Install a SQL\IIS\.NET stack in Azure

, we install a SQL\IIS\.NET stack using Azure PowerShell. This stack consists of two VMs running Windows Server 2016, one with IIS and .NET and the other with SQL Server.

* Create a VM using New-AzVM
* Install IIS and the .NET Core SDK on the VM
* Create a VM running SQL Server
* Install the SQL Server extension

Create a IIS VM

In this example, we use the [New-AzVM](https://www.powershellgallery.com/packages/AzureRM.Compute.Experiments) cmdlet in the PowerShell Cloud Shell to quickly create a Windows Server 2016 VM and then install IIS and the .NET Framework. The IIS and SQL VMs share a resource group and virtual network, so we create variables for those names.

Click on the **Try It** button to the upper right of the code block to launch Cloud Shell in this window. You will be asked to provide credentials for the virtual machine at the cmd prompt.

Azure PowerShellCopyTry It

$vNetName = "myIISSQLvNet"

$resourceGroup = "myIISSQLGroup"

New-AzVm -Name myIISVM -ResourceGroupName $resourceGroup -VirtualNetworkName $vNetName

Install IIS and the .NET framework using the custom script extension.

Azure PowerShellCopyTry It

Set-AzureRmVMExtension -ResourceGroupName $resourceGroup `

-ExtensionName IIS `

-VMName myIISVM `

-Publisher Microsoft.Compute `

-ExtensionType CustomScriptExtension `

-TypeHandlerVersion 1.4 `

-SettingString '{"commandToExecute":"powershell Add-WindowsFeature Web-Server,Web-Asp-Net45,NET-Framework-Features"}' `

-Location EastUS

Azure SQL VM

We use a pre-configured Azure marketplace image of a SQL server to create the SQL VM. We first create the VM, then we install the SQL Server Extension on the VM.

Azure PowerShellCopyTry It

# Create user object. You get a pop-up prompting you to enter the credentials for the VM.

$cred = Get-Credential -Message "Enter a username and password for the virtual machine."

# Create a subnet configuration

$vNet = Get-AzureRmVirtualNetwork -Name $vNetName -ResourceGroupName $resourceGroup

Add-AzureRmVirtualNetworkSubnetConfig -Name mySQLSubnet -VirtualNetwork $vNet -AddressPrefix "192.168.2.0/24"

Set-AzureRmVirtualNetwork -VirtualNetwork $vNet

# Create a public IP address and specify a DNS name

$pip = New-AzureRmPublicIpAddress -ResourceGroupName $resourceGroup -Location eastus `

-Name "mypublicdns$(Get-Random)" -AllocationMethod Static -IdleTimeoutInMinutes 4

# Create an inbound network security group rule for port 3389

$nsgRuleRDP = New-AzureRmNetworkSecurityRuleConfig -Name myNetworkSecurityGroupRuleRDP -Protocol Tcp `

-Direction Inbound -Priority 1000 -SourceAddressPrefix \* -SourcePortRange \* -DestinationAddressPrefix \* `

-DestinationPortRange 3389 -Access Allow

# Create a network security group

$nsg = New-AzureRmNetworkSecurityGroup -ResourceGroupName $resourceGroup -Location eastus `

-Name myNetworkSecurityGroup -SecurityRules $nsgRuleRDP

# Create a virtual network card and associate with public IP address and NSG

$nic = New-AzureRmNetworkInterface -Name mySQLNic -ResourceGroupName $resourceGroup -Location eastus `

-SubnetId $vnet.Subnets[0].Id -PublicIpAddressId $pip.Id -NetworkSecurityGroupId $nsg.Id

# Create a virtual machine configuration

$vmConfig = New-AzureRmVMConfig -VMName mySQLVM -VMSize Standard\_D1 | `

Set-AzureRmVMOperatingSystem -Windows -ComputerName mySQLVM -Credential $cred | `

Set-AzureRmVMSourceImage -PublisherName MicrosoftSQLServer -Offer SQL2014SP2-WS2012R2 -Skus Enterprise -Version latest | `

Add-AzureRmVMNetworkInterface -Id $nic.Id

# Create the VM

New-AzureRmVM -ResourceGroupName $resourceGroup -Location eastus -VM $vmConfig

Use [Set-AzureRmVMSqlServerExtension](https://docs.microsoft.com/en-us/powershell/module/azurerm.compute/set-azurermvmsqlserverextension) to add the [SQL Server extension](https://docs.microsoft.com/en-us/sql/virtual-machines-windows-sql-server-agent-extension.md) to the SQL VM.

Azure PowerShellCopyTry It

Set-AzureRmVMSqlServerExtension -ResourceGroupName $resourceGroup -VMName mySQLVM -name "SQLExtension"