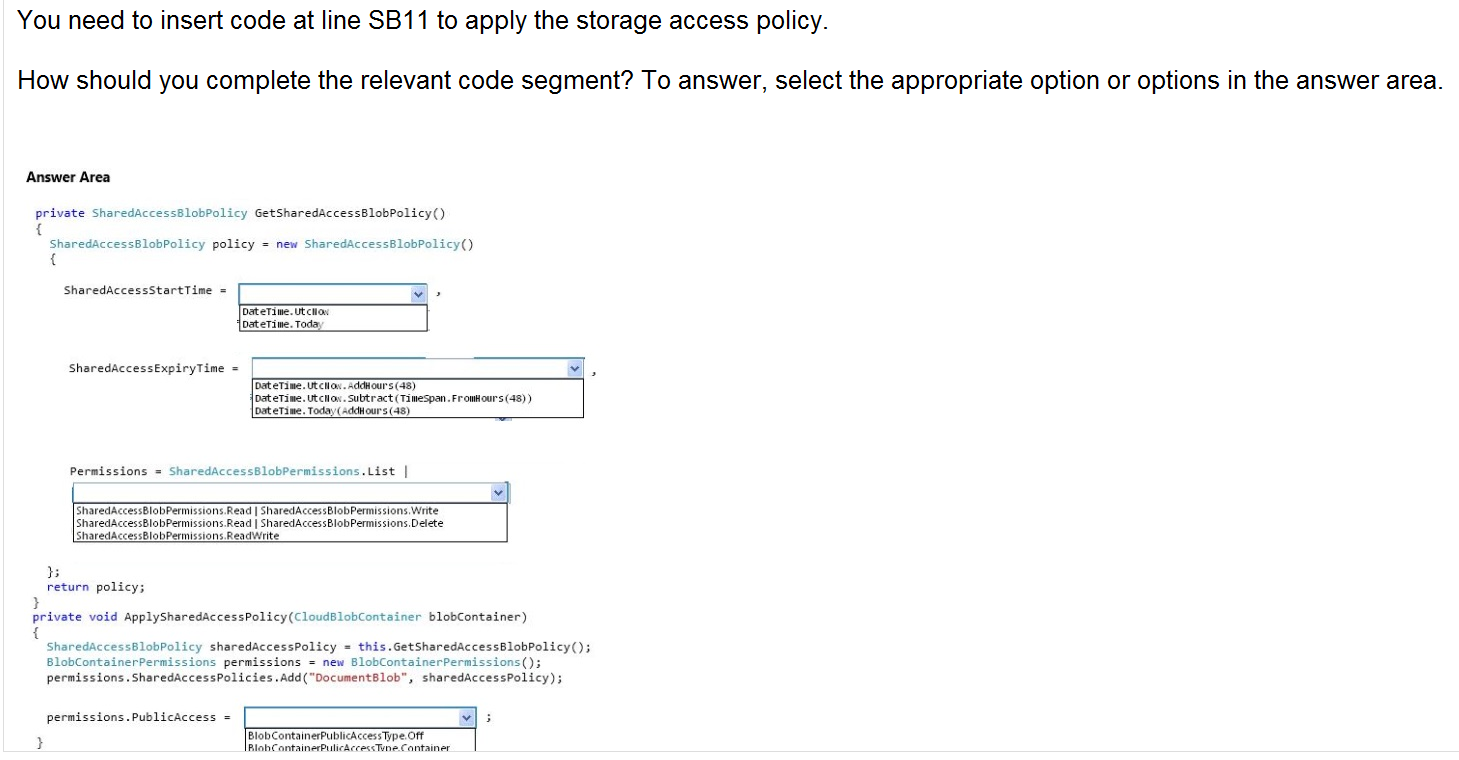


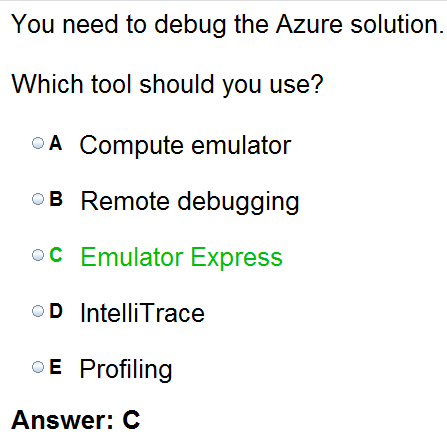
1)



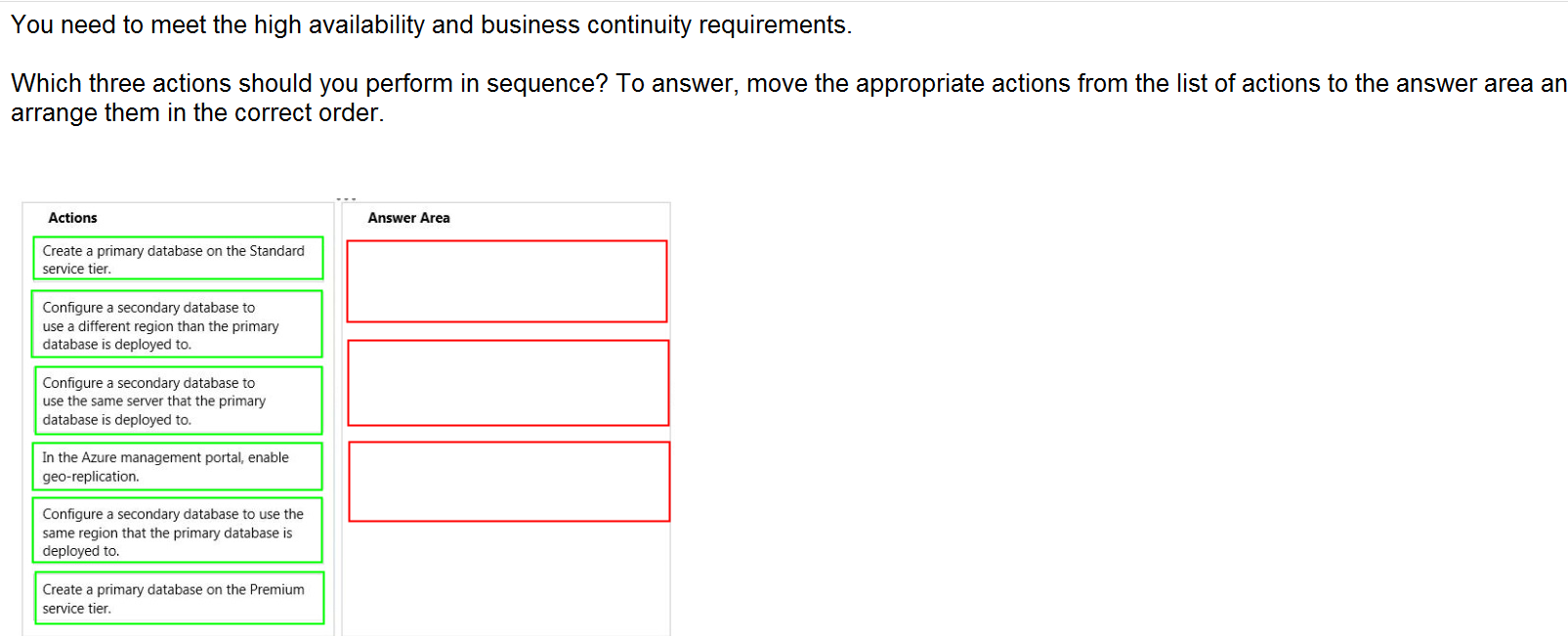


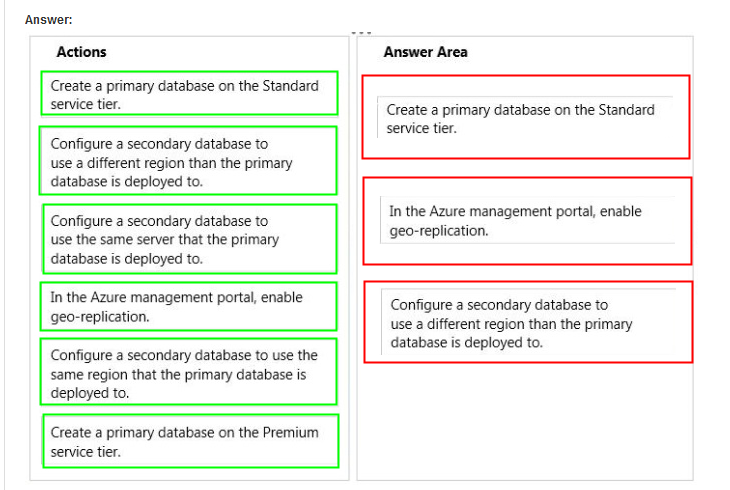


2)

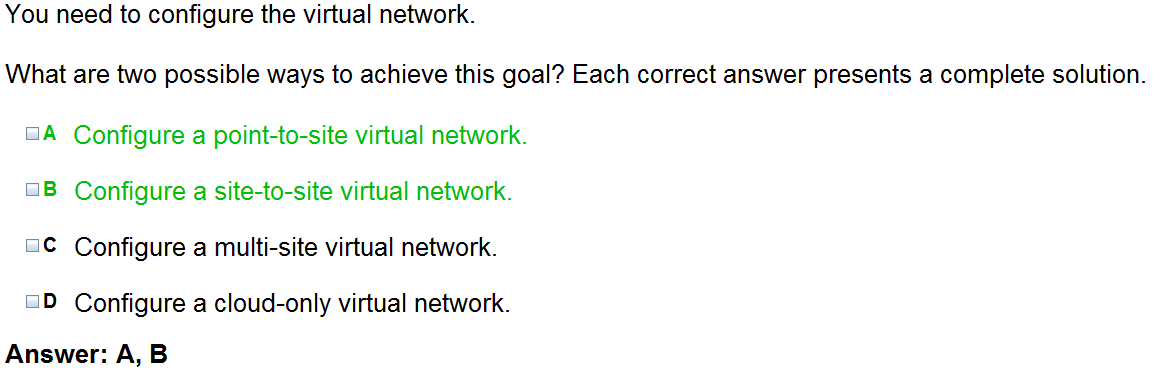


3)

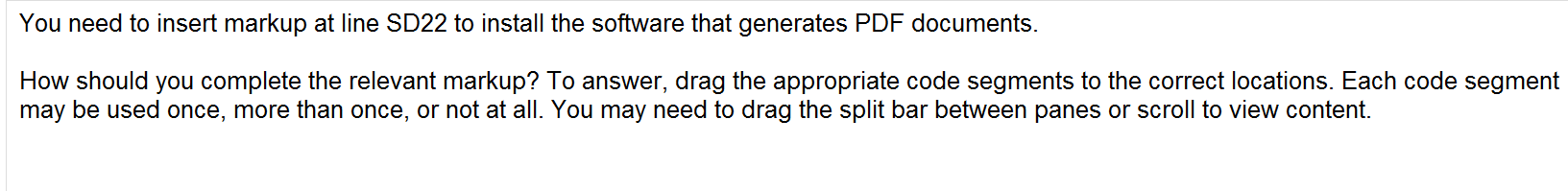


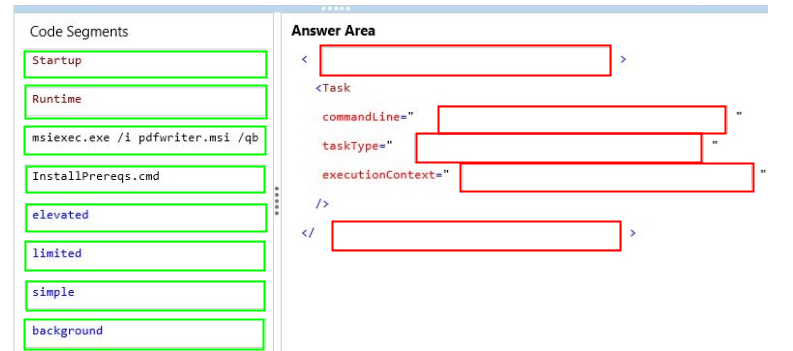


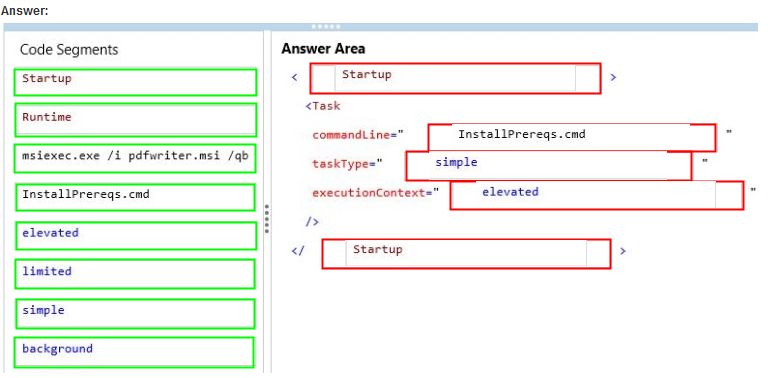
4)



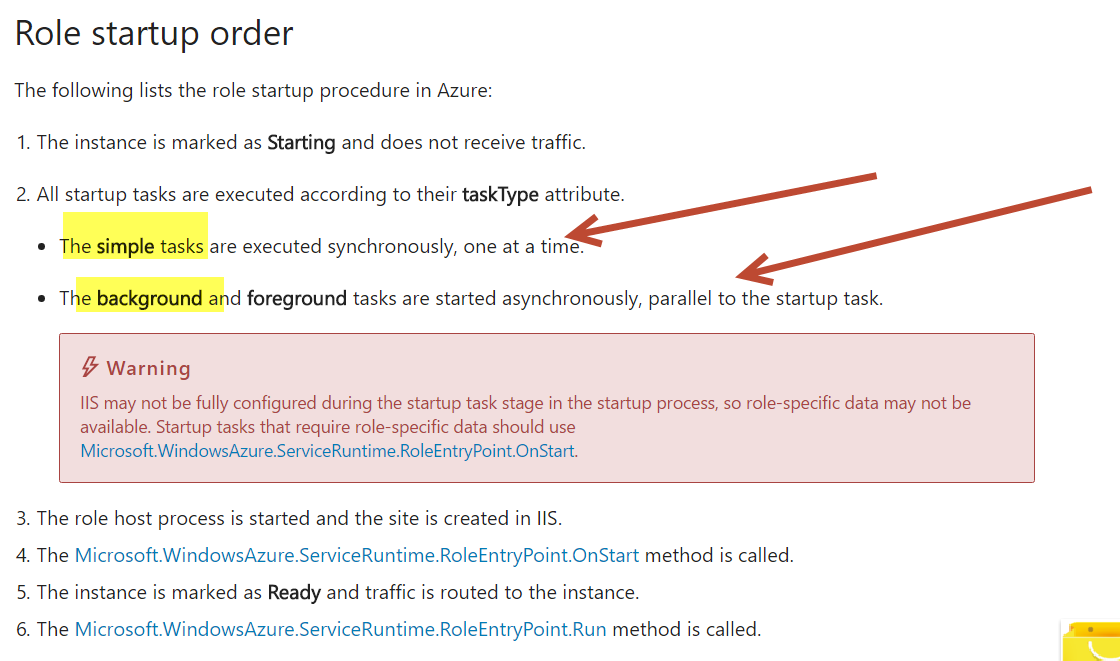
5)

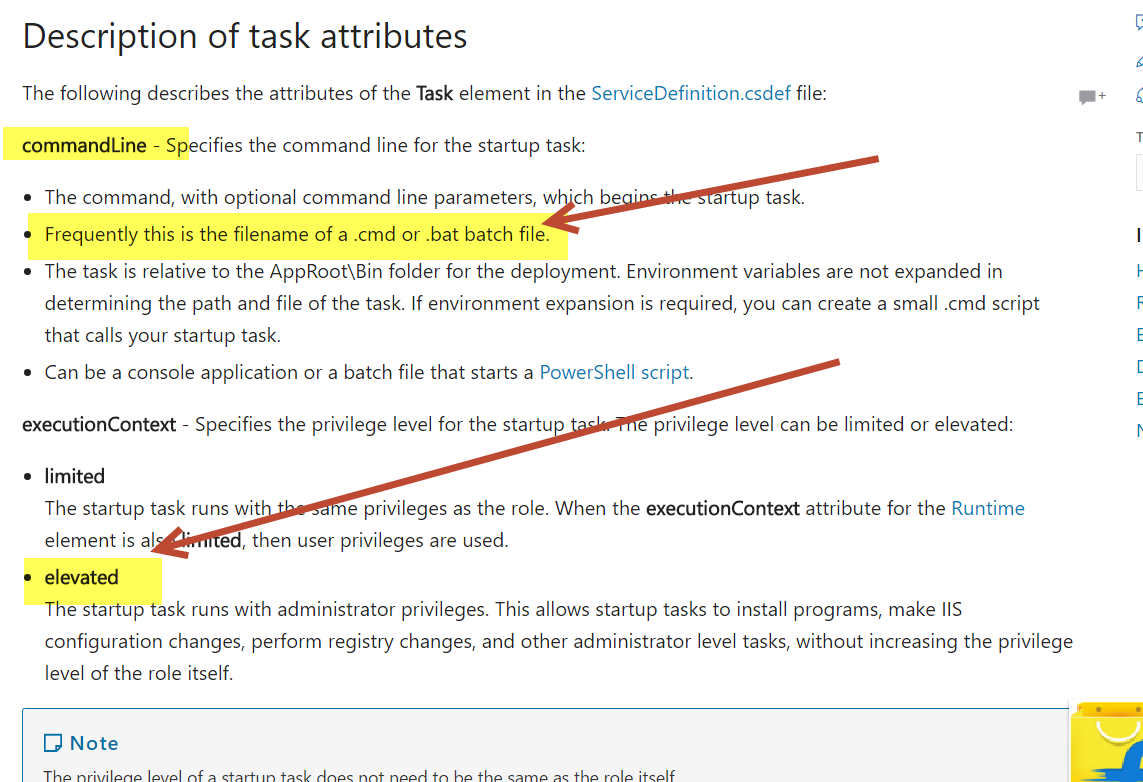






How to configure and run startup tasks for a cloud service

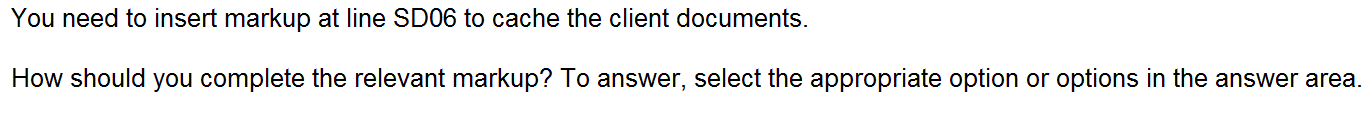


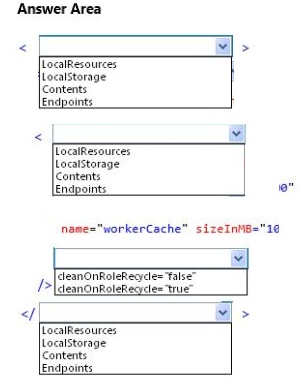


Also in the Definition file

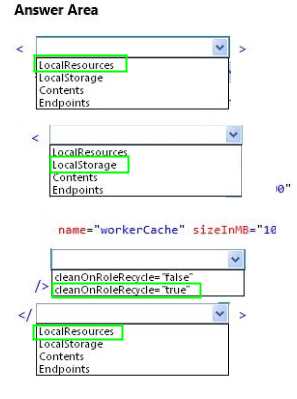


6)



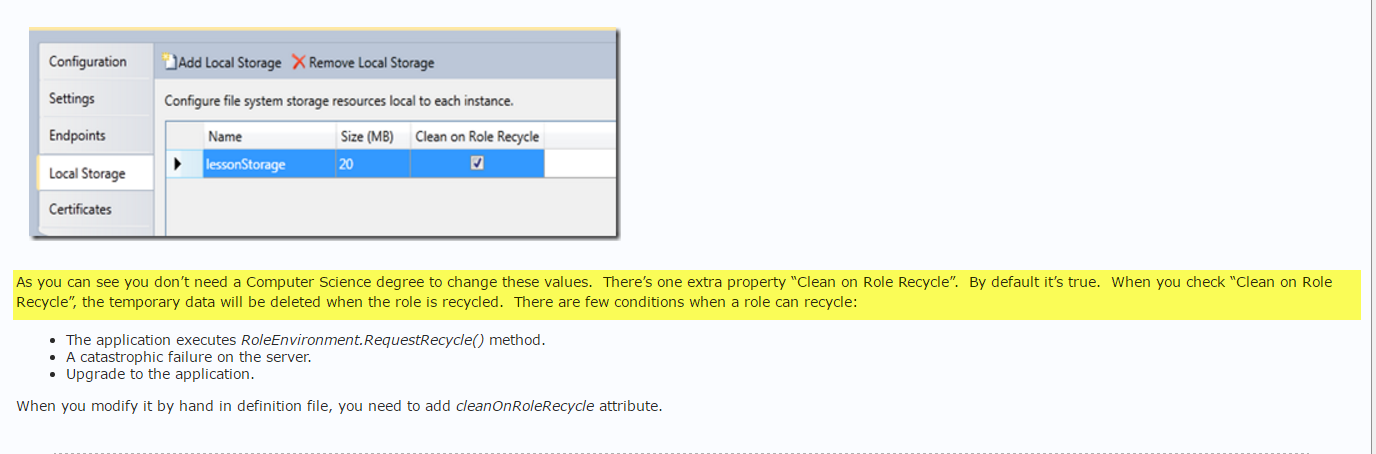


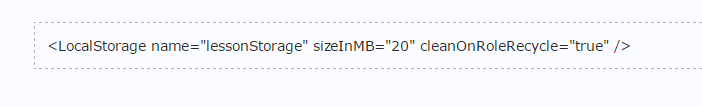
Ans



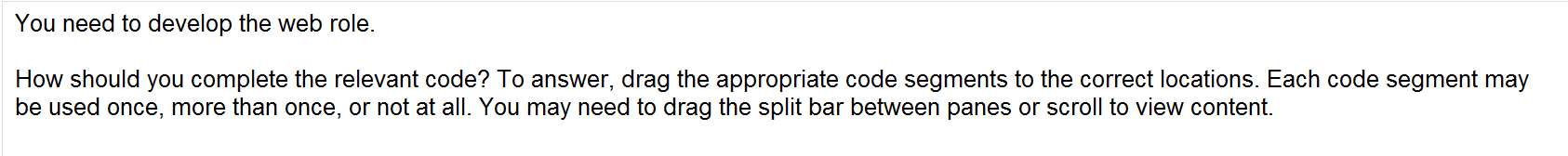
What is Local Storage –

There’s also Local Storage and that is what I want to talk about. Local Storage is a temporary file system storage area.  If you didn’t get from previous  sentence, it’s **a temporary storage**.  Local Storage also available to a single role instance and cannot be shared across multiple role instances.  What does it mean for you?  It means that you should not store any data that you want to be persistent for a long time.  If you upgrade a role instance or your instance dies, you will never see the data in the Local Storage again.  All important data you should store in Blob, Table Storage or SQL Azure.  Advantage of Local Storage is that it’s faster than other storage mechanisms mentioned earlier.  Blobs, Table Storage and SQL Azure are located on different servers than your Azure VM.  Local Storage is on the same VM where the role instance is running, therefore you’re going to have less latency.





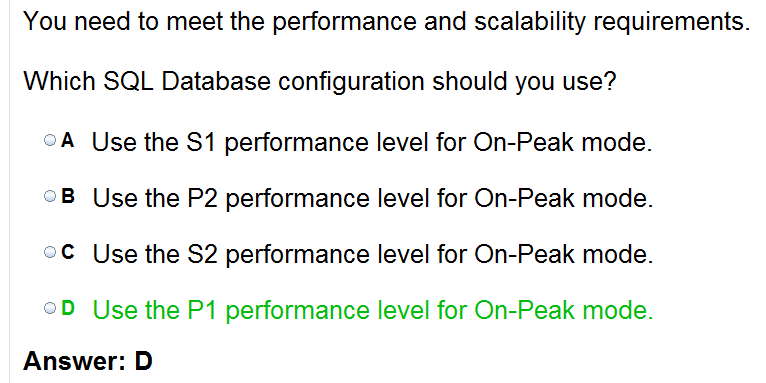
7)



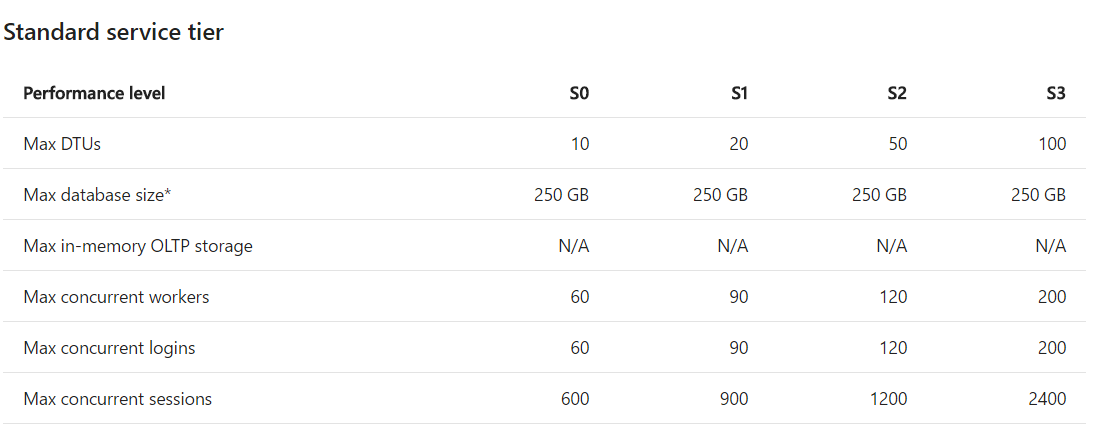


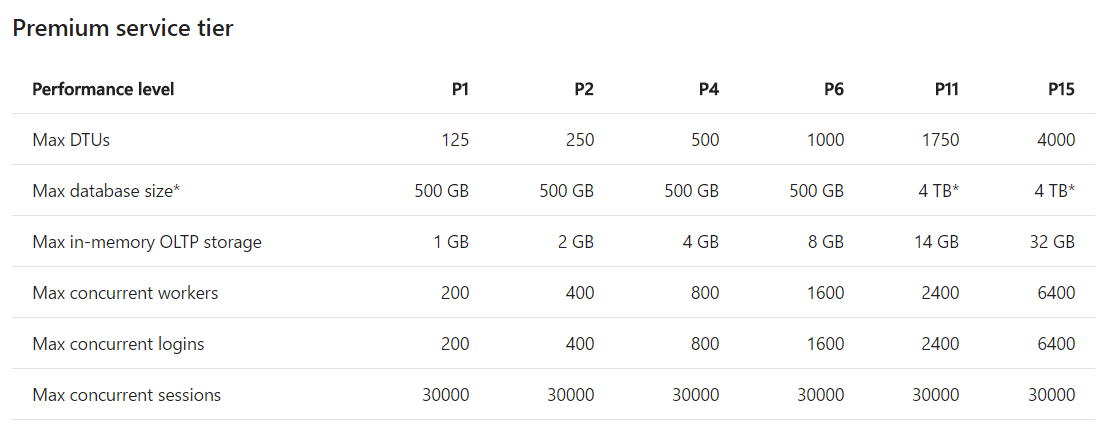


**8)**

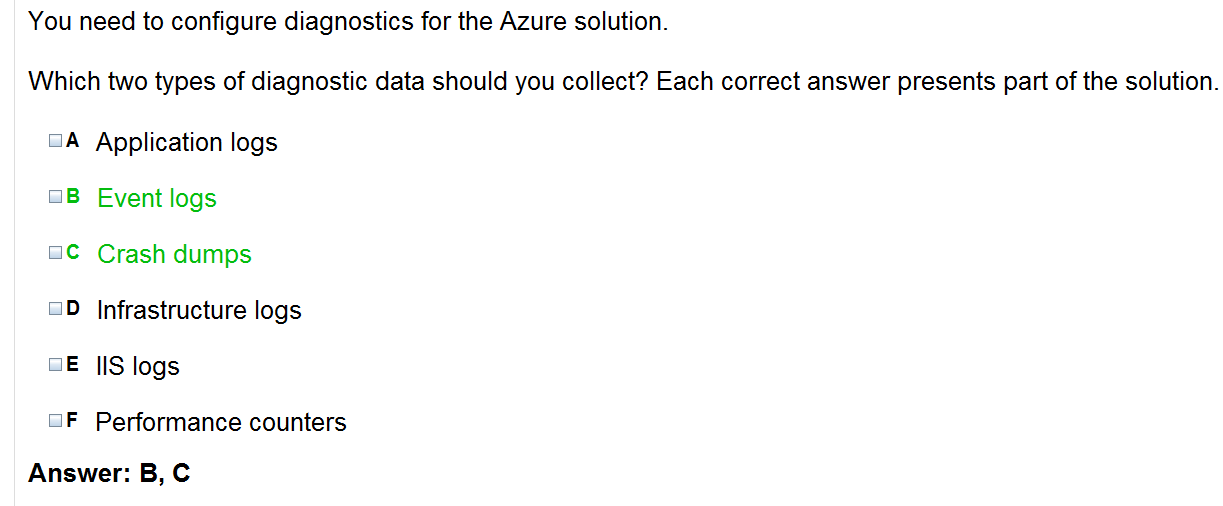


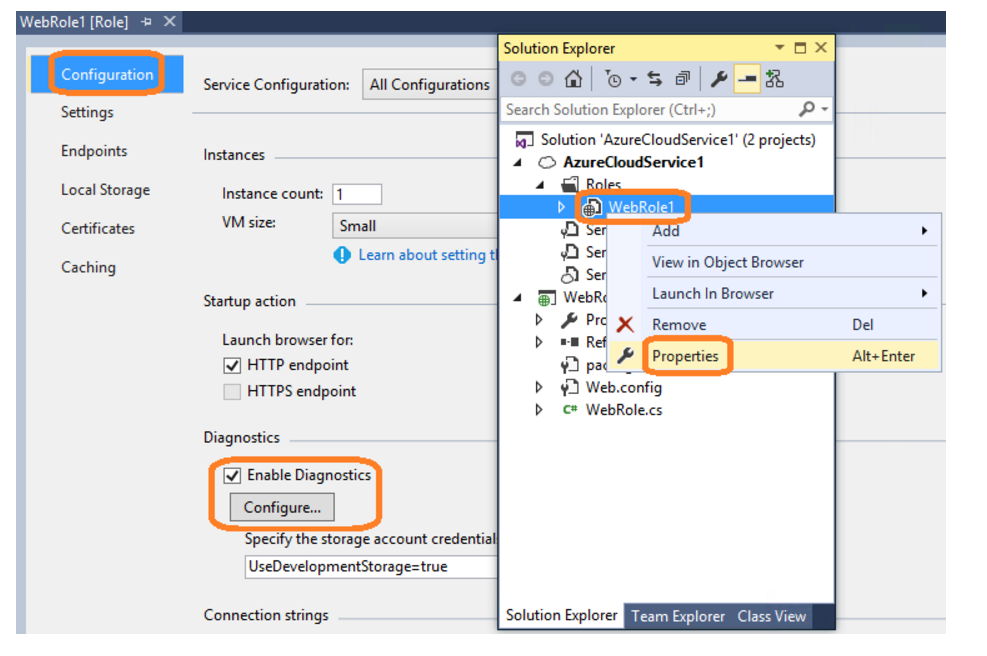
[**https://docs.microsoft.com/en-us/azure/sql-database/sql-database-resource-limits**](https://docs.microsoft.com/en-us/azure/sql-database/sql-database-resource-limits)

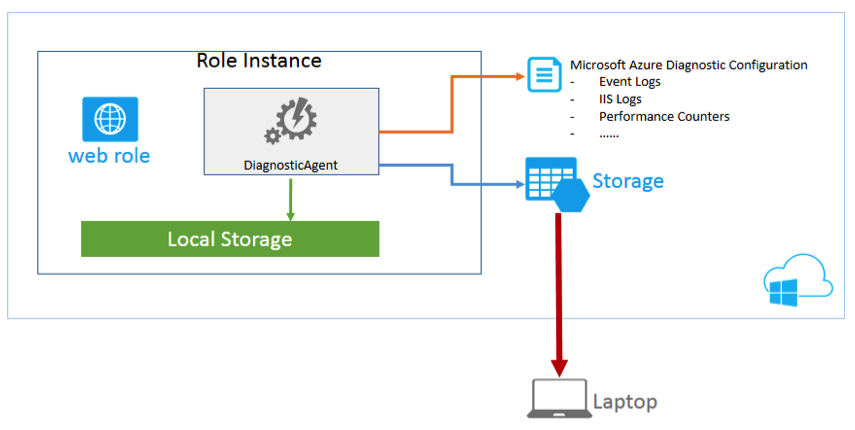




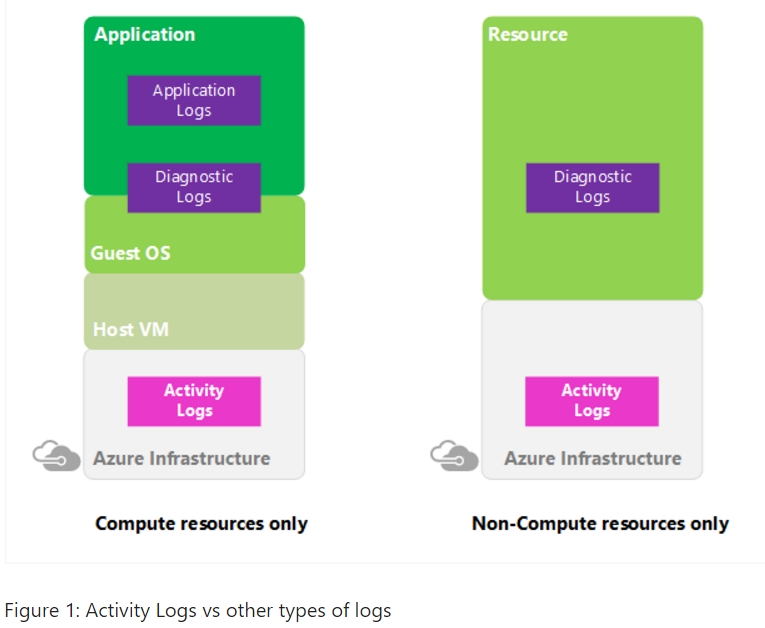
**9)**



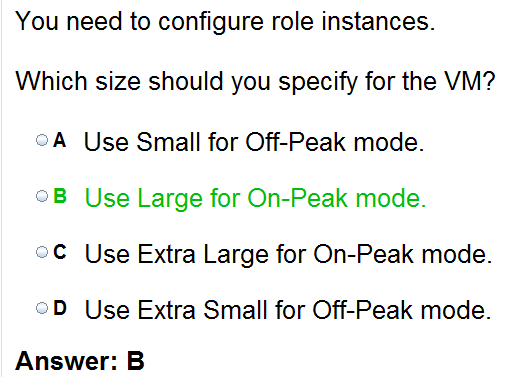




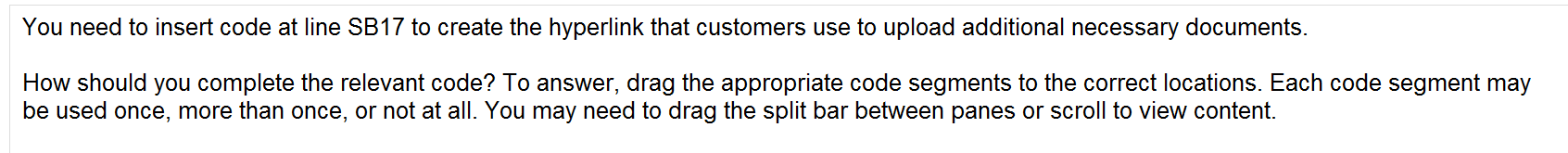
[**https://docs.microsoft.com/en-us/azure/monitoring-and-diagnostics/monitoring-overview-activity-logs**](https://docs.microsoft.com/en-us/azure/monitoring-and-diagnostics/monitoring-overview-activity-logs)

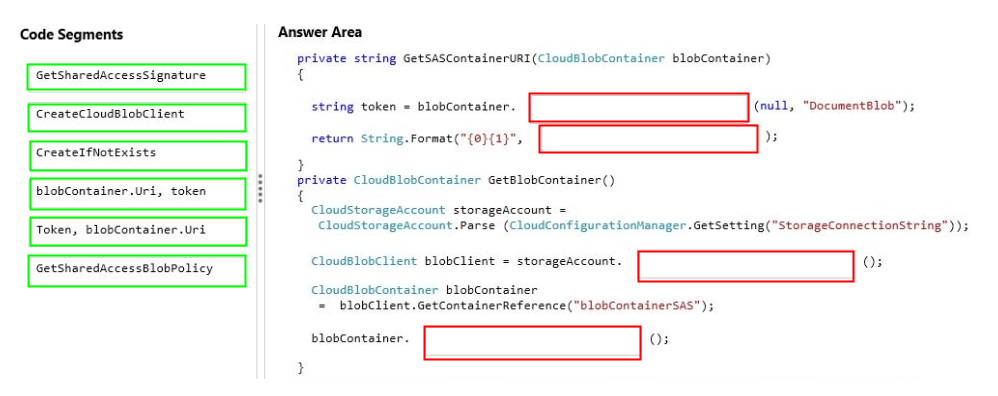


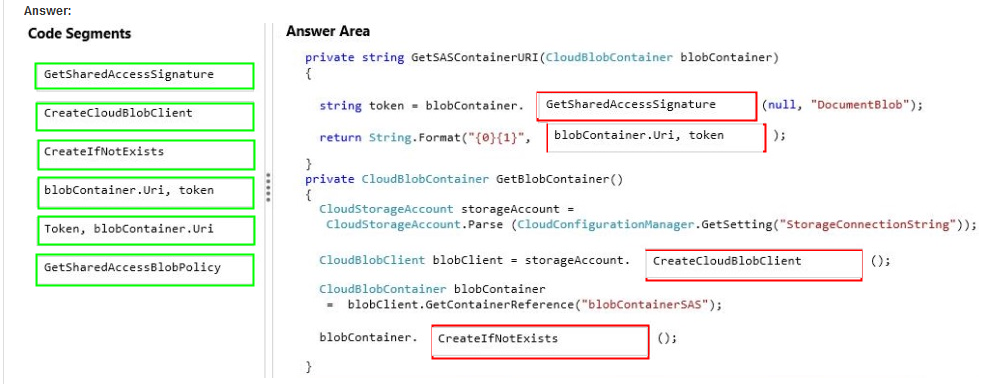
**10)**



**11)**

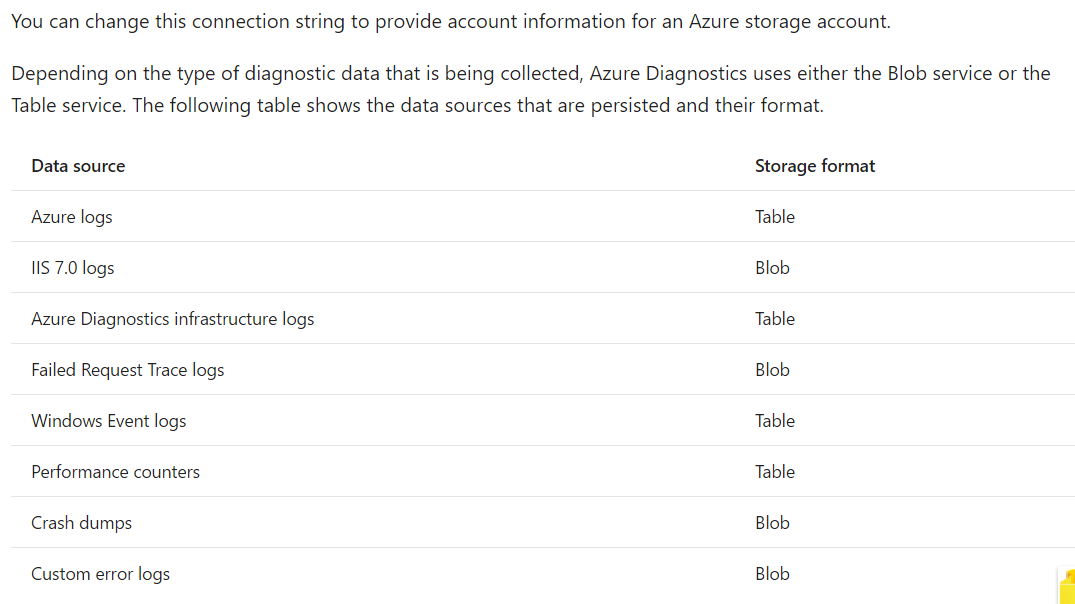


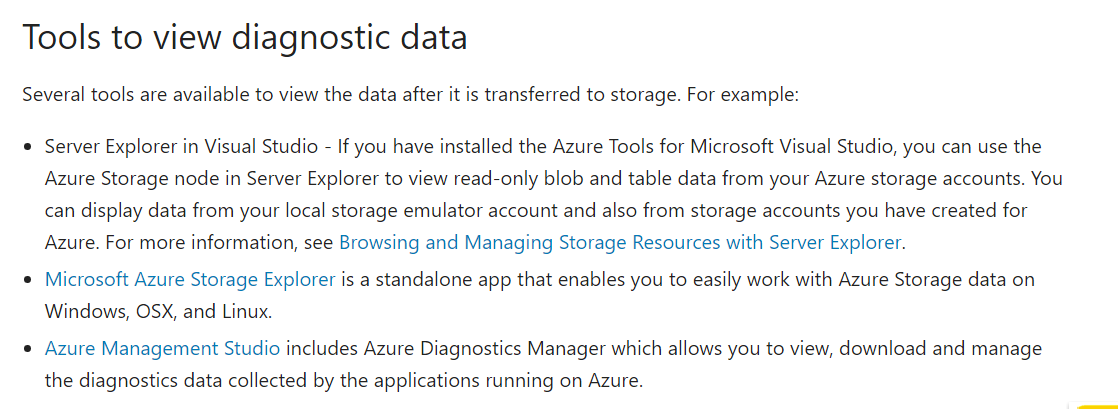




**Storage and Data Source – Diagnostic**

[**https://docs.microsoft.com/en-us/azure/cloud-services/cloud-services-dotnet-diagnostics-storage**](https://docs.microsoft.com/en-us/azure/cloud-services/cloud-services-dotnet-diagnostics-storage)





# Configure the Roles for an Azure Cloud Service with Visual Studio

11/11/2016 17 min to read Contributors

* [Tom Archer](https://github.com/TomArcher)

* [Ralph Squillace](https://github.com/squillace)

* [Andy De George](https://github.com/Thraka)

* [Kemp Brown](https://github.com/kempb)

An Azure cloud service can have one or more worker or web roles. For each role you need to define how that role is set up and also configure how that role runs. To learn more about roles in cloud services, see the video [Introduction to Azure Cloud Services](https://channel9.msdn.com/Series/Windows-Azure-Cloud-Services-Tutorials/Introduction-to-Windows-Azure-Cloud-Services). The information for your cloud service is stored in the following files:

* **ServiceDefinition.csdef**

The service definition file defines the runtime settings for your cloud service including what roles are required, endpoints, and virtual machine size. None of the data stored in this file can be changed when your role is running.

* **ServiceConfiguration.cscfg**

The service configuration file configures how many instances of a role are run and the values of the settings defined for a role. The data stored in this file can be changed while your role is running.

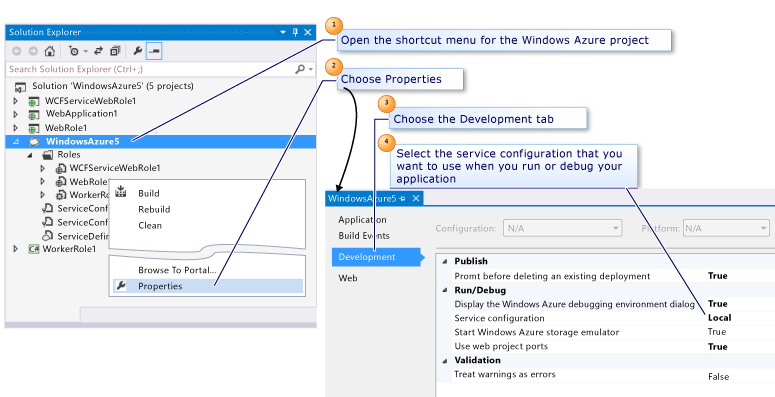
To be able to store different values for these settings for how your role runs, you can have multiple service configurations. You can use a different service configuration for each deployment environment. For example, you can set your storage account connection string to use the local Azure storage emulator in a local service configuration and create another service configuration to use the Azure storage in the cloud.

When you create a new Azure cloud service in Visual Studio, two service configurations are created by default. These configurations are added to your Azure project. The configurations are named:

* ServiceConfiguration.Cloud.cscfg
* ServiceConfiguration.Local.cscfg

## **Configure an Azure cloud service**

You can configure an Azure cloud service from Solution Explorer in Visual Studio, as shown in the following illustration.



### **To configure an Azure cloud service**

1. To configure each role in your Azure project from **Solution Explorer**, open the shortcut menu for the role in the Azure project and then choose **Properties**.

A page with the name of the role is displayed in the Visual Studio editor. The page displays the fields for the **Configuration** tab.

1. In the **Service Configuration** list, choose the name of the service configuration that you want to edit.

If you want to make changes to all of the service configurations for this role, you can choose **All Configurations**.

##### **Important**

If you choose a specific service configuration, some properties are disabled because they can only be set for all configurations. To edit these properties, you must choose All Configurations.

You can now choose a tab to update any enabled properties on that view.

## **Change the number of role instances**

To improve the performance of your cloud service, you can change the number of instances of a role that are running, based on the number of users or the load expected for a particular role. A separate virtual machine is created for each instance of a role when the cloud service runs in Azure. This will affect the billing for the deployment of this cloud service. For more information about billing, see [Understand your bill for Microsoft Azure](https://docs.microsoft.com/en-us/azure/billing/billing-understand-your-bill).

### **To change the number of instances for a role**

1. Choose the **Configuration** tab.
2. In the **Service Configuration** list, choose the service configuration that you want to update.

##### **Note**

You can set the instance count for a specific service configuration or for all service configurations.

1. In the **Instance count** text box, enter the number of instances that you want to start for this role.

##### **Note**

Each instance is run on a separate virtual machine when you publish your cloud service to Azure.

1. Choose the **Save** button on the toolbar to save these changes to the service configuration file.

## **Manage connection strings for storage accounts**

You can add, remove or modify connection strings for your service configurations. You might want different connection strings for different service configurations. For example, you might want a local connection string for a local service configuration that has a value of UseDevelopmentStorage=true. You might also want to configure a cloud service configuration that uses a storage account in Azure.

##### **Warning**

When you enter the Azure storage account key information for a storage account connection string, this information is stored locally in the service configuration file. However, this information is currently not stored as encrypted text.

By using a different value for each service configuration, you do not have to use different connection strings in your cloud service or modify your code when you publish your cloud service to Azure. You can use the same name for the connection string in your code and the value will be different, based on the service configuration that you select when you build your cloud service or when you publish it.

### **To manage connection strings for storage accounts**

1. Choose the **Settings** tab.
2. In the **Service Configuration** list, choose the service configuration that you want to update.

##### **Note**

You can update connection strings for a specific service configuration, but if you need to add or delete a connection string you must select All Configurations.

1. To add a connection string, choose the **Add Setting** button. A new entry is added to the list.
2. In the **Name** text box, type the name that you want to use for the connection string.
3. In the **Type** drop-down list, choose **Connection String**.
4. To change the value for the connection string, choose the ellipsis (...) button. The **Create Storage Connection String** dialog box appears.
5. To use the local storage account emulator, choose the **Microsoft Azure storage emulator** option button and then choose the **OK** button.
6. To use a storage account in Azure, choose the **Your subscription** option button and select the desired storage account.
7. To use custom credentials, choose the **Manually entered credentials** options button. Enter the storage account name, and either the primary or second key. For information about how to create a storage account and how to enter the details for the storage account in the **Create Storage Connection String** dialog box, see [Prepare to publish or deploy an Azure application from Visual Studio](https://docs.microsoft.com/en-us/azure/vs-azure-tools-cloud-service-publish-set-up-required-services-in-visual-studio).
8. To delete a connection string, select the connection string and then choose the **Remove Setting** button.
9. Choose the **Save** icon on the toolbar to save these changes to the service configuration file.
10. To access the connection string in the service configuration file, you must get the value of the configuration setting. The following code shows an example where blob storage is created and data uploaded using a connection string MyConnectionString from the service configuration file when a user chooses **Button1** on the Default.aspx page in the web role for an Azure cloud service. Add the following using statements to Default.aspx.cs:

Copy

using Microsoft.WindowsAzure;

using Microsoft.WindowsAzure.Storage;

using Microsoft.WindowsAzure.ServiceRuntime;

1. Open Default.aspx.cs in design view, and add a button from the toolbox. Add the following code to the Button1\_Click method. This code uses GetConfigurationSettingValue to get the value from the service configuration file for the connection string. Then a blob is created in the storage account that is referenced in the connection string MyConnectionString and finally the program adds text to the blob.

Copy

protected void Button1\_Click(object sender, EventArgs e)

{

// Setup the connection to Azure Storage

var storageAccount = CloudStorageAccount.Parse(RoleEnvironment.GetConfigurationSettingValue("MyConnectionString"));

var blobClient = storageAccount.CreateCloudBlobClient();

// Get and create the container

var blobContainer = blobClient.GetContainerReference("quicklap");

blobContainer.CreateIfNotExists();

// upload a text blob

var blob = blobContainer.GetBlockBlobReference(Guid.NewGuid().ToString());

blob.UploadText("Hello Azure");

}

## **Add custom settings to use in your Azure cloud service**

Custom settings in the service configuration file let you add a name and value for a string for a specific service configuration. You might choose to use this setting to configure a feature in your cloud service by reading the value of the setting and using this value to control the logic in your code. You can change these service configuration values without having to rebuild your service package or when your cloud service is running. Your code can check for notifications of when a setting changes. See [RoleEnvironment.Changing Event](https://msdn.microsoft.com/library/azure/microsoft.windowsazure.serviceruntime.roleenvironment.changing.aspx).

You can add, remove or modify custom settings for your service configurations. You might want different values for these strings for different service configurations.

By using a different value for each service configuration, you do not have to use different strings in your cloud service or modify your code when you publish your cloud service to Azure. You can use the same name for the string in your code and the value will be different, based on the service configuration that you select when you build your cloud service or when you publish it.

### **To add custom settings to use in your Azure cloud service**

1. Choose the **Settings** tab.
2. In the **Service Configuration** list, choose the service configuration that you want to update.

##### **Note**

You can update strings for a specific service configuration, but if you need to add or delete a string, you must select **All Configurations**.

1. To add a string, choose the **Add Setting** button. A new entry is added to the list.
2. In the **Name** text box, type the name that you want to use for the string.
3. In the **Type** drop-down list, choose **String**.
4. To add or change the value for the string, in the **Value** text box type the new value.
5. To delete a string, select the string and then choose the **Remove Setting** button.
6. Choose the **Save** button on the toolbar to save these changes to the service configuration file.
7. To access the string in the service configuration file, you must get the value of the configuration setting.

You need to make sure that the following using statements are already added to Default.aspx.cs just as you did in the previous procedure.

Copy

using Microsoft.WindowsAzure;

using Microsoft.WindowsAzure.Storage;

using Microsoft.WindowsAzure.ServiceRuntime;

1. Add the following code to the Button1\_Click method to access this string in the same way that you access a connection string. Your code can then perform some specific code based on the value of the settings string for the service configuration file that is used.

Copy

var settingValue = RoleEnvironment.GetConfigurationSettingValue("MySetting");

if (settingValue == “ThisValue”)

{

// Perform these lines of code

}

## **Manage local storage for each role instance**

You can add local file system storage for each instance of a role. You can store local data here that does not need to be accessed by other roles. Any data that you do not need to save into table, blob, or SQL Database storage can be stored in here. For example, you could use this local storage to cache data that might need to be used again. This stored data can’t be accessed by other instances of a role.

Local storage settings apply to all service configurations. You can only add, remove, or modify local storage for all service configurations.

### **To manage local storage for each role instance**

1. Choose the **Local Storage** tab.
2. In the **Service Configuration** list, choose **All Configurations**.
3. To add a local storage entry, choose the **Add Local Storage** button. A new entry is added to the list.
4. In the **Name** text box, type the name that you want to use for this local storage.
5. In the **Size** text box, type the size in MB that you need for this local storage.
6. To remove the data in this local storage when the virtual machine for this role is recycled, select the **Clean on role recycle** check box.
7. To edit an existing local storage entry, choose the row that you need to update. Then you can edit the fields, as described in the previous steps.
8. To delete a local storage entry, choose the storage entry in the list and then choose the **Remove Local Storage** button.
9. To save these changes to the service configuration files, choose the **Save** icon on the toolbar.
10. To access the local storage that you have added in the service configuration file, you must get the value of the local resource configuration setting. Use the following lines of code to access this value and create a file called **MyStorageTest.txt** and write a line of test data into that file. You can add this code into the Button\_Click method that you used in the previous procedures:
11. You need to make sure that the following using statements are added to Default.aspx.cs:

Copy

using System.IO;

using System.Text;

1. Add the following code to the Button1\_Click method. This creates the file in the local storage and writes test data into that file.

Copy

// Retrieve an object that points to the local storage resource

LocalResource localResource = RoleEnvironment.GetLocalResource("LocalStorage1");

//Define the file name and path

string[] paths = { localResource.RootPath, "MyStorageTest.txt" };

String filePath = Path.Combine(paths);

using (FileStream writeStream = File.Create(filePath))

{

Byte[] textToWrite = new UTF8Encoding(true).GetBytes("Testing Web role storage");

writeStream.Write(textToWrite, 0, textToWrite.Length);

}

1. (Optional) To view this file that you created when you run your cloud service locally, use the following steps:
   1. Run the web role and select **Button1** to make sure that the code inside Button1\_Click gets called.
   2. In the notification area, open the shortcut menu for the Azure icon and choose **Show Compute Emulator UI**. The **Azure Compute Emulator** dialog box appears.
   3. Select the web role.
   4. On the menu bar, choose **Tools**, **Open local store**. A Windows Explorer window appears.
   5. On the menu bar, enter **MyStorageTest.txt** into the **Search** text box and then choose **Enter** to start the search.

The file is displayed in the search results.

* 1. To view the contents of the file, open the shortcut menu for the file and choose **Open**.

## **Collect cloud service diagnostics**

You can collect diagnostics data for your Azure cloud service. This data is added to a storage account. You might want different connection strings for different service configurations. For example, you might want a local storage account for a local service configuration that has a value of UseDevelopmentStorage=true. You might also want to configure a cloud service configuration that uses a storage account in Azure. For more information about Azure diagnostics, see Collect Logging Data by Using Azure Diagnostics.

##### **Note**

The local service configuration is already configured to use local resources. If you use the cloud service configuration to publish your Azure cloud service, the connection string that you specify when you publish is also used for the diagnostics connection string unless you have specified a connection string. If you package your cloud service using Visual Studio, the connection string in the service configuration is not changed.

### **To collect cloud service diagnostics**

1. Choose the **Configuration** tab.
2. In the **Service Configuration** list, choose the service configuration that you want to update or choose **All Configurations**.

##### **Note**

You can update the storage account for a specific service configuration, but if you want to enable or disable diagnostics you must choose All Configurations.

1. To enable diagnostics, select the **Enable Diagnostics** check box.
2. To change the value for the storage account, choose the ellipsis (...) button.

The **Create Storage Connection String** dialog box appears.

1. To use a local connection string, choose Azure storage emulator option and then choose the **OK** button.
2. To use a storage account associated with your Azure subscription, choose the **Your subscription** option.
3. To use a storage account for the local connection string, choose the **Manually entered credentials** option.

For more information about how to create a storage account and how to enter the details for the storage account in the **Create Storage Connection String** dialog box, see [Prepare to publish or deploy an Azure application from Visual Studio](https://docs.microsoft.com/en-us/azure/vs-azure-tools-cloud-service-publish-set-up-required-services-in-visual-studio).

1. Choose the storage account you want to use in **Account name**.

If you are manually entering your storage account credentials, copy or type your primary key in **Account key**. This key can be copied from the [Azure classic portal](http://go.microsoft.com/fwlink/?LinkID=213885). To copy this key, following these steps from the **Storage Accounts** view in the [Azure classic portal](http://go.microsoft.com/fwlink/?LinkID=213885):

* 1. Select the storage account that you want to use for your cloud service.
  2. Choose the **Manage Access Keys** button located at the bottom of the screen. The **Manage Access Keys** dialog box appears.
  3. To copy the access key, choose the **Copy to clipboard** button. You can now paste this key into the **Account key** field.

1. To use the storage account that you provide, as the connection string for diagnostics (and caching) when you publish your cloud service to Azure, select the **Update development storage connection strings for Diagnostics and Caching with Azure storage account credentials when publishing to Azure** check box.
2. Choose the **Save** button on the toolbar to save these changes to the service configuration file.

## **Change the size of the virtual machine used for each role**

You can set the virtual machine size for each role. You can only set this size for all service configurations. If you select a smaller machine size, then less CPU cores, memory and local disk storage is allocated. The allocated bandwidth is also smaller. For more information about these sizes and the resources allocated, see [Sizes for Cloud Services](https://docs.microsoft.com/en-us/azure/cloud-services/cloud-services-sizes-specs).

The resources required for each virtual machine in Azure affects the cost of running your cloud service in Azure. For more information about Azure Billing, see [Understand your bill for Microsoft Azure](https://docs.microsoft.com/en-us/azure/billing/billing-understand-your-bill).

### **To change the size of the virtual machine**

1. Choose the **Configuration** tab.
2. In the **Service Configuration** list, choose **All Configurations**.
3. To select the size for the virtual machine for this role, choose the appropriate size from the **VM size** list.
4. Choose the **Save** button on the toolbar to save these changes to the service configuration file.

## **Manage endpoints and certificates for your roles**

You configure networking endpoints by specifying the protocol, the port number, and, for HTTPS, the SSL certificate information. Releases before June 2012 support HTTP, HTTPS, and TCP. The June 2012 release supports those protocols and UDP. You can’t use UDP for input endpoints in the compute emulator. You can use that protocol only for internal endpoints.

To improve the security of your Azure cloud service, you can create endpoints that use the HTTPS protocol. For example, if you have a cloud service that is used by customers to purchase orders, you want to make sure that their information is secure by using SSL.

You can also add endpoints that can be used internally or externally. External endpoints are called input endpoints. An input endpoint allows another access point to users to your cloud service. If you have a WCF service, you might want to expose an internal endpoint for a web role to use to access this service.

##### **Important**

You can only update endpoints for all service configurations.

If you add HTTPS endpoints, you need to use an SSL certificate. To do this you can associate certificates with your role for all service configurations and use these for your endpoints.

##### **Important**

These certificates are not packaged with your service. You must upload your certificates separately to Azure through the [Azure classic portal](http://go.microsoft.com/fwlink/?LinkID=213885).

Any management certificates that you associate with your service configurations apply only when your cloud service runs in Azure. When your cloud service runs in the local development environment, a standard certificate that is managed by the Azure compute emulator is used.

### **To add a certificate to a role**

1. Choose the **Certificates** tab.
2. In the **Service Configuration** list, choose **All Configurations**.

##### **Note**

To add or remove certificates, you must select All Configurations. You can update the name and the thumbprint for a specific service configuration if it is required.

1. To add a certificate for this role, choose the **Add Certificate** button. A new entry is added to the list.
2. In the **Name** text box, enter the name for the certificate.
3. In the **Store Location** list, choose the location for the certificate that you want to add.
4. In the **Store Name** list, choose the store that you want to use to select the certificate.
5. To add the certificate, choose the ellipsis (...) button. The **Windows Security** dialog box appears.
6. Choose the certificate that you want to use from the list and then choose the **OK** button.

##### **Note**

When you add a certificate from the certificate store, any intermediate certificates are added automatically to the configuration settings for you. These intermediate certificates must also be uploaded to Azure in order to correctly configure your service for SSL.

1. To delete a certificate, choose the certificate and then choose the **Remove Certificate** button.
2. Choose the **Save** icon in the toolbar to save these changes to the service configuration files.

### **To manage endpoints for a role**

1. Choose the **Endpoints** tab.
2. In the **Service Configuration** list, choose **All Configurations**.
3. To add an endpoint, choose the **Add Endpoint** button. A new entry is added to the list.
4. In the **Name** text box, type the name that you want to use for this endpoint.
5. Choose the type of endpoint that you need from the **Type** list.
6. Choose the protocol for the endpoint that you need from the **Protocol** list.
7. If it is an input endpoint, in the **Public Port** text box, enter the public port to use.
8. In the **Private Port** text box type the private port to use.
9. If the endpoint requires the https protocol, in the **SSL Certificate Name** list choose a certificate to use.

##### **Note**

This list shows the certificates that you have added for this role in the **Certificates** tab.

1. Choose the **Save** button on the toolbar to save these changes to the service configuration files.

## **Next steps**

Learn more about Azure projects in Visual Studio by reading [Configuring an Azure Project](https://docs.microsoft.com/en-us/azure/vs-azure-tools-configuring-an-azure-project). Learn more about the cloud service schema by reading [Schema Reference](https://msdn.microsoft.com/library/azure/dd179398).