, and other attributes.
- Location API – This API provides location information regarding shipping ports of call and truck stops.
- Shipping REST API – This API provides shipping information for use and display on the shipping website.

Shipping Data
The application uses MongoDB JSON document storage database for all container and transport information.
Shipping Web Site
The site displays shipping container tracking information and container contents. The site is located at http://shipping.wideworldimporters.com
Proposed solution
The on-premises shipping application must be moved to Azure. The VM has been migrated to a new Standard_D16s_v3 Azure VM by using Azure Site Recovery and must remain running in Azure to complete the BizTalk component migrations. You create a Standard_D16s_v3 Azure VM to host BizTalk Server. The Azure architecture diagram for the proposed solution is shown below:



Shipping Logic App
The Shipping Logic app must meet the following requirements:
- Support the ocean transport and inland transport workflows by using a Logic App.
- Support industry-standard protocol X12 message format for various messages including vessel content details and arrival notices.
- Secure resources to the corporate VNet and use dedicated storage resources with a fixed costing model.
- Maintain on-premises connectivity to support legacy applications and final BizTalk migrations.

Shipping Function app
Implement secure function endpoints by using app-level security and include Azure Active Directory (Azure AD).
REST APIs
The REST API's that support the solution must meet the following requirements:
- Secure resources to the corporate VNet.
- Allow deployment to a testing location within Azure while not incurring additional costs.
- Automatically scale to double capacity during peak shipping times while not causing application downtime.
- Minimize costs when selecting an Azure payment model.

Shipping data
Data migration from on-premises to Azure must minimize costs and downtime.
Shipping website
Use Azure Content Delivery Network (CDN) and ensure maximum performance for dynamic content while minimizing latency and costs.
Issues
Windows Server 2016 VM
The VM shows high network latency, jitter, and high CPU utilization. The VM is critical and has not been backed up in the past. The VM must enable a quick restore from a 7-day snapshot to include in-place restore of disks in case of failure.
Shipping website and REST APIs
The following error message displays while you are testing the website:

```
Failed to load http://test-shippingapi.wideworldimporters.com/: No 'Access-Control-Allow-Origin'
header is present on the requested resource. Origin 'http://testwideworldimporters.com/' is
therefore not allowed access.
```

You need to migrate on-premises shipping data to Azure.
What should you use?

A. Azure Cosmos DB Data Migration tool (dt.exe)
B. Azure Database Migration service
C. AzCopy
D. Azure Migrate

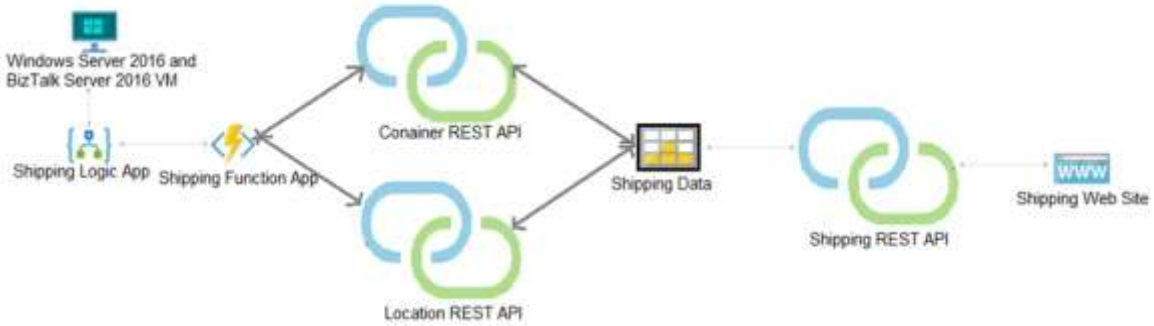
Answer: B
Explanation:
Migrate from on-premises or cloud implementations of MongoDB to Azure Cosmos DB with minimal downtime by using Azure Database Migration Service. Perform resilient migrations of MongoDB data at scale and with high reliability.
Scenario: Data migration from on-premises to Azure must minimize costs and downtime.
The application uses MongoDB JSON document storage database for all container and transport information.
References:

https://azure.microsoft.com/en-us/updates/mongodb-to-azure-cosmos-db-online-and-offline-migrationsare-now-available/

QUESTION 114
Case Study 5 - Wide World Importers
Background
Wide World Importers is moving all their datacenters to Azure. The company has developed several applications and services to support supply chain operations and would like to leverage serverless computing where possible.
Current environment
Windows Server 2016 virtual machine
This virtual machine (VM) runs Biz Talk Server 2016. The VM runs the following workflows:
- Ocean Transport – This workflow gathers and validates container information including container contents and arrival notices at various shipping ports.
- Inland Transport – This workflow gathers and validates trucking information including fuel usage, number of stops, and routes.

The VM supports the following REST API calls:
- Container API – This API provides container information including weight, contents, and other attributes.
- Location API – This API provides location information regarding shipping ports of call and truck stops.
- Shipping REST API – This API provides shipping information for use and display on the shipping website.

Shipping Data
The application uses MongoDB JSON document storage database for all container and transport information.
Shipping Web Site
The site displays shipping container tracking information and container contents. The site is located at http://shipping.wideworldimporters.com
Proposed solution
The on-premises shipping application must be moved to Azure. The VM has been migrated to a new Standard_D16s_v3 Azure VM by using Azure Site Recovery and must remain running in Azure to complete the BizTalk component migrations. You create a Standard_D16s_v3 Azure VM to host BizTalk Server. The Azure architecture diagram for the proposed solution is shown below:



Shipping Logic App
The Shipping Logic app must meet the following requirements:
- Support the ocean transport and inland transport workflows by using a Logic App.
- Support industry-standard protocol X12 message format for various messages including vessel content details and arrival notices.
- Secure resources to the corporate VNet and use dedicated storage resources with a fixed costing model.
- Maintain on-premises connectivity to support legacy applications and final BizTalk migrations.

Shipping Function app
Implement secure function endpoints by using app-level security and include Azure Active Directory (Azure AD).
REST APIs
The REST API's that support the solution must meet the following requirements:
- Secure resources to the corporate VNet.
- Allow deployment to a testing location within Azure while not incurring additional costs.
- Automatically scale to double capacity during peak shipping times while not causing application downtime.
- Minimize costs when selecting an Azure payment model.

Shipping data
Data migration from on-premises to Azure must minimize costs and downtime.

Shipping website
Use Azure Content Delivery Network (CDN) and ensure maximum performance for dynamic content while minimizing latency and costs.
Issues
Windows Server 2016 VM
The VM shows high network latency, jitter, and high CPU utilization. The VM is critical and has not been backed up in the past. The VM must enable a quick restore from a 7-day snapshot to include in-place restore of disks in case of failure.
Shipping website and REST APIs
The following error message displays while you are testing the website:

```
Failed to load http://test-shippingapi.wideworldimporters.com/: No 'Access-Control-Allow-Origin'
header is present on the requested resource. Origin 'http://testwideworldimporters.com/' is
therefore not allowed access.
```

You need to secure the Shipping Logic App.
What should you use?

A. Azure App Service Environment (ASE)
B. Azure AD B2B integration
C. Integration Service Environment (ISE)
D. VNet service endpoint

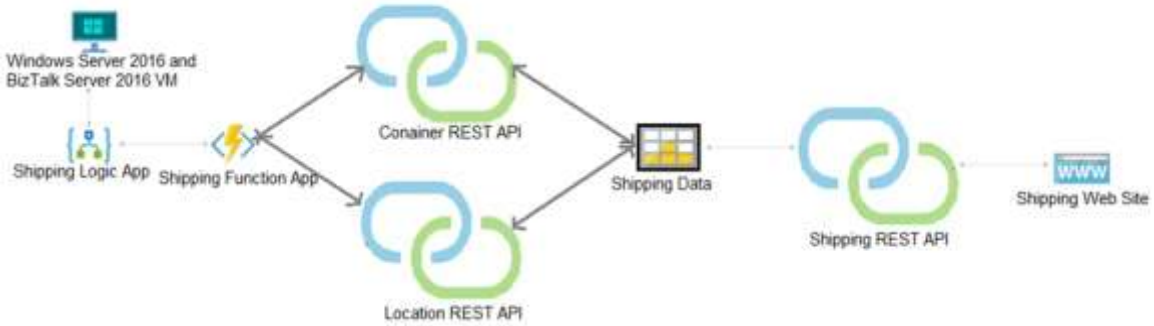
Answer: C
Explanation:
Scenario: The Shipping Logic App requires secure resources to the corporate VNet and use dedicated storage resources with a fixed costing model.
You can access to Azure Virtual Network resources from Azure Logic Apps by using integration service environments (ISEs).
Sometimes, your logic apps and integration accounts need access to secured resources, such as virtual machines (VMs) and other systems or services, that are inside an Azure virtual network. To set up this access, you can create an integration service environment (ISE) where you can run your logic apps and create your integration accounts.
References:
https://docs.microsoft.com/en-us/azure/logic-apps/connect-virtual-network-vnet-isolated-environmentoverview

QUESTION 115
Case Study 5 - Wide World Importers
Background
Wide World Importers is moving all their datacenters to Azure. The company has developed several applications and services to support supply chain operations and would like to leverage serverless computing where possible.
Current environment
Windows Server 2016 virtual machine
This virtual machine (VM) runs Biz Talk Server 2016. The VM runs the following workflows:
- Ocean Transport – This workflow gathers and validates container information including container contents and arrival notices at various shipping ports.
- Inland Transport – This workflow gathers and validates trucking information including fuel usage, number of stops, and routes.

The VM supports the following REST API calls:
- Container API – This API provides container information including weight, contents, and other attributes.
- Location API – This API provides location information regarding shipping ports of call and truck stops.
- Shipping REST API – This API provides shipping information for use and display on the shipping website.

Shipping Data
The application uses MongoDB JSON document storage database for all container and transport information.
Shipping Web Site
The site displays shipping container tracking information and container contents. The site is located at http://shipping.wideworldimporters.com
Proposed solution
The on-premises shipping application must be moved to Azure. The VM has been migrated to a new Standard_D16s_v3 Azure VM by using Azure Site Recovery and must remain running in Azure to complete the BizTalk

component migrations. You create a Standard_D16s_v3 Azure VM to host BizTalk Server. The Azure architecture diagram for the proposed solution is shown below:



Shipping Logic App

The Shipping Logic app must meet the following requirements:

- Support the ocean transport and inland transport workflows by using a Logic App.
- Support industry-standard protocol X12 message format for various messages including vessel content details and arrival notices.
- Secure resources to the corporate VNet and use dedicated storage resources with a fixed costing model.
- Maintain on-premises connectivity to support legacy applications and final BizTalk migrations.

Shipping Function app

Implement secure function endpoints by using app-level security and include Azure Active Directory (Azure AD).

REST APIs

The REST API's that support the solution must meet the following requirements:

- Secure resources to the corporate VNet.
- Allow deployment to a testing location within Azure while not incurring additional costs.
- Automatically scale to double capacity during peak shipping times while not causing application downtime.
- Minimize costs when selecting an Azure payment model.

Shipping data

Data migration from on-premises to Azure must minimize costs and downtime.

Shipping website

Use Azure Content Delivery Network (CDN) and ensure maximum performance for dynamic content while minimizing latency and costs.

Issues

Windows Server 2016 VM

The VM shows high network latency, jitter, and high CPU utilization. The VM is critical and has not been backed up in the past. The VM must enable a quick restore from a 7-day snapshot to include in-place restore of disks in case of failure.

Shipping website and REST APIs

The following error message displays while you are testing the website:

```
Failed to load http://test-shippingapi.wideworldimporters.com/: No 'Access-Control-Allow-Origin'
header is present on the requested resource. Origin 'http://testwideworldimporters.com/' is
therefore not allowed access.
```

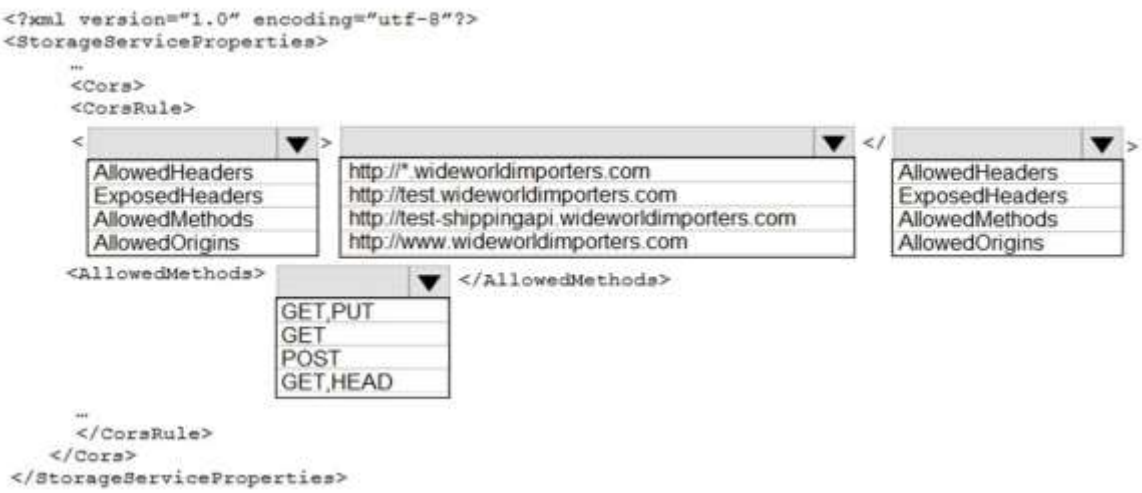
Hotspot Question

You need to resolve the Shipping web site error.

How should you configure the Azure Table Storage service? To answer, select the appropriate options in the answer area.

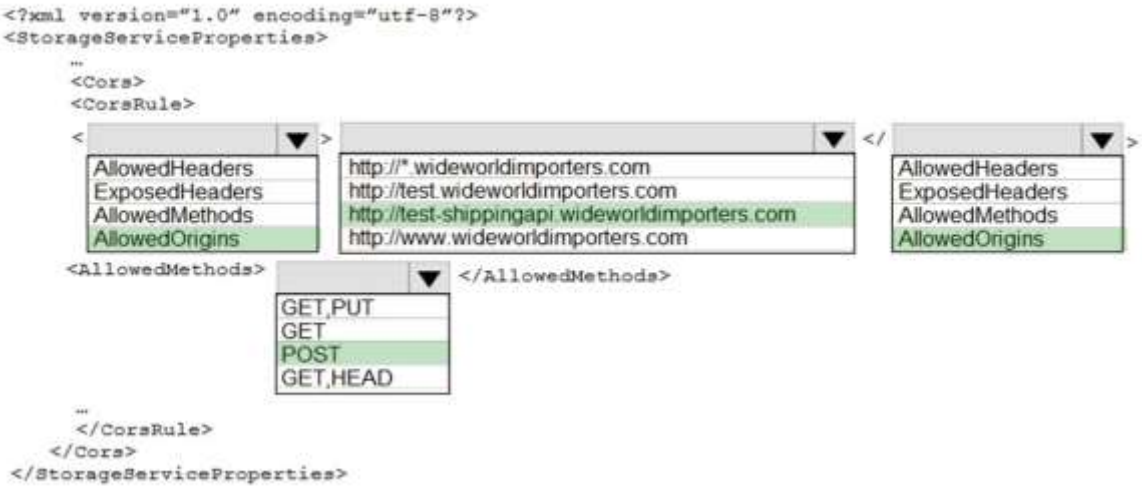
NOTE: Each correct selection is worth one point.

Answer Area



Answer:

Answer Area



Explanation:

Box 1: AllowedOrigins

A CORS request will fail if Access-Control-Allow-Origin is missing.

Scenario:

The following error message displays while you are testing the website:

```
Failed to load http://test-shippingapi.wideworldimporters.com/: No 'Access-Cc
header is present on the requested resource. Origin 'http://testwideworldimpc
therefore not allowed access.
```

Box 2: http://test-shippingapi.wideworldimporters.com

Syntax: Access-Control-Allow-Origin: *

Access-Control-Allow-Origin: <origin>

Access-Control-Allow-Origin: null

<origin> Specifies an origin. Only a single origin can be specified.

Box 3: AllowedOrigins

Box 4: POST

The only allowed methods are GET, HEAD, and POST. In this case POST is used.

"<Corsrule>" "allowedmethods" Failed to load no "Access-control-Origin" header is present

References:

https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Access-Control-Allow-Origin

QUESTION 116

Case Study 5 - Wide World Importers

Background

Wide World Importers is moving all their datacenters to Azure. The company has developed several applications and services to support supply chain operations and would like to leverage serverless computing where possible.

Current environment

Windows Server 2016 virtual machine

This virtual machine (VM) runs Biz Talk Server 2016. The VM runs the following workflows:

- Ocean Transport – This workflow gathers and validates container information including container contents and arrival notices at various shipping ports.
- Inland Transport – This workflow gathers and validates trucking information including fuel usage, number of stops, and routes.

The VM supports the following REST API calls:

- Container API – This API provides container information including weight, contents, and other attributes.
- Location API – This API provides location information regarding shipping ports of call and truck stops.
- Shipping REST API – This API provides shipping information for use and display on the shipping website.

Shipping Data

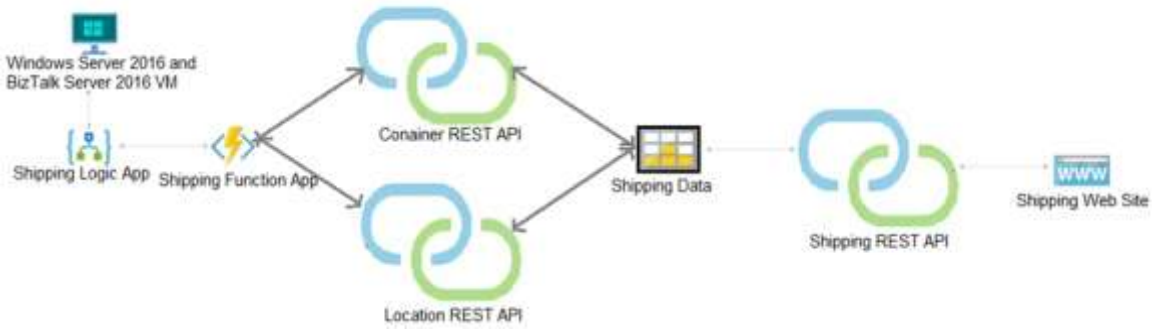
The application uses MongoDB JSON document storage database for all container and transport information.

Shipping Web Site

The site displays shipping container tracking information and container contents. The site is located at http://shipping.wideworldimporters.com

Proposed solution

The on-premises shipping application must be moved to Azure. The VM has been migrated to a new Standard_D16s_v3 Azure VM by using Azure Site Recovery and must remain running in Azure to complete the BizTalk component migrations. You create a Standard_D16s_v3 Azure VM to host BizTalk Server. The Azure architecture diagram for the proposed solution is shown below:



Shipping Logic App

The Shipping Logic app must meet the following requirements:

- Support the ocean transport and inland transport workflows by using a Logic App.
- Support industry-standard protocol X12 message format for various messages including vessel content details and arrival notices.
- Secure resources to the corporate VNet and use dedicated storage resources with a fixed costing model.
- Maintain on-premises connectivity to support legacy applications and final BizTalk migrations.

Shipping Function app

Implement secure function endpoints by using app-level security and include Azure Active Directory (Azure AD).

REST APIs

The REST API's that support the solution must meet the following requirements:

- Secure resources to the corporate VNet.
- Allow deployment to a testing location within Azure while not incurring additional costs.
- Automatically scale to double capacity during peak shipping times while not causing application downtime.
- Minimize costs when selecting an Azure payment model.

Shipping data

Data migration from on-premises to Azure must minimize costs and downtime.

Shipping website

Use Azure Content Delivery Network (CDN) and ensure maximum performance for dynamic content while minimizing latency and costs.

Issues

Windows Server 2016 VM

The VM shows high network latency, jitter, and high CPU utilization. The VM is critical and has not been backed up in the past. The VM must enable a quick restore from a 7-day snapshot to include in-place restore of disks in case of failure.

Shipping website and REST APIs

The following error message displays while you are testing the website:

```
Failed to load http://test-shippingapi.wideworldimporters.com/: No 'Access-Control-Allow-Origin'
header is present on the requested resource. Origin 'http://testwideworldimporters.com/' is
therefore not allowed access.
```

Hotspot Question

You need to secure the Shipping Function app.

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

NOTE: Each correct selection is worth one point.

Answer Area

Setting	Value
Authorization level	Function / Anonymous / Admin
User claims	JSON Web Token (JWT) / Shared Access Signature (SAS) token / API Key
Trigger type	blob / HTTP / queue / timer

Answer:

Answer Area

Setting	Value
Authorization level	Function / Anonymous / Admin
User claims	JSON Web Token (JWT) / Shared Access Signature (SAS) token / API Key
Trigger type	blob / HTTP / queue / timer

Explanation:
Scenario: Shipping Function app: Implement secure function endpoints by using app-level security and include Azure Active Directory (Azure AD).
Box 1: Function
Box 2: JSON based Token (JWT)
Azure AD uses JSON based tokens (JWTs) that contain claims
Box 3: HTTP
How a web app delegates sign-in to Azure AD and obtains a token User authentication happens via the browser. The OpenID protocol uses standard HTTP protocol messages.
References:
https://docs.microsoft.com/en-us/azure/active-directory/develop/authentication-scenarios