# Practice M1: Core Infrastructure Services

For this module we will need a PC running recent version of **Windows**, **Linux**, or **macOS**. It can be either physical or virtual machine

All resource related tasks can be executed either in the locally installed tools or in the **Azure Cloud Shell**

## Part 1: Azure Introduction

If you do not have a subscription yet, you must create one

In general, you have two options:

* Use a student subscription – if you are studying at an education instruction, you may apply for a student subscription
* Use a free trial subscription – visit this URL (<https://azure.microsoft.com/en-us/free/>) and follow the instructions. This step includes entering personal information including full name, phone number, credit/debit card information, etc.

The better option from the above is the trial subscription due to two reasons. First, it covers all services that will be covered during the course. And second, it can be converted to Pay-As-You-Go subscription. The latter applies to the student subscription as well

Alternatively, you may use an existing subscription either your own, or one provided by your employer

### Explore the Azure Portal

Navigate to <https://portal.azure.com>

Enter your credentials

You will be brought to the **Home** screen

On the top there is a toolbar with several sections:

* On the most left side there is a button that shows/hides the menu. There, from the top to bottom, we can see:
  + **Create a resource** command - Quickly create a resource. Or just view the wide variety of services in the Azure Marketplace
  + **Home** command - View key info quickly - top **Azure** services, common tasks, recently used resources, online training, the **Azure** blog, and more
  + **Dashboard** command - View your customizable dashboard that displays exactly what you want: resources, services, alerts, tutorials, and more
  + **All Services** command - Browse the variety of available **Azure** services
  + **Favorites** section - Browse all your favorite services. To modify your favorites, select **All services** and toggle the star icons
* In the middle there is the **search** box. It allows us to quickly find resources we are looking for
* Next, there are few buttons:
  + **Cloud Shell** - Activates the integrated shell interface
  + **Directory + Subscription** - Switch between directories or use global subscription filter to filter resources in portal
  + **Notifications** - View status updates and other information
  + **Settings** - Adjust portal settings
  + **Help + support** - Get help or open a support request
  + **Feedback** - Functionality to send feedback
* Lastly, on the far right, there is the **account information**. It allows us to switch directories, change our password, view our permissions, submit an idea, or view our bill

Go to **settings** and change the behavior of the menu. Change the default view. Try another theme

Let’s go to the **Marketplace** and explore different categories. Try the search and filter capabilities

Explore the **dashboard** mode

Go to **all services** and explore them. Add a few more to the **favorites** section, or remove some

## Part 2: Core Services

Login to the Azure portal if you are not

### Create a Linux (Ubuntu-based) VM

Let’s create an **Ubuntu 18.04** machine with all settings by default

Go to the **home** view if you are not there

In the **search** box start typing **virtual machines**. When you see the option **Virtual machines**, click on it

Click on the **+ Add** button and select **Virtual machine** in the drop-down list

Select your **subscription** if not selected

In the **resource group** section click on **create new**

Enter a name, for example **RG-Demo-1** and hit **OK**

For **virtual machine name** enter **VM-Demo-Ubuntu-1.** It must be unique for the resource group

In the **region** drop-down select something closer to your location, for example **(Europe) West Europe**

In the **image** drop-down select **Ubuntu Server 18.04 LTS - Gen 1**

Click on the **select size** option

Explore the available VM sizes. Select something smaller, for example **B1s**. Then click on the **select** button

For **authentication type** select the **password** option

In the **username** field enter for example **demouser**

**Password** must comply with the following:

* Length between 12 and 72 characters
* Must have three of the following: 1 lower case, 1 upper case, 1 number, and 1 special character
* Must not include reserved words or unsupported characters

Enter for example **DemoPassword-2021**

Now, we have entered the minimal amount of information needed to create a VM

Let’s hit the **review + create** button to examine what we are about to create

Click on the **create** button to initiate the creation process

Now, we can sit and watch the **deployment** process. It will finish in 30 or so seconds

From the main menu choose the **Home** option to go back to the home screen

Click on **Resource groups** option in the **Azure services** section

In the list of groups click on the **RG-Demo-1** group

Now, we can see the full list of resources created together with our virtual machine

#### Connect to the VM

Being in the resource group overview, find the virtual machine **VM-Demo-Ubuntu-1** and click on it

Now, we are in **Overview** mode. Here, we can see that it is running. We can retrieve information about its size, public and private IP addresses, etc.

Most of the information is in the **Properties** tab

If we switch to the **Monitoring** tab, we will see that there are also a few charts that show us how the machine is performing

From this screen, we can stop, restart, and delete the machine as well. We will not take such actions now

Click on the **Connect** command in the top-left part of the **Overview** screen and select the **SSH** option from the list

Examine the provided instructions. As you can see, they suppose that we are using a key pair to connect to the VM

In the current situation, we will use username and password pair instead

Depending on your host operating system and the available software, you must either start an SSH client (like **PuTTY**) or open a terminal window (if running on **Linux**, **macOS**, or **Windows 10** with **OpenSSH** client installed)

If you are going to use a dedicated application, then you must use the username (**demouser**) and the public IP address separately

Examples that follow, will presume that we are working in a shell session. This won’t affect the way you interact with the VM, the only difference is how you connect to the VM

To connect to the VM, execute this:

**ssh demouser@<public-ip-address>**

If this is the first time, we connect to the VM, we will be asked if we trust the host we are about to connect. Answer with **yes**

Then enter the password (**DemoPassword-2021**) and voila, we are in

Here we can execute standard Linux commands like **ps**, **top**, **ls**, **free**, **uname**, etc. Let’s experiment with a few

#### Install a web server

Because this is a standard Ubuntu installation, we can add software as if we are working on a “regular” VM

Let’s refresh the repositories information, upgrade installed packages, and then install the **nginx** web server:

**sudo apt-get update**

**…**

**sudo apt-get upgrade**

**…**

**sudo apt-get install -y nginx**

**…**

We used the **sudo** command in front of every command above, because we need higher privileges to modify the system configuration

Now, let’s check if our web server is running:

**systemctl status nginx**

If it is not running, we could start and enable (auto start on reboot) it with (skip it if the above showed **running**):

**sudo systemctl start nginx**

**sudo systemctl enable nginx**

Knowing the public IP address from earlier steps, let’s open a browser tab and paste the address there

After a while, an error will be returned that the page is not accessible. The reason for this is that our deployments are secured by default and if we do not explicitly specify that we want certain communication from the outside (inbound communication) to happen, it won’t

#### Adjust the security

Return to the **resource group** in the **Azure Portal**

Find the **VM-Demo-Ubuntu-1-nsg** security group created during the virtual machine deployment and click on it

In the **Overview** mode we can see that there are two sections – **Inbound security rules** and **Outbound security rules**. They are like a summary (overview) of how the security group is configured

Click on the **Inbound security rules** under the **Settings** section

Click on the **Default rules** button to hide the default security rules. Only the one for **SSH** remains. It was created during the machine deployment

Now click on the **+ Add** button to add one more rule

Change the **Destination port ranges** value to **80**

Set the **Protocol** selector to **TCP**

In the **Name** field enter for example **HTTP-Port-80**

Click on the **Add** button to confirm

Now, switch back to the browser tab with the error and hit **refresh**

You should see the default **NGINX** web page

#### Add additional disk

Return to the shell session or re-establish a new one if you closed it

Let’s check what disk devices are available and how they are utilized:

**lsblk**

We can see that there are two disk drives – **sda** (operating system is installed there) and **sdb** (temp disk)

Let’s return to the portal

Go to the **VM-Demo-Ubuntu-1** virtual machine

Click on **Disks** command in the **Settings** section

Click on **+ Create and attach a new disk** button under **Data disks** section

For **Disk name** enter **VM-Demo-Ubuntu-1\_disk2**

You can change **Storage type** and **Size**

Select for example **Standard SSD** for storage type and **16 GiB** for size

Confirm with the **Save** button (top left)

After a while saving process will finish

Return to the resource group and hit **Refresh** button to see the new resource

Return to the console session or if you closed it, establish a new one

Check again what disks are available with:

**lsblk**

**…**

**sdc 8:32 0 16G 0 disk**

**…**

You should see now that there is a new device – **sdc**

Now, let’s issue set of commands to make the disk usable. Start the disk partitioning program:

**sudo fdisk /dev/sdc**

**…**

**Command (m for help): n**

**Partition type**

**p primary (0 primary, 0 extended, 4 free)**

**e extended (container for logical partitions)**

**Select (default p): p**

**Partition number (1-4, default 1): 1**

**First sector (2048-33554431, default 2048):**

**Last sector, +sectors or +size{K,M,G,T,P} (2048-33554431, default 33554431):**

**Created a new partition 1 of type 'Linux' and of size 16 GiB.**

**Command (m for help): w**

**…**

When in the program, hit the following sequence of keys:

**n** – to create a new partition

**p** – to create a primary partition

**1** – to set partition number to 1

**Enter** key – to confirm the first sector

**Enter** key again – to confirm the last sector

**w** – to write the changes and exit

Now, we can check again the situation with the disks:

**lsblk**

**…**

**sdc 8:32 0 16G 0 disk**

**└─sdc1 8:33 0 16G 0 part**

**...**

We can see that we have a partition

Let’s create a file system there:

**sudo mkfs.ext4 /dev/sdc1**

Our new disk is ready to be mounted and used. First, we will create a folder, and then we will mount it:

**sudo mkdir /disk**

**sudo mount /dev/sdc1 /disk**

If we ask again for information about the block devices, we will see that our disk (partition) is mounted:

**lsblk**

**…**

**sdc 8:32 0 16G 0 disk**

**└─sdc1 8:33 0 16G 0 part /disk**

**…**

Should we want our disk to be auto mounted after a reboot, we must change the **/etc/fstab** file. Be very careful with this file, as if gets corrupted the system won’t boot

To modify the **/etc/fstab** file we can invoke the **nano** text editor:

**sudo nano /etc/fstab**

Use the arrow keys to go to last (empty) line

Enter the following text:

**/dev/sdc1 /disk ext4 defaults 0 0**

Hit **Ctr+O** and then the **Enter** key to confirm

Exit with **Ctrl+X**

A valid way to test if everything with the **/etc/fstab** file is okay is to issue the following command:

**sudo mount -a**

It will try to mount all filesystems that exist in the **/etc/fstab** file

If you do not see any error, then you are good to go

### Create a Windows VM

This time we will change the sequence a little bit. We will create the **resource group** first, and then the VM

Go to the **Home** view

Select **Resource groups** from the **Azure services** section

Hit the **+ Add** button

Select the appropriate **Subscription** if you have more than one

For **Resource group** name enter **RG-Demo-2**

In the **Region** drop-down select **(Europe) West Europe**

Click on the **Review + create** button

If all looks good, proceed by clicking the **Create** button

Now, that we have the resource group, we can continue with the machine creation

In the **Search** box (top of the screen) type **Marketplace** and hit the **Enter** key

Then click on **Compute** item in **Categories**

Find **Windows Server** and click on it

In the **Select a plan** drop-down select **[smalldisk] Windows Server 2012 R2 Datacenter**

Click on **Create**

Select the appropriate **Subscription**

In the **Resource group** drop-down select **RG-Demo-2**

For **Virtual machine name** enter **VM-Windows-2012**

Click on **Select size**

Select the **B1s** size (**B1ms** or **B2** will be better, but will incur more expenses) and click on the **Select** button

In the **Username** field enter **DemoUser**

For a password use for example **DemoPassword-2021**

Click on the **Review + create** button

Click on the **Create** button

Sit back and watch the deployment process. After a while, between one and two minutes, the machine will be ready

From the main menu choose the **Home** option to go back to the home screen

Click on **Resource groups** option in the **Azure services** section

In the list of groups click on the **RG-Demo-2** group

Now, we can see the full list of resources created together with our virtual machine

#### Connect to the VM

Being in the resource group **Overview** mode, find the virtual machine **VM-Windows-2012** and click on it

Now, we can see that it is running. We can retrieve information about its size, public and private IP addresses, etc.

There are also a few charts (under the **Monitoring** tab) that show us how the machine is performing

From this screen, we can stop, restart, and delete the machine as well. We will not take such actions now

Click on the **Connect** command in the top-left part of the **Overview** screen and select **RDP** from the list

Click on the **Download RDP File** button to download the connection information and initiate a connection

When asked, select the option to open the file and confirm with **OK**

A warning will appear saying that there is issue with the certificate of the remote machine. Confirm with **Connect**

Enter the credentials used during the creation process and confirm with **OK**

Another certificate related warning will appear, confirm with **Yes**

Now, we should see the desktop of our virtual machine

#### Install a web server

Let’s install the **IIS** web server

Start a **PowerShell** session with the **Run as administration** option

Execute the following command (**-Restart** at the end can be skipped in this situation):

**Add-WindowsFeature -Name Web-Server -IncludeManagementTools -Restart**

Of course, there are other ways to achieve the same. For example, using the **Server Manager** and **Add Roles and Features** option

After a while, the feature will be ready

Open the **Internet Explorer** browser and navigate to <http://localhost>

The default site should appear

If you return to your host machine and try to reach the site from the outside, an error will be returned

#### Adjust the security

Return to the **resource group** in the **Azure Portal**

Find the **VM-Windows-2012-nsg** security group created during the virtual machine deployment and click on it

In the **Overview** mode we can see that there are two sections – **Inbound security rules** and **Outbound security rules**. They are like a summary (overview) of how the security group is configured

Click on the **Inbound security rules** under the **Settings** section

Click on the **Default rules** button to hide the default security rules. Only the one for **RDP** remains. It was created during the machine deployment

Now click on the **+ Add** button to add one more rule

Change the **Destination port ranges** value to **80**

Set the **Protocol** selector to **TCP**

In the **Name** field enter for example **HTTP-Port-80**

Click on the **Add** button to confirm

Now, switch back to the browser tab with the error and hit **refresh**

You should see the default **IIS** web page

#### Add additional disk

Return to the RDP session or re-establish a new one if you closed it

Let’s check in **Disk Management** what disk devices are available and how they are utilized

We can see that there are two disk drives – **Disk 0** (C: - operating system is installed there) and **Disk 1** (D: - temp disk)

Let’s return to the portal

Go to the **VM-Windows-2012** virtual machine

Click on **Disks** command in the **Settings** section

Click on **+ Create and attach a new disk** button under **Data disks** section

For **Disk name** enter **VM-Windows-2012\_DataDisk**

You can change **Storage type** and **Size**

Select for example **Standard SSD** for storage type and **16 GiB** for size

Confirm with the **Save** button (top left)

After a while saving process will finish. Return to the resource group and hit **Refresh** button to see the new resource

Return to the RDP session or if you closed it, establish a new one

Start again the **Disk Management** tool if not started

There you will see the new disk. It will appear as **Disk 2.** Click on it and choose **Initialize Disk** from the context menu

Confirm with the **OK** button

Start the **New Simple Volume Wizard** and follow the instructions. After a while a new **NTFS** drive with the letter **F:** will appear

## Part 3: Azure the Console Way

### Install Azure CLI

Azure CLI is available for most major operating systems including Windows, macOS, and different Linux distributions

#### Install Azure CLI on Windows

You can download the latest version of **Azure CLI** for **Windows** from here: <https://aka.ms/installazurecliwindows>

Once, the installer file is downloaded, you can install it like any other **Windows** application

Alternatively, you can execute the following to install it from a **PowerShell** session with elevated permissions:

**Invoke-WebRequest -Uri https://aka.ms/installazurecliwindows -OutFile .\AzureCLI.msi**

**Start-Process msiexec.exe -Wait -ArgumentList '/I AzureCLI.msi /quiet'**

Once the installation is finished, open a shell session, either in CMD or PowerShell, and test if everything is okay:

**az --version**

To see available commands, you can execute the binary without any parameters:

**az**

#### Install Azure CLI on Linux

Installation of **Azure CLI** varies among distributions, but in general, the preferred approach is to add a special **Microsoft** repository and use the package manager of the distribution

For example, on **openSUSE** you can execute the following set of commands. First, add the prerequisites:

**sudo zypper install -y curl**

Then import the Microsoft repository key:

**sudo rpm --import https://packages.microsoft.com/keys/microsoft.asc**

Register the **Microsoft’s** repository:

**sudo zypper addrepo --name 'Azure CLI' --check https://packages.microsoft.com/yumrepos/azure-cli azure-cli**

Install the **Azure CLI** package:

**sudo zypper install --from azure-cli -y azure-cli**

Same approach for **RedHat**/**Fedora**/**CentOS**: <https://docs.microsoft.com/en-us/cli/azure/install-azure-cli-yum>

Same applies to **Debian**-based distributions: <https://docs.microsoft.com/en-us/cli/azure/install-azure-cli-apt>

#### Install Azure CLI on macOS

The easiest way to install **Azure CLI** under **macOS** is to use **Homebrew**

It is enough to execute the following:

**brew update && brew install azure-cli**

### Create а Simple Solution with Azure CLI

Before we can do anything with our **Azure** subscription from **Azure CLI**, we must login first:

**az login**

Then, we must enter our credentials in the window that appeared

Let’s create a simple but complete set of resources from the **Azure CLI**

We will create a resource group and a Linux-based VM, then we will install a web server and open a port

First, we need to get familiar with some of the commands provided by the **Azure CLI**. To ask for help for the **group** command, we must execute:

**az group --help**

Then if we need help for a specific action like **create**, we can execute:

**az group create --help**

Now, we are ready to proceed further. To create the resource group, we must execute:

**az group create --name RG-Demo-3 --location westeurope**

Then, we can create the VM with command like this (do not execute it yet):

**az vm create --name VM-Ubuntu --resource-group RG-Demo-3 --image UbuntuLTS --size Standard\_B1s**

If we decide to use the short form of the parameters, the above will become:

**az vm create -n VM-Ubuntu -g RG-Demo-3 --image UbuntuLTS --size Standard\_B1s**

As you can see, not every parameter has a short variant. Which form we will use is a matter of personal preference

Let’s extend the command with username and password. The final command will look like:

**az vm create --name VM-Ubuntu --resource-group RG-Demo-3 --image UbuntuLTS --size Standard\_B1s --admin-username demouser --admin-password DemoPassword-2021**

After a successful execution, which will take a while, we can ask for all resources in the resource group:

**az resource list --resource-group RG-Demo-3**

The output by default is formatted as **JSON**, but we can ask for different format:

**az resource list --resource-group RG-Demo-3 --output table**

Furthermore, we can ask for particular fields:

**az resource list --resource-group RG-Demo-3 --query [].[name,type] --output table**

Or with some additional formatting:

**az resource list --resource-group RG-Demo-3 --query "[].{Name:name,Type:type}" --output table**

Get the VM’s public IP address with:

**az vm show --name VM-Ubuntu --resource-group RG-Demo-3 --show-details --query [publicIps] --output tsv**

Now, that we know the public IP address, we can reach the machine the usual way. Then we can install a web server. Both steps are the same as with the ones done in **Part 2**

Again, the site is not reachable from the outside, so we must open a port. This can be done with:

**az vm open-port --name VM-Ubuntu --resource-group RG-Demo-3 --port 80**

The site must be reachable now

#### Add Additional Disk via the Azure CLI

We can replicate all steps, that we did in the **Azure Portal** earlier to add a data disk, but this time with one single command:

**az vm disk attach --vm-name VM-Ubuntu --resource-group RG-Demo-3 --name VM-Ubuntu\_Disk2 --size-gb 16 --sku Standard\_LRS --new**

The rest of the steps to initialize the disk, create a filesystem and mount it permanently are the same

#### Clean up with Azure CLI

We can delete resources one-by-one, but it would be easier if we delete the whole resource group. After all, this is one of the reasons we use resource groups – easier management, based on their life cycle

Let’s delete it (you can skip this one, as at the end there will be a separate instruction on cleaning):

**az group delete --name RG-Demo-3 --yes**

If we don’t want to wait while the group and the resources are being deleted, we can modify the command to:

**az group delete --name RG-Demo-3 --yes --no-wait**

### Install Azure PowerShell Module

Before we continue, we must make sure that we are running the latest version of PowerShell (Core)

#### Install PowerShell Core on Windows

Download the latest MSI package from <https://github.com/PowerShell/PowerShell> and follow the wizard

#### Install PowerShell Core on Linux

Some distributions offer it via their repository systems. You can check here: <https://github.com/PowerShell/PowerShell>

There is a universal way - by using binary archive. It includes the following steps:

Download the archive from **GitHub**:

**curl -L https://github.com/PowerShell/PowerShell/releases/download/v7.0.3/powershell-7.0.3-linux-x64.tar.gz -o /tmp/powershell.tar.gz**

Create the target installation folder:

**sudo mkdir -p /opt/microsoft/powershell/7**

Extract the archive:

**sudo tar zxf /tmp/powershell.tar.gz -C /opt/microsoft/powershell/7**

Set appropriate permissions:

**sudo chmod +x /opt/microsoft/powershell/7/pwsh**

Create a symbolic link:

**sudo ln -s /opt/microsoft/powershell/7/pwsh /usr/bin/pwsh**

Start it with:

**pwsh**

You may need to install additional dependency packages. Check here:

<https://docs.microsoft.com/bg-bg/powershell/scripting/install/installing-powershell-core-on-linux?view=powershell-7#dependencies>

#### Install PowerShell Core on macOS

Installation process on **macOS 10.12+** is very simple and consists of just one step:

**brew cask install powershell**

Start it with:

**pwsh**

#### Install Azure PowerShell Module

Open an elevated (**Run as Administrator** under **Windows** or with **sudo** under **Linux** and **macOS**) **PowerShell** session

Execute the following command:

**Install-Module -Name Az -AllowClobber -Scope CurrentUser**

This will install the module for the current user only. If you want to install it for all users, change the command to:

**Install-Module -Name Az -AllowClobber -Scope AllUsers**

By default, the **PowerShell** gallery isn't configured as a trusted repository for **PowerShellGet**. The first time you use the **PSGallery** you see a prompt that asks if you would like to continue with the installation

Answer **Yes** or **Yes to All** to continue with the installation

Now, that we have the **Az module** installed, we can ask what commands are provided by it:

**Get-Command -Module Az.\***

It is a huge list. We can browse it page by page (or line by line) with:

**Get-Command -Module Az.\* | More**

Or count how many commands are there with:

**Get-Command -Module Az.\* | Measure-Object -Line**

We can always ask for help how a command is used with:

**Get-Help New-AzVM**

### Create Simple Solution with PowerShell

Before we can do anything with our **Azure** subscription from **PowerShell**, we must login first:

**Connect-AzAccount**

Then, we must enter our credentials in the window that appeared

Let’s create a simple but complete set of resources from the **Azure CLI**

We will create a resource group and a Linux-based VM, then we will install a web server and open a port

To create the resource group, we must execute:

**New-AzResourceGroup -Name RG-Demo-4 -Location WestEurope**

Then, we can create the VM (same as with the **Azure CLI** earlier) with:

**New-AzVM -Name "VM-Ubuntu" -Location "WestEurope" -ResourceGroupName "RG-Demo-4" `**

**-Size "Standard\_B1s" -Credential (Get-Credential) -Image "UbuntuLTS"**

Get the VM’s public IP address with:

**(Get-AzVM -ResourceGroupName "RG-Demo-4" -VMName "VM-Ubuntu" `**

**| Get-AzPublicIpAddress).IpAddress**

Knowing the public IP address, we can connect to the VM and install a web server as done in **Part 2**

As with the previous experiments, the demo web site is not reachable from the outside, so we must open a port:

**Get-AzNetworkSecurityGroup -ResourceGroupName "RG-Demo-4" `**

**| Add-AzNetworkSecurityRuleConfig -Name "HTTP-Port-80" -Description "Allow HTTP Port 80" `**

**-Access "Allow" -Protocol "Tcp" -Direction "Inbound" -Priority 100 `**

**-SourceAddressPrefix "Internet" -SourcePortRange "\*" `**

**-DestinationAddressPrefix "\*" -DestinationPortRange "80" `**

**| Set-AzNetworkSecurityGroup**

The site must be reachable now

#### Add Additional Disk via the Azure PowerShell

With **PowerShell** this action is a little bit longer

At first, we create a disk configuration:

**$diskConfig = New-AzDiskConfig -SkuName "Standard\_LRS" -Location "West Europe" -CreateOption Empty -DiskSizeGB 16**

Then we create the actual disk:

**$dataDisk = New-AzDisk -DiskName "VM-Ubuntu\_Disk2" -Disk $diskConfig -ResourceGroupName "RG-Demo-4"**

Az next step, we are getting the VM:

**$vm = Get-AzVM -VMName "VM-Ubuntu" -ResourceGroupName "RG-Demo-4"**

Then, we attach the disk to it:

**$vm = Add-AzVMDataDisk -VM $vm -Name "VM-Ubuntu\_Disk2" -CreateOption Attach -ManagedDiskId $dataDisk.Id -Lun 0**

Finally, we update the VM configuration:

**Update-AzVM -VM $vm -ResourceGroupName "RG-Demo-4"**

The rest of the steps to partition the disk and create a filesystem are the same as the ones in **Part 2**

## A Reminder

Don’t forget to delete resources that you do not need

In **Azure CLI**, to be platform independent (as may use it under **Bash** or **CMD**), you can execute the following commands to delete all four resource groups:

**az group delete --name RG-Demo-1 --yes --no-wait**

**az group delete --name RG-Demo-2 --yes --no-wait**

**az group delete --name RG-Demo-3 --yes --no-wait**

**az group delete --name RG-Demo-4 --yes --no-wait**

In **PowerShell**, no matter on which OS, the same procedure can be shortened to look like:

**Get-AzResourceGroup RG-Demo-\* | Remove-AzResourceGroup -Force**