



Build COMPETENCY across your TEAM

With our comprehensive training interventions for your Architects,
Developers, IT Pro's and Sales & Presales teams

Understanding Azure Resource Manager

Sonu Sathyadas

What is Azure Resource Manager?

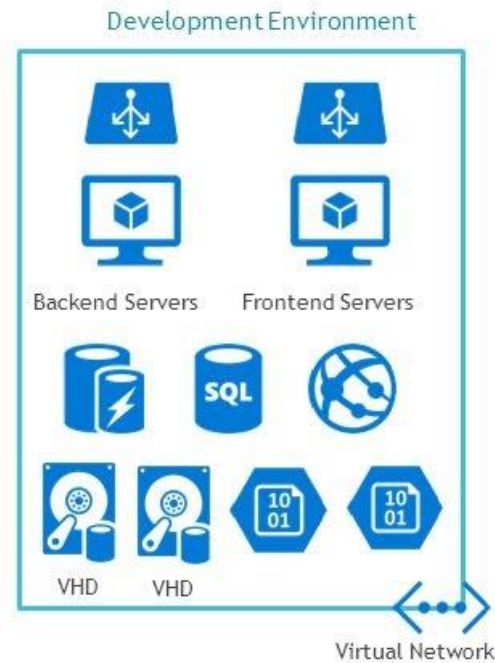
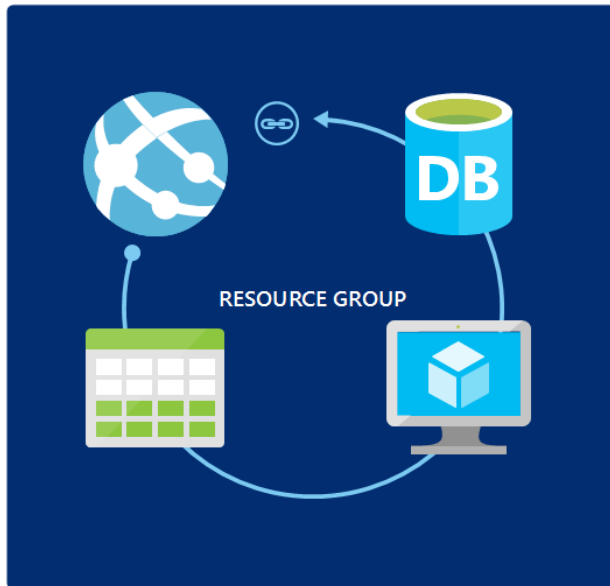
- Application infrastructure consists of many components – VM, storage, network, database etc.
- User need to deploy, manage and monitor them as a group.
- Azure Resource Manager (ARM) enables user to work on them as a group.
- User can deploy, update and delete resources for the solution in a single, coordinated operation.
- Resource Manager provides security, auditing and tagging features for resources.
- Azure Resource Manager enables you to repeatedly deploy your app and have confidence your resources are deployed in a consistent state.

- **Resource:-** A manageable item that is available through Azure. Eg: VM, storage account, Web App, Virtual Network etc.
- **Resource Group:-** A container that holds related resources for an Azure solution. Resource Group contains *related resources* for a solution.
- **Resource Provider:-** A service that supplies the resources you can deploy and manage through Resource Manager. eg: Microsoft.Compute, Microsoft.Storage, Microsoft.Web.
- **Resource Manager template:-** A JSON file that defines one or more resources to deploy to a resource group. It also defines the *dependencies* between the deployed resources.
- **Declarative syntax:-** Syntax that tells “What to be created” without writing commands. ARM template is an example.

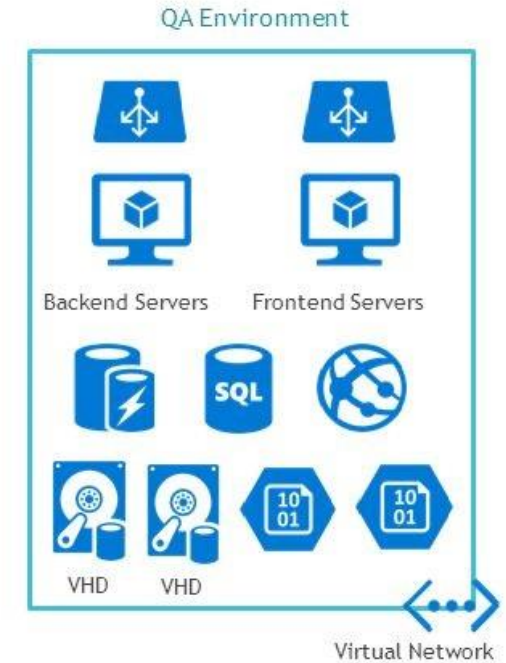
Why Resource Manager?

- Create, Deploy and Manage resources as a group.

RESOURCE GROUP PATTERNS - ENVIRONMENT

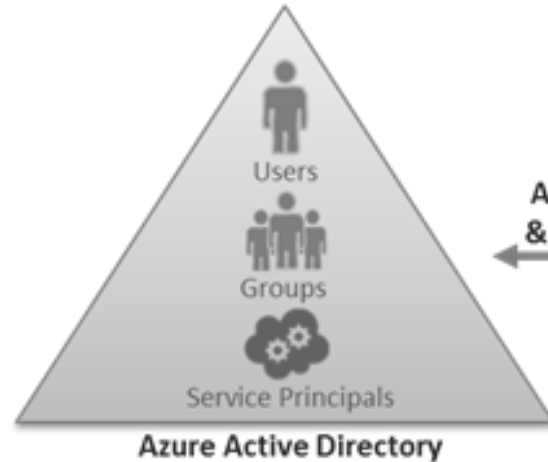


Resource Group
as
Container for
System Environment

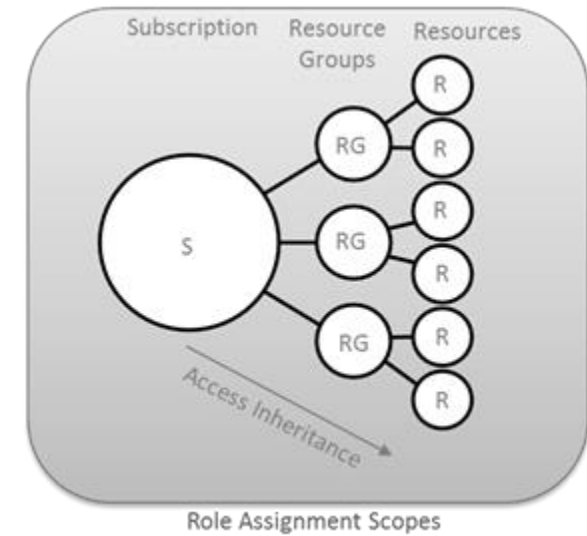
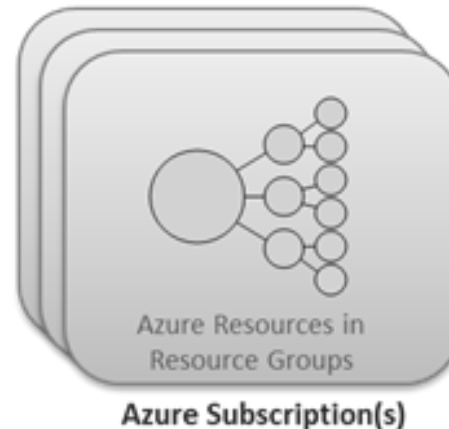


Why Resource Manager?

■ Role Based Access Control - RBAC



Authentication
& Authorization



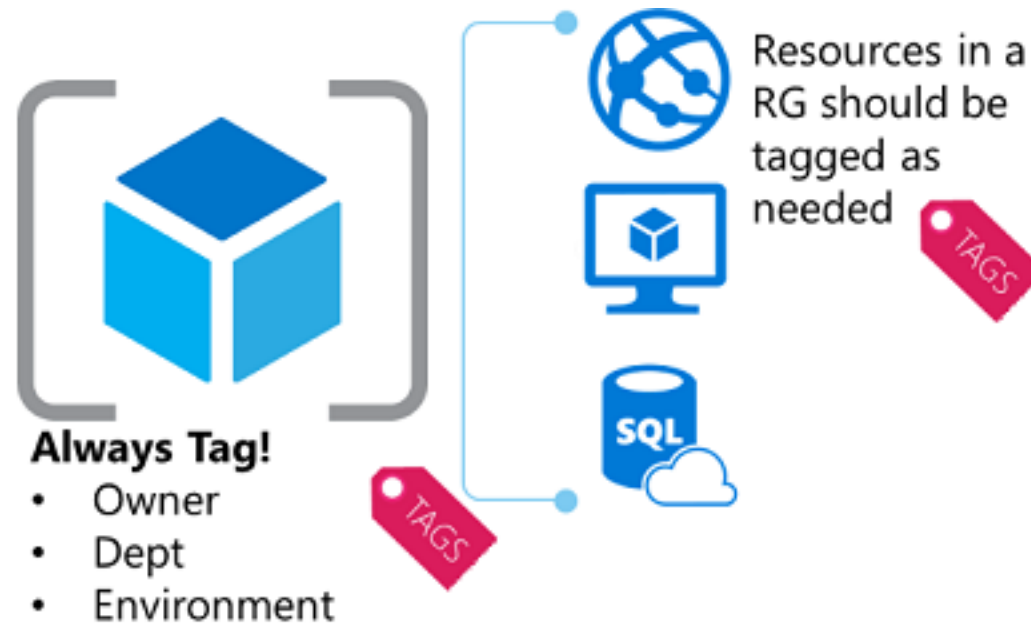
Why Resource Manager?

- Declarative code vs Imperative code.
- Azure RM Templates

```
{
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "adminUsername": {
      "type": "string",
      "metadata": {
        "description": "Username for the Virtual Machine."
      }
    },
    "adminPassword": {
      "type": "securestring",
      "metadata": {
        "description": "Password for the Virtual Machine."
      }
    },
    "dnsLabelPrefix": {
      "type": "string",
      "metadata": {
        "description": "Unique DNS Name for the Public IP used to access the Virtual Machine."
      }
    },
    "windowsOSVersion": {
      "type": "string",
      "defaultValue": "2012-R2-Datacenter",
      "allowedValues": [
        "2012-R2-Datacenter",
        "2012-R2-Datacenter",
        "2012-R2-Datacenter"
      ]
    }
  }
}
```

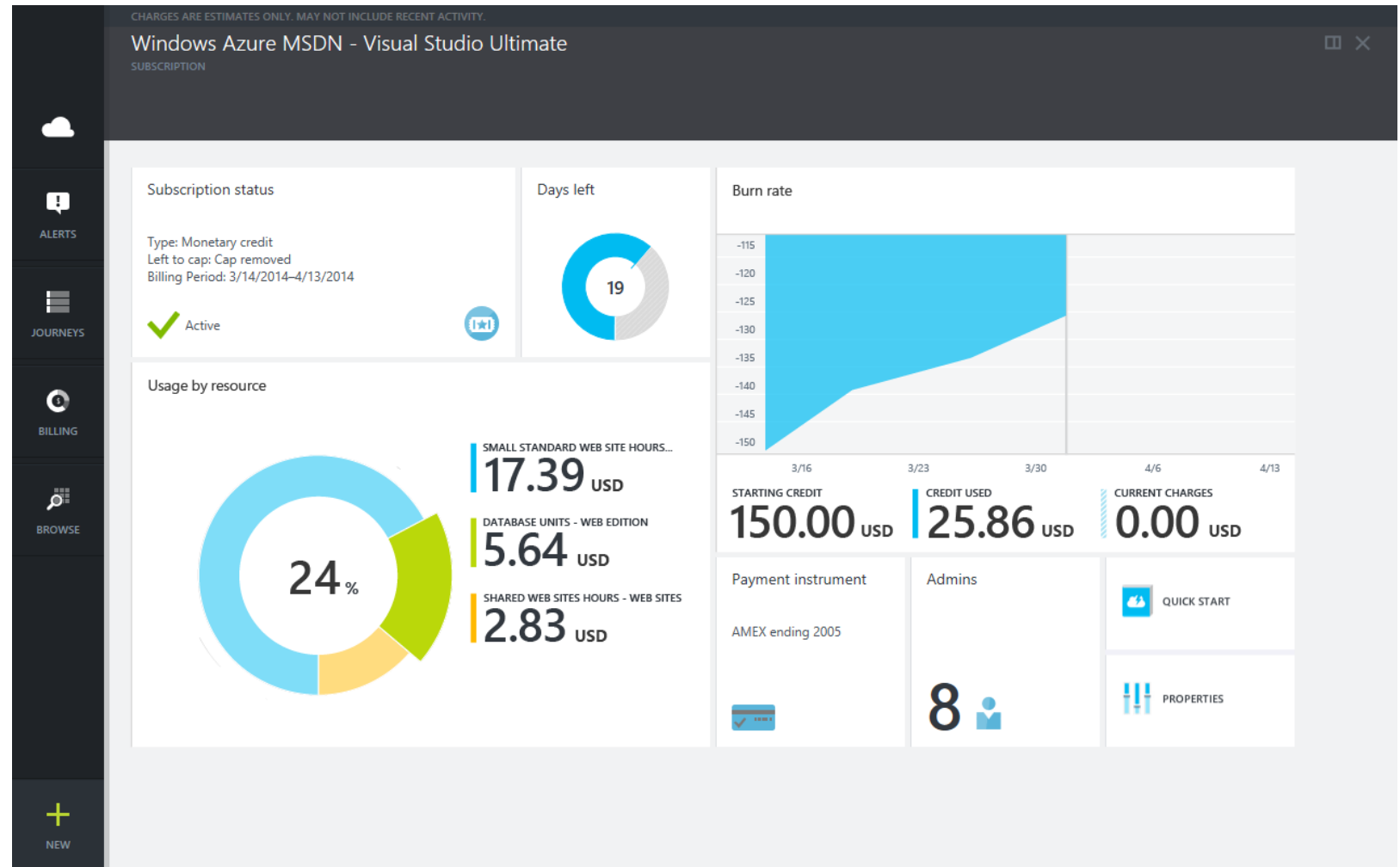
Why Resource Manager?

- Apply tags to resources to logically organize all the resources.



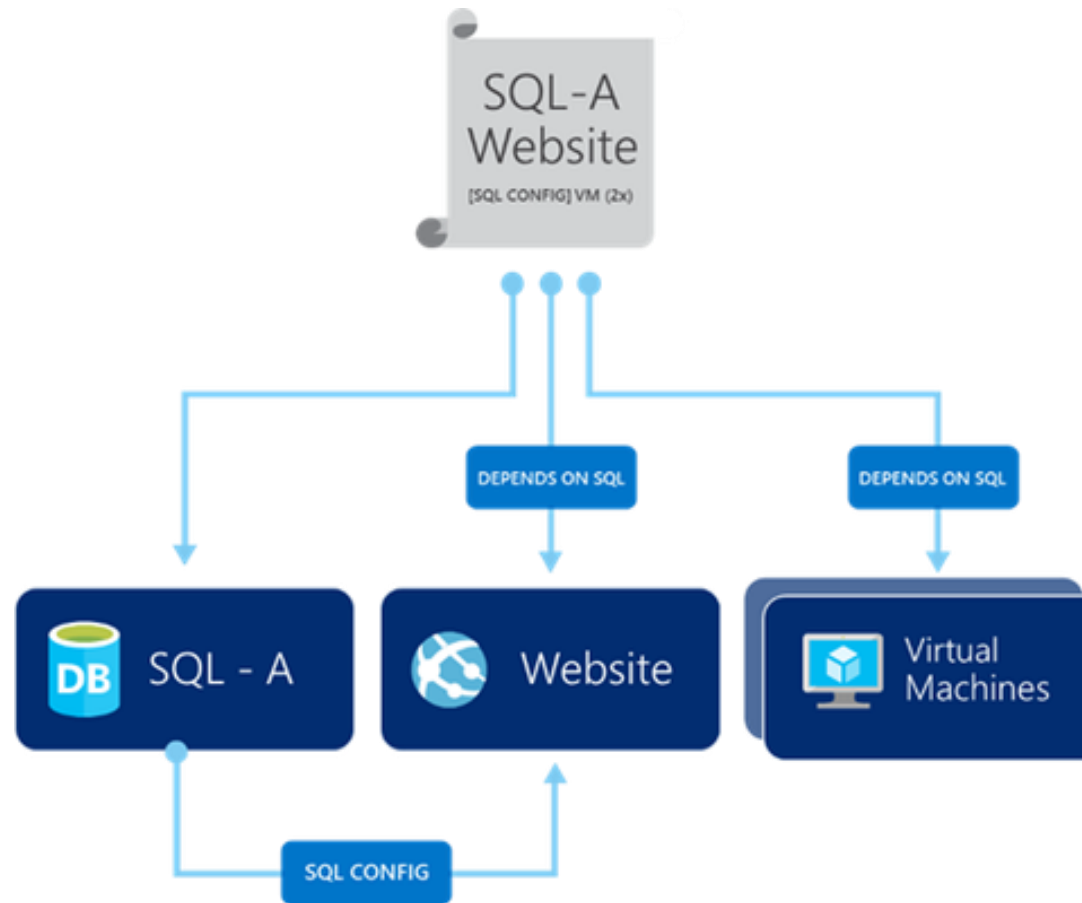
Why Resource Manager?

- Billing for Groups
- Cost for resources



Why Resource Manager?

- Repeated deployment using templates.



- A single entity in Azure RM
- Eg: VM, Storage Account, LB, Cloud Service etc
- We can group them using Resource Group
- Provided by Resource Providers
 - Storage Account => Microsoft.Storage
 - Virtual Machine => Microsoft.Compute
 - Load Balancer => Microsoft.Network
 - Virtual Network => Microsoft.Network
 - Cloud Service => Microsoft.Compute

- Each resource provider offers set of resources.
- Retrieve list of resource providers using following PS command

Powershell: **Get-AzureRmResourceProvider -ListAvailable**

Azure CLI : **azure provider list**

- To get details about a resource provider, add the provider namespace to your command.

Powershell:

(Get-AzureRmResourceProvider -ProviderNamespace Microsoft.Compute).ResourceTypes

Azure CLI:

azure provider show Microsoft.Compute --json > c:\Azure\compute.json

Why Resource Group ?

Microsoft Azure | CREDIT STATUS | Subscriptions | paulschnack@hotmail.com

ALL ITEMS

all items

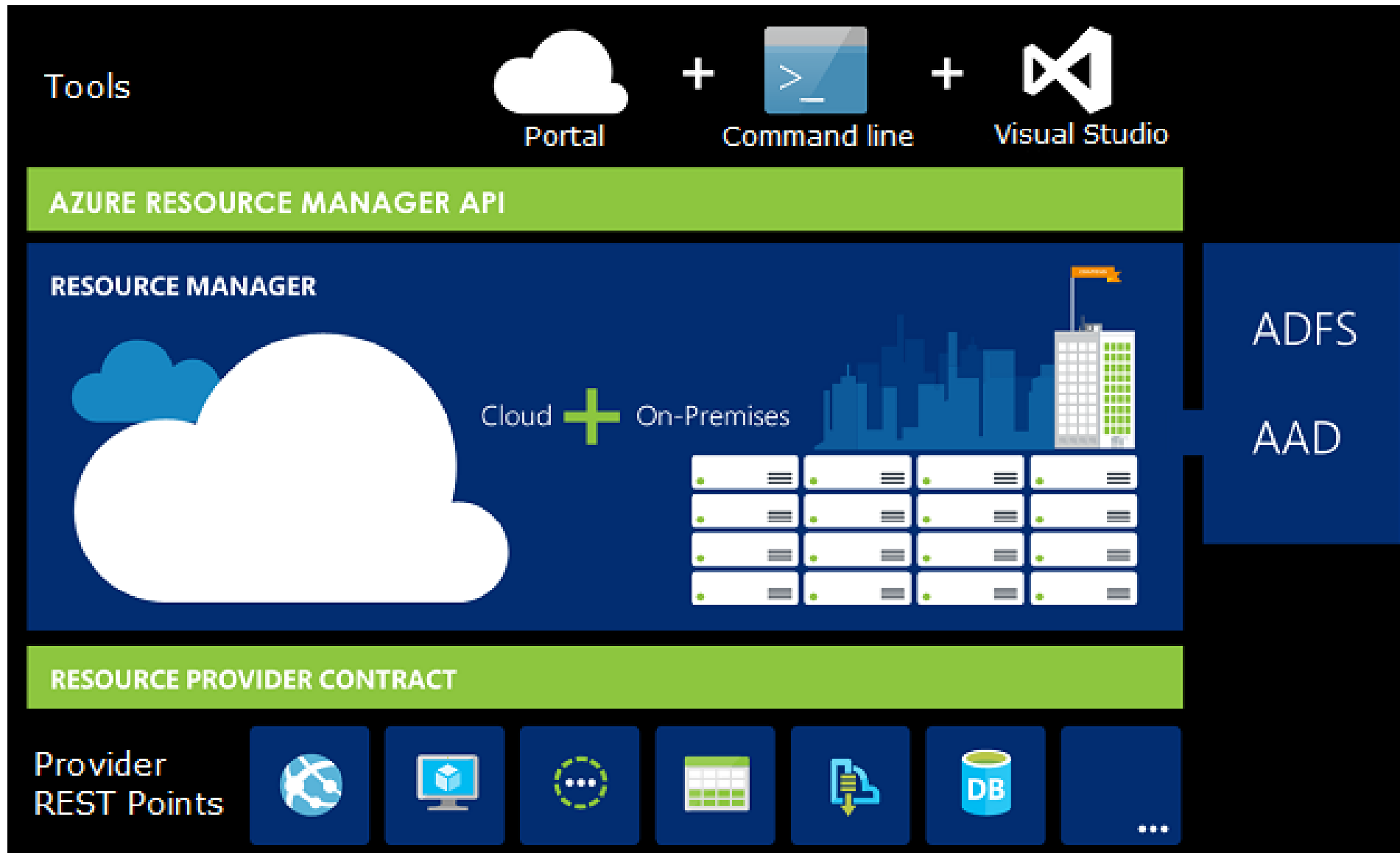
NAME	TYPE	STATUS	SUBSCRIPTION	LOCATION
JABB-DC1	Cloud service	Stopped	Microsoft Partner Network	East US
expertit	Web app	Running	Microsoft Partner Network	East Asia
jabbsysprodvms	Storage Account	Online	Microsoft Partner Network	East US
paulschnack	Storage Account	Online	Microsoft Partner Network	Southeast Asia
portalvhds5jxbgjd5hrzc	Storage Account	Online	Microsoft Partner Network	Australia Southeast
portalvhds7cm43qxh7n11k	Storage Account	Online	Microsoft Partner Network	West US
portalvhdsxws04twbhgbf5	Storage Account	Online	Microsoft Partner Network	East Asia
JABB-ASR	Azure Site Recovery...	Active	Microsoft Partner Network	Australia East
jabbsysprodbackup	Backup Vault	Active	Microsoft Partner Network	East US
Azure	Virtual Network	Created	Microsoft Partner Network	Southeast Asia
Group Group JABBSysProd	Virtual Network	Created	Microsoft Partner Network	East US
Tenant1_Vnet	Virtual Network	Created	Microsoft Partner Network	East US
4Sysops	Directory	Active	Shared by all 4Sysops su...	Asia, Europe, United...
JABB Systems	Directory	Active	Shared by all JABB Syste...	United States
Default Directory	Directory	Active	Shared by all Default Dir...	Asia, Europe, United...
expertit8587-bts	Active Directory Acc...	Active	Microsoft Partner Network	East Asia
Sysops	Automation Account	Ready	Microsoft Partner Network	East US 2
az597066	CDN Endpoint	Enabled	Microsoft Partner Network	-
JABB-DC1	Virtual machine	Stopped (D...	Microsoft Partner Network	East US

Resource groups – factors/guidelines

- All the resources in a group should share the same lifecycle. Deploy, update, and delete them together.
- Each resource can only exist in one resource group.
- Can add or remove a resource to a resource group at any time.
- Can move a resource from one resource group to another group.
- A resource group can contain resources that reside in different regions.
- A resource group can be used to scope access control for administrative actions (RBAC).
- A resource can interact with resources in other resource groups. This interaction is common when the two resources are related but do not share the same lifecycle.

- RM provides a consistent management layer for the tasks we perform through Azure Powershell, Azure CLI, Azure portal, REST API and development tools.
 - The API passes requests to the Resource Manager service, which authenticates and authorizes the requests.
 - Resource Manager then routes the requests to the appropriate resource providers.

Consistent Management Layer



Classic VS RM deployments

- In classic model resources are deployed independently, no groups.
- Difficult to maintain the order if dependent resources exists.
- Difficult to track related/dependent resources for a solution.
- RM deploys resources a group, manage as group and can delete as a group.
- Applies policies for a group of resources.
- Tags to logically organize the resources.
- Template support for RM
- Define dependencies and deploy in order.

Resource Manager with PowerShell

List all Providers and Resources

Login-AzureRmAccount

Set-AzureRmContext -SubscriptionName '<Subscription name>'

Get-AzureRmResourceProvider -ListAvailable

(Get-AzureRmResourceProvider -ProviderNamespace Microsoft.Web).ResourceTypes

ARM Templates

- ARM template is a JSON file that defines the infrastructure and configuration of your Azure solution.
- Dependencies can be defined in ARM templates.
- When you create a solution from the portal, the solution automatically includes a deployment template.
- Resource Manager processes the template like any other request, parses the template and converts its syntax into REST API operations for the appropriate resource providers.

Template format

```
{
  "$schema": "http://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion": "",
  "parameters": { },
  "variables": { },
  "resources": [ ],
  "outputs": { }
}
```

Element name	Required	Description
\$schema	Yes	Location of the JSON schema file that describes the version of the template language.
contentVersion	Yes	Version of the template (such as 1.0.0.0). You can provide any value for this element. When deploying resources using the template, this value can be used to make sure that the right template is being used.
parameters	No	Values that are provided when deployment is executed to customize resource deployment.
variables	No	Values that are used as JSON fragments in the template to simplify template language expressions.
resources	Yes	Resource types that are deployed or updated in a resource group.
outputs	No	Values that are returned after deployment.

- With expressions, you create values that are not strict literal values.
- Expressions are enclosed with brackets [and].
- Evaluated when the template is deployed.
- Expressions can appear anywhere in a JSON string value and always return another JSON value.
- To use a literal string that starts with a bracket [, you must use two brackets [[.
- Use expressions with functions to perform operations for configuring the deployment.
- Function calls are formatted as functionName(arg1,arg2,arg3).

Expressions and functions in template

```
"variables": {  
  "location": "[resourceGroup().location]",  
  "usernameAndPassword": "[concat(parameters('username'), ':', parameters('password'))]",  
  "authorizationHeader": "[concat('Basic ', base64(variables('usernameAndPassword')))]"  
}
```

<https://docs.microsoft.com/en-us/azure/azure-resource-manager/resource-group-template-functions>

Parameters in Template

- Specify which values you can input when deploying the resources.
- Parameter values enable you to customize the deployment by providing values that are tailored for a particular environment.
- Only parameters that are declared in the parameters section can be used in other sections of the template.

```
"parameters": {  
  "<parameter-name>" : {  
    "type" : "<type-of-parameter-value>",  
    "defaultValue": "<default-value-of-parameter>",  
    "allowedValues": [ "<array-of-allowed-values>" ],  
    "minValue": <minimum-value-for-int>,  
    "maxValue": <maximum-value-for-int>,  
    "minLength": <minimum-length-for-string-or-array>,  
    "maxLength": <maximum-length-for-string-or-array-parameters>,  
    "metadata": {  
      "description": "<description-of-the parameter>"  
    }  
  }  
}
```


Variables in Templates

- Construct values that can be used throughout the template.
- Typically, variables are based on values provided from the parameters.
- Not mandatory in templates.
- Simplify the template by reducing the complexity of expressions.

```
"variables": {  
    "<variable-name>": "<variable-value>",  
    "<variable-name>": {  
        <variable-complex-type-value>  
    }  
}
```

Resources in Templates

- Define the resources that are deployed or updated.

```
"resources": [  
  {  
    "apiVersion": "<api-version-of-resource>",  
    "type": "<resource-provider-namespace/resource-type-name>",  
    "name": "<name-of-the-resource>",  
    "location": "<location-of-resource>",  
    "tags": "<name-value-pairs-for-resource-tagging>",  
    "comments": "<your-reference-notes>",  
    "dependsOn": [  
      "<array-of-related-resource-names>"  
    ],  
    "properties": "<settings-for-the-resource>",  
    "copy": {  
      "name": "<name-of-copy-loop>",  
      "count": "<number-of-iterations>"  
    }  
    "resources": [  
      "<array-of-child-resources>"  
    ]  
  }  
]
```

Input Template file

```
"resources": [  
  {  
    "apiVersion": "2016-01-01",  
    "type": "Microsoft.Storage/storageAccounts",  
    "name": "mystorageaccount",  
    "location": "westus",  
    "sku": {  
      "name": "Standard_LRS"  
    },  
    "kind": "Storage",  
    "properties": { }  
  }  
]
```

Output API

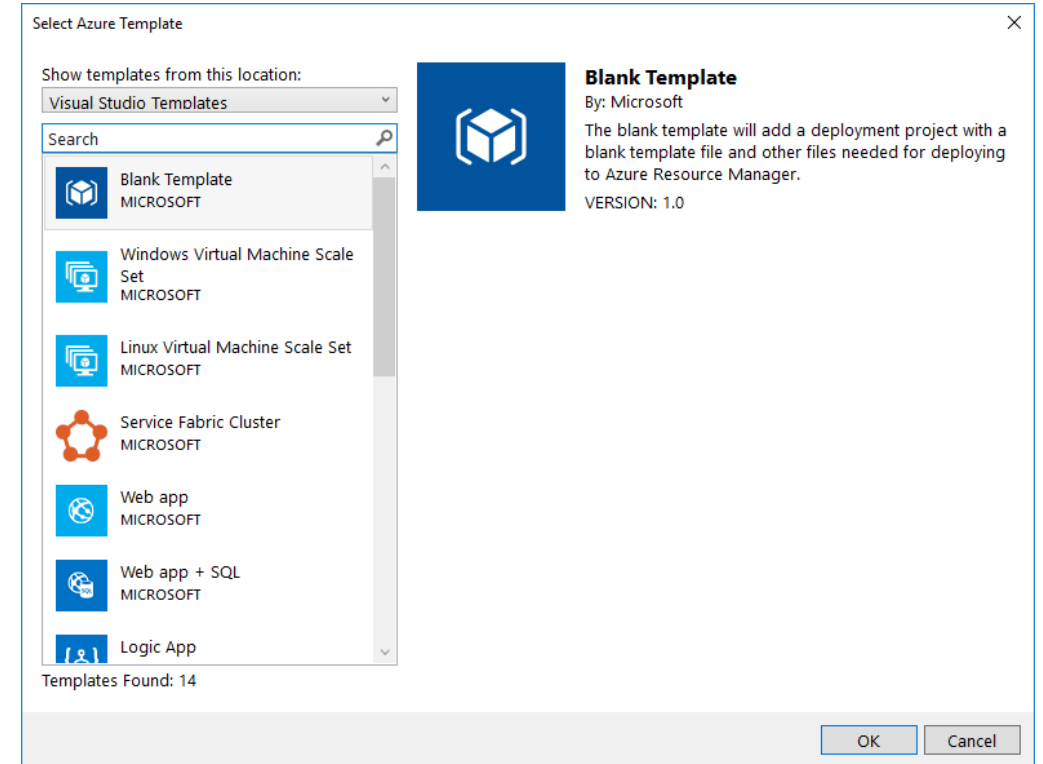
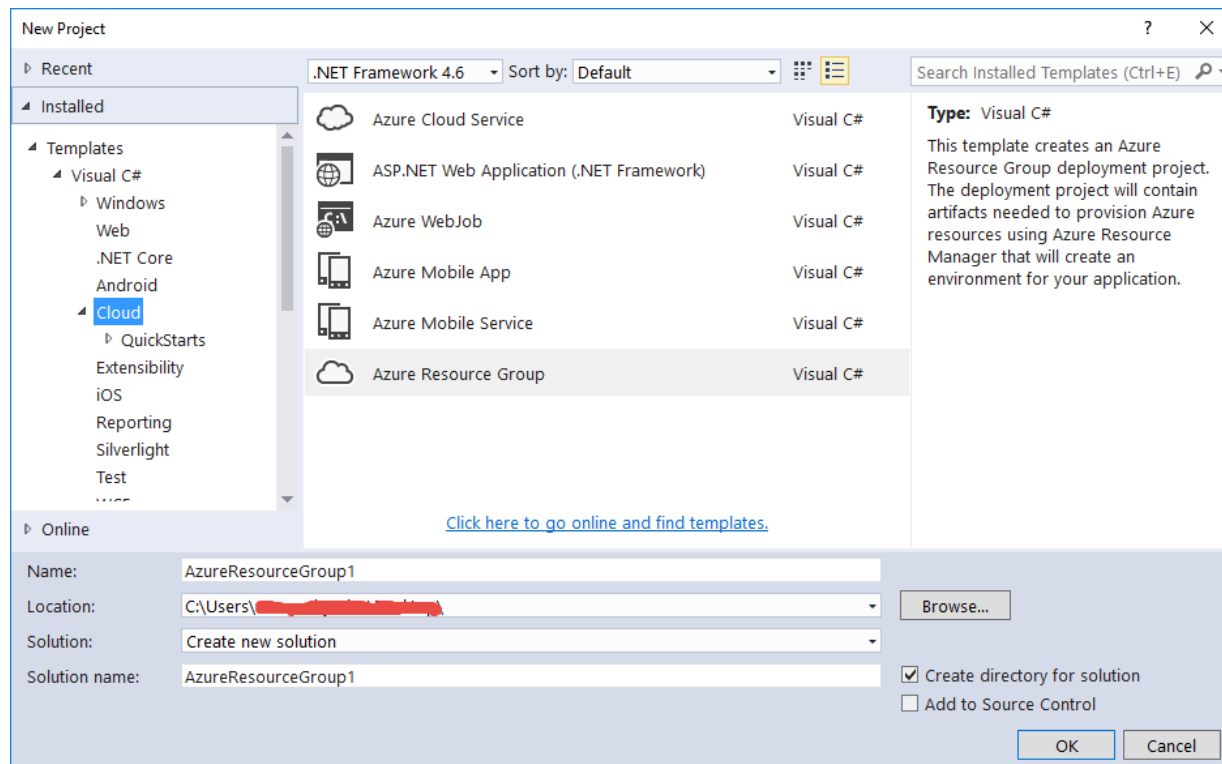
```
PUT  
https://management.azure.com/subscriptions/{subscriptionId}/resourceGroups/{resourceGroupName}/providers/Microsoft.Storage/storageAccounts/mystorageaccount?api-version=2016-01-01  
REQUEST BODY  
{  
  "location": "westus",  
  "properties": {  
  },  
  "sku": {  
    "name": "Standard_LRS"  
  },  
  "kind": "Storage"  
}
```

DEMO

Azure RM Templates

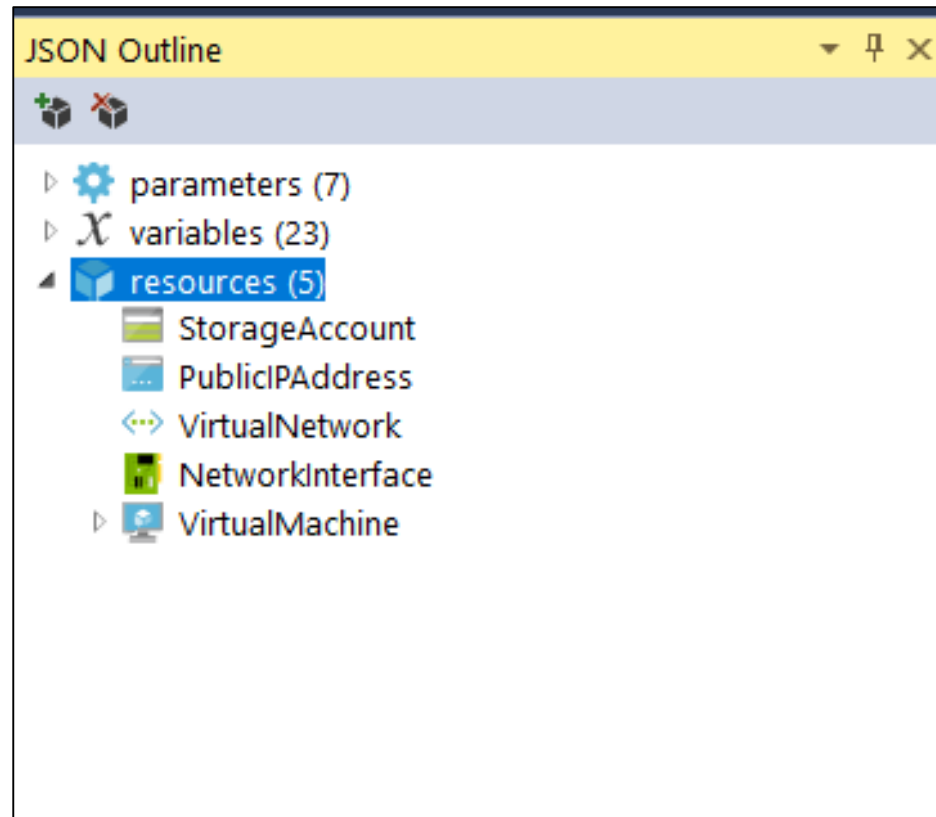
Templates using Visual Studio

- Microsoft Azure Tools 2.6 or later
- Visual Studio 2015 or later
- File > New Project > Cloud > Resource Group

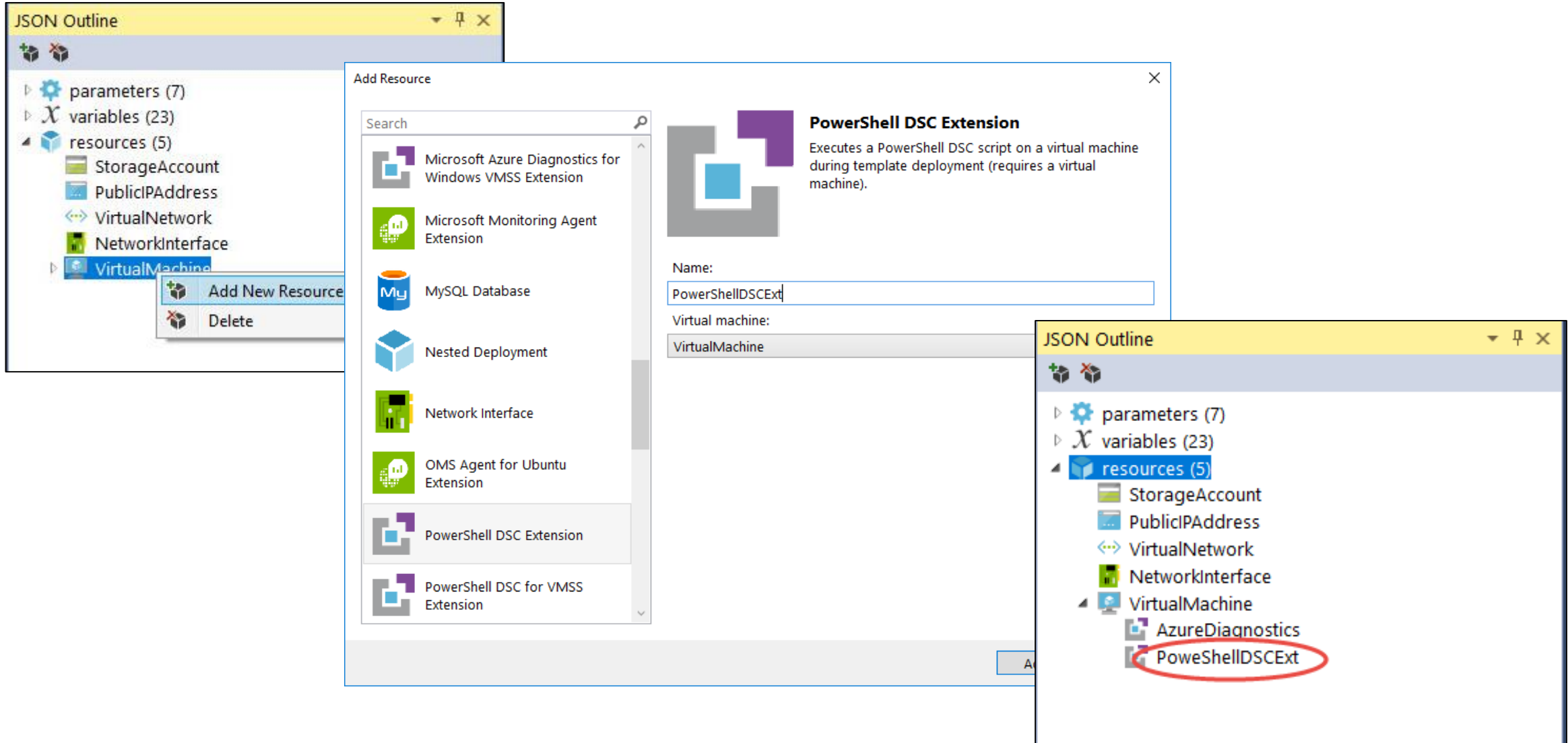


Templates using Visual Studio

- JSON Outline window shows the parameters, variables and resources



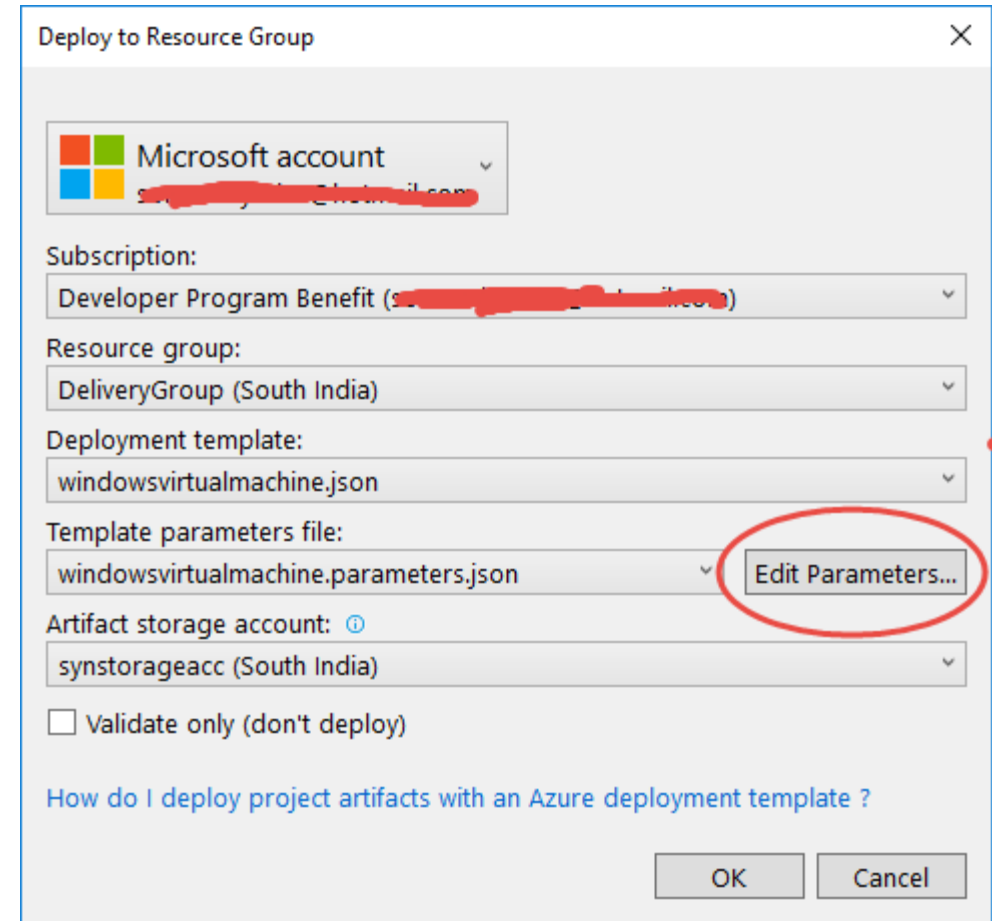
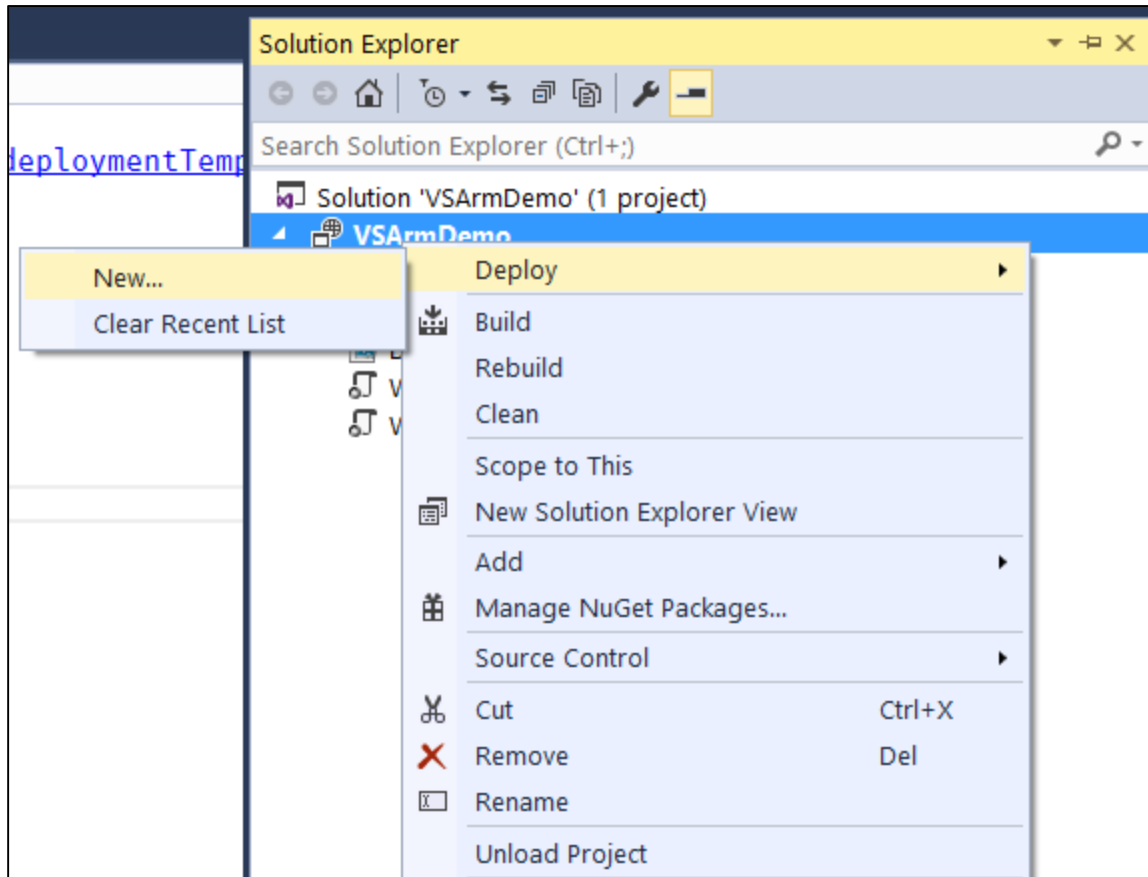
Adding resources to template using VS



The image shows a Visual Studio interface with three main components:

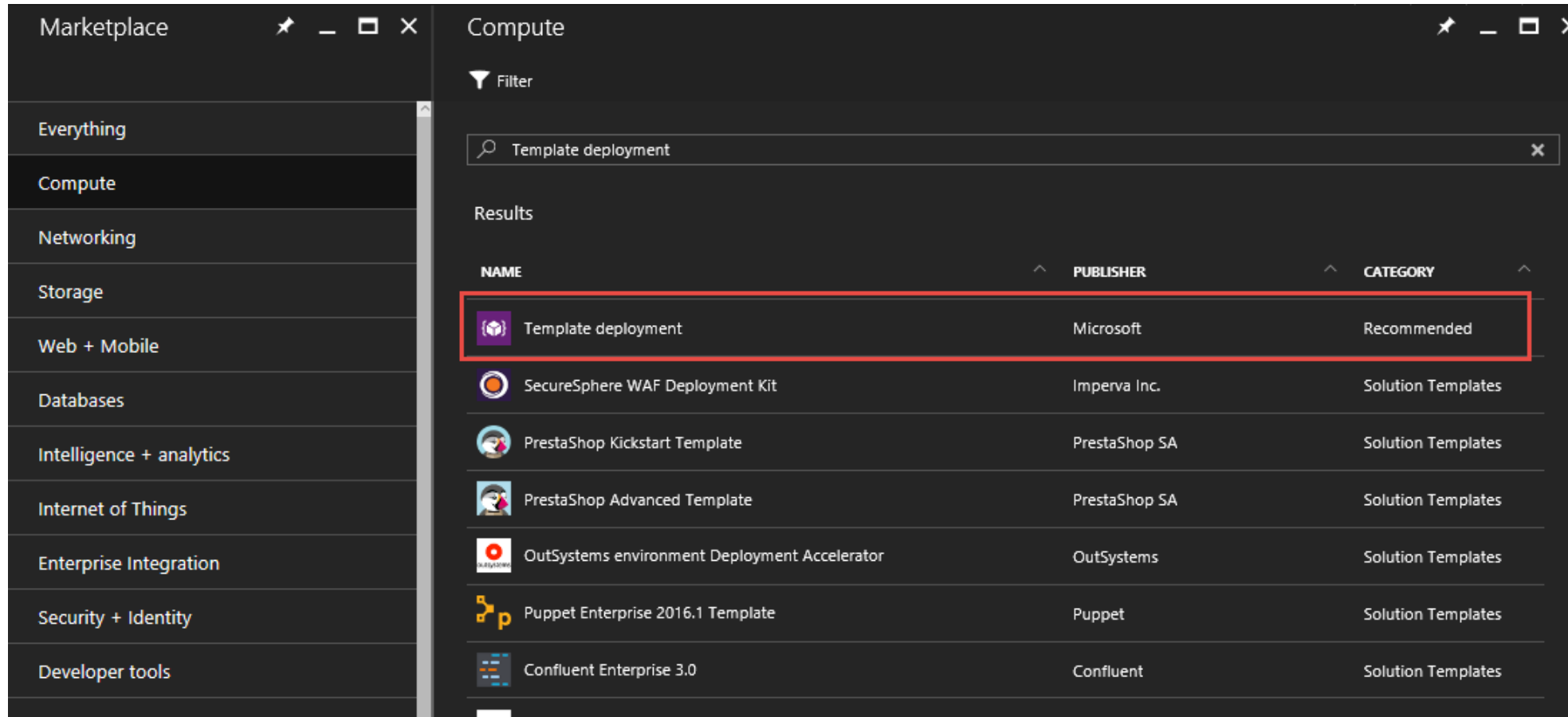
- JSON Outline (Left):** A tree view showing the template structure. It includes `parameters (7)`, `variables (23)`, and `resources (5)`. The `resources` section lists `StorageAccount`, `PublicIPAddress`, `VirtualNetwork`, `NetworkInterface`, and `VirtualMachine`. A context menu is open over `VirtualMachine` with options `Add New Resource` and `Delete`.
- Add Resource (Center):** A dialog box for selecting a resource. It features a search bar and a list of resources. The `PowerShell DSC Extension` is selected. The `Name` field contains `PowerShellDSCExt` and the `Virtual machine` field contains `VirtualMachine`.
- JSON Outline (Right):** A second instance of the JSON Outline window showing the updated template. The `resources (5)` section now includes `AzureDiagnostics` and `PoweShellDSCExt` (circled in red).

Deploying using Visual Studio



Deploying using Portal

- Navigate to <https://portal.azure.com> and login to Azure account.
- Click on the **+(New)** to add a **Template deployment**.



The screenshot shows the Azure Marketplace interface. On the left is a sidebar with categories: Everything, Compute, Networking, Storage, Web + Mobile, Databases, Intelligence + analytics, Internet of Things, Enterprise Integration, Security + Identity, and Developer tools. The main panel is titled 'Compute' and contains a search bar with the text 'Template deployment'. Below the search bar, a table of results is displayed. The first row, 'Template deployment' by Microsoft, is highlighted with a red border. The table has columns for NAME, PUBLISHER, and CATEGORY.

NAME	PUBLISHER	CATEGORY
Template deployment	Microsoft	Recommended
SecureSphere WAF Deployment Kit	Imperva Inc.	Solution Templates
PrestaShop Kickstart Template	PrestaShop SA	Solution Templates
PrestaShop Advanced Template	PrestaShop SA	Solution Templates
OutSystems environment Deployment Accelerator	OutSystems	Solution Templates
Puppet Enterprise 2016.1 Template	Puppet	Solution Templates
Confluent Enterprise 3.0	Confluent	Solution Templates

Deploying using Portal

Custom deployment

Deploy from a custom template

TEMPLATE



Template has no resources. Edit the template

Edit template

Edit your Azure Resource Manager template

+ Add resource ↑ Quickstart tem... ↓ Download

- Parameters (7)
- Variables (23)
- Resources (5)
 - StorageAccount (Microsoft.Storage...)
 - PublicIPAddress (Microsoft.Netwo...)
 - VirtualNetwork (Microsoft.Networ...)
 - NetworkInterface (Microsoft.Netw...)
 - VirtualMachine (Microsoft.Comput...)

```
1 {
2   "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
3   "contentVersion": "1.0.0.0",
4   "parameters": {
5     "adminUsername": {
6       "type": "string",
7       "minLength": 1,
8       "metadata": {
9         "description": "Username for the Virtual Machine."
10      }
11    },
12    "adminPassword": {
13      "type": "securestring",
14      "metadata": {
15        "description": "Password for the Virtual Machine."
16      }
17    },
18    "dnsNameForPublicIP": {
19      "type": "string",
20      "minLength": 1,
21      "metadata": {
22        "description": "Globally unique DNS Name for the Public IP used to access the Virtual Machine."
23      }
24    },
25    "windowsOSVersion": {
```

Copy and paste the template in the window, Click Save

Save

Discard

Deploying using Portal

- Specify the subscription, resource group name and parameter values.

Custom deployment

Deploy from a custom template

5 resources

EditLearn more

BASICS

* Subscription

Developer Program Benefit

* Resource group ⓘ

Create new

Use existing

ARMDemoGroup

* Location

Southeast Asia

SETTINGS

* Admin Username ⓘ

synadmin

✓

* Admin Password ⓘ

••••••••••

✓

* Dns Name For Public IP ⓘ

synarmdemopip

✓

Windows OS Version ⓘ

2012-R2-Datacenter

✓

* _artifacts Location ⓘ

☐ Pin to dashboard

Purchase

Deploying using PowerShell

```
Login-AzureRmAccount
```

```
Set-AzureRmContext -SubscriptionName '<subscription name>'
```

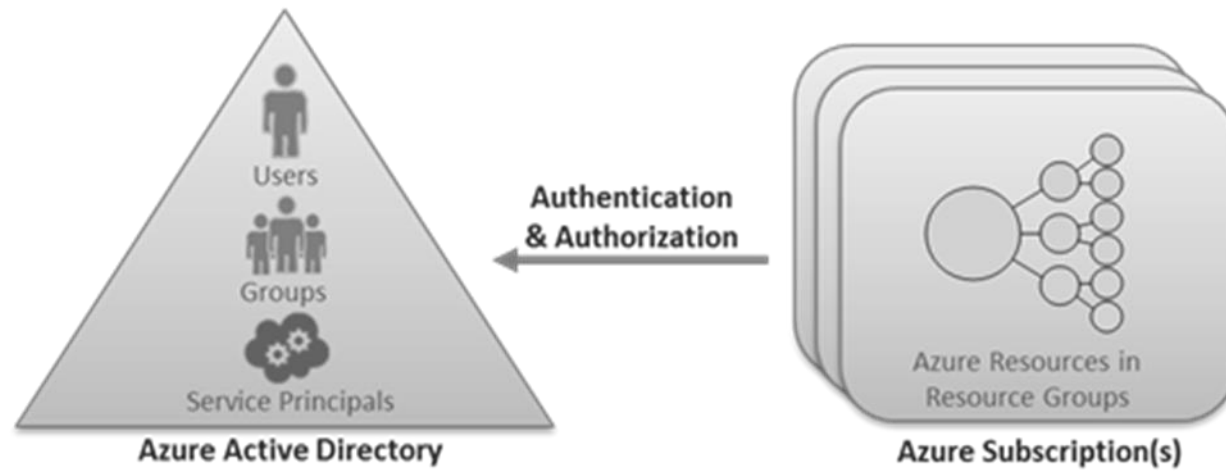
```
New-AzureRmResourceGroup -Name 'MyResourceGroup' `
                           -Location 'SouthEast Asia'
```

```
New-AzureRmResourceGroupDeployment -Name 'ARMDeploymentDemo' `
    -ResourceGroupName 'MyResourceGroup' `
    -TemplateFile 'WindowsVirtualMachine.json' `
    -TemplateParameterFile 'WindowsVirtualMachine.parameters.json' `
    -Force -Verbose
```

Role Based Access Control (RBAC)

Role Based Access Control (RBAC)

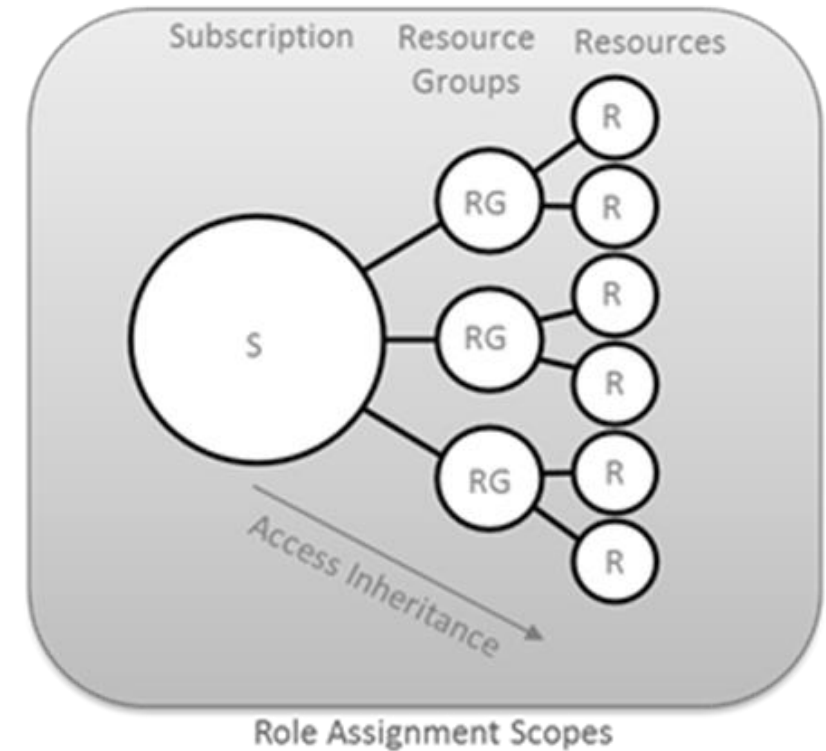
- RBAC is a way to manage the access to different resources in Azure.
- Provide user or group of users access to different parts of azure subscription.
- Azure Active Directory.



- Role
 - A collection of actions that can be performed on a resource.
 - Built-in Roles
- Role Assignment
 - Users
 - Groups
 - Service Principals

Role Based Access Control

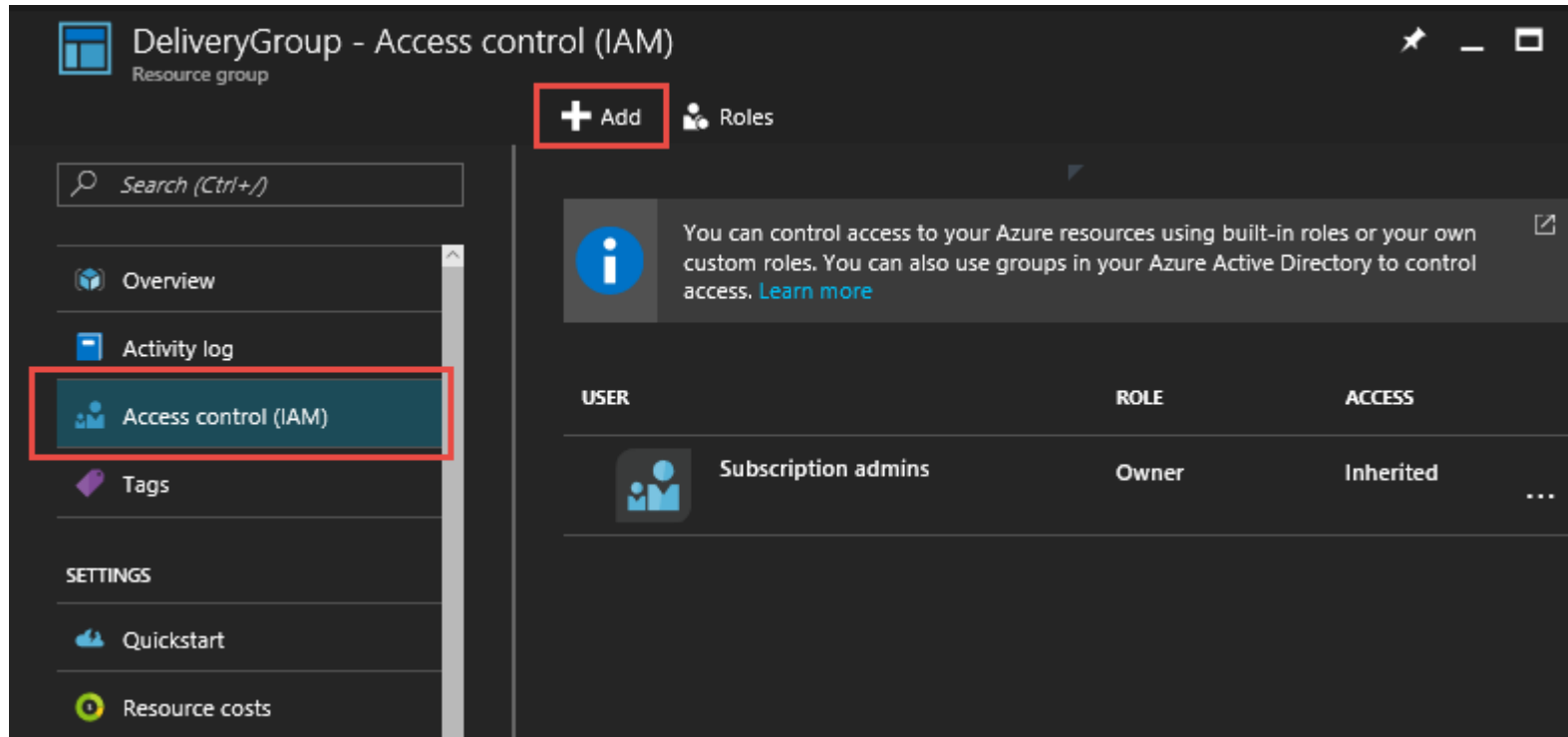
- Resource Scope
 - Access to a single resource
 - Access to a resource group
 - Inherited access policies



- Co-administrators and RBAC?
 - Full access
 - Subscription level
 - Owners of the subscription
 - RBAC does not provide access to old portal and APIs.
 - RBAC roles are not co-administrators

RBAC Users and Groups

- Add roles to users and groups on a resource group.
- Open portal and select the resource group.
- Select Access Control(IAM) from the settings blade.
- Click on '+Add' from command bar to add user or roles.



DeliveryGroup - Access control (IAM)
Resource group


+ Add Roles

Search (Ctrl+ /)

Overview
Activity log
Access control (IAM)
Tags

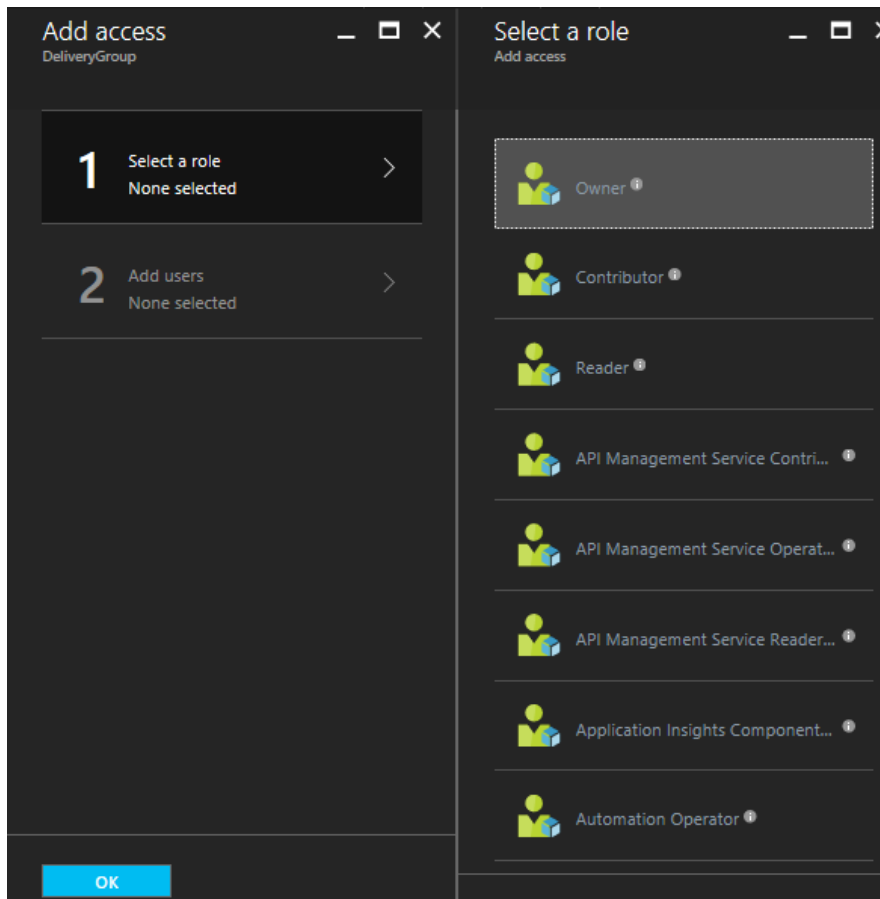
SETTINGS
Quickstart
Resource costs

USER **ROLE** **ACCESS**

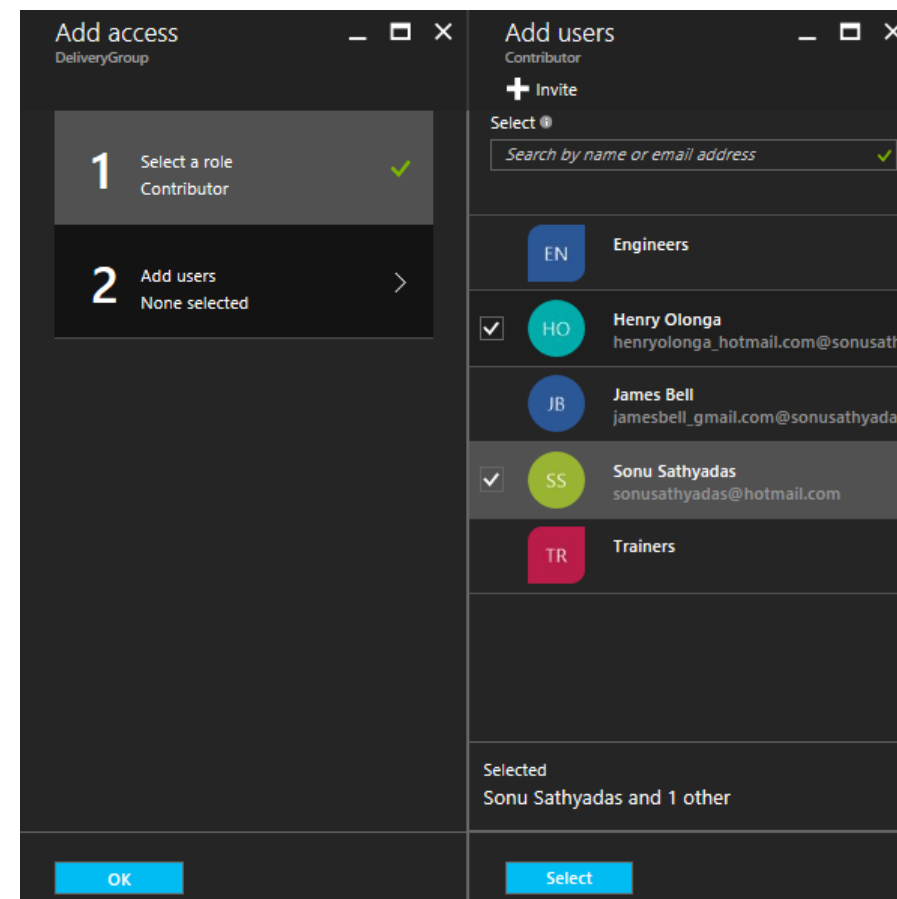
 Subscription admins	Owner	Inherited ...
-----------------------------------------------------------------------------------------------------------	-------	---------------

RBAC Users and Groups

- Choose the role and Users/Groups.
- User can be an external user also.



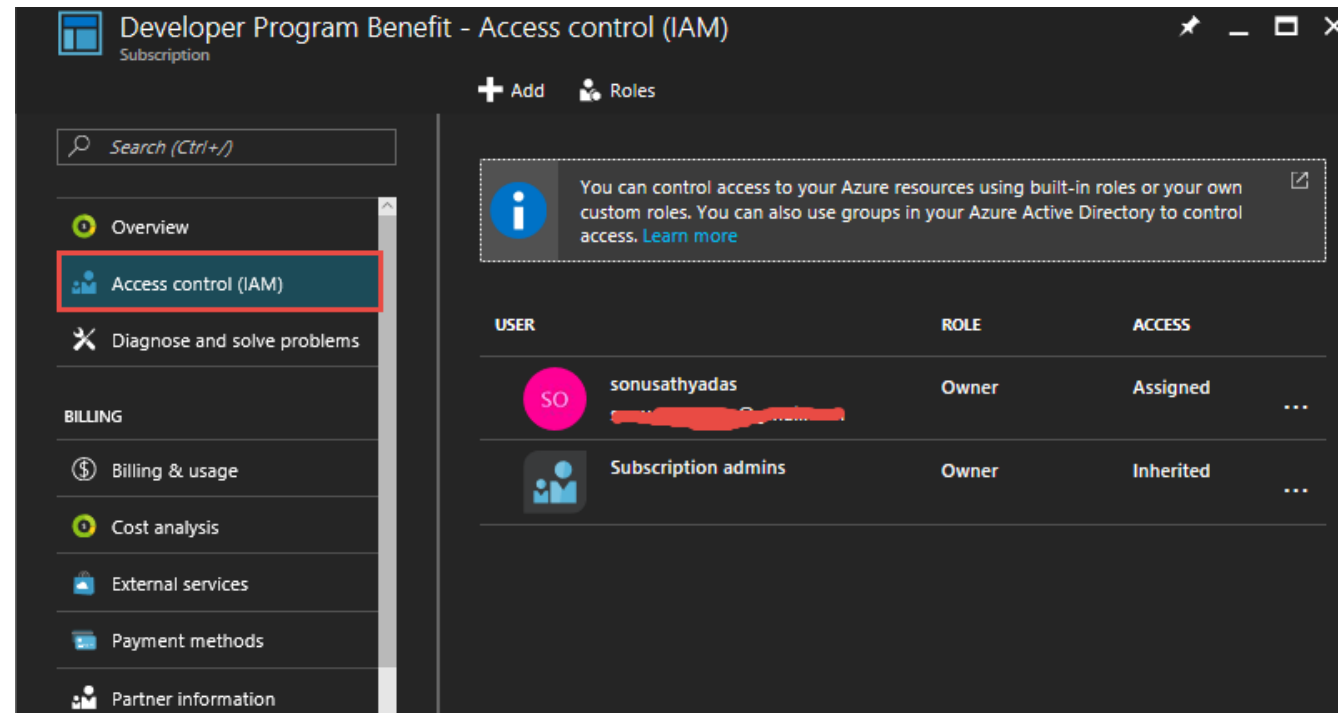
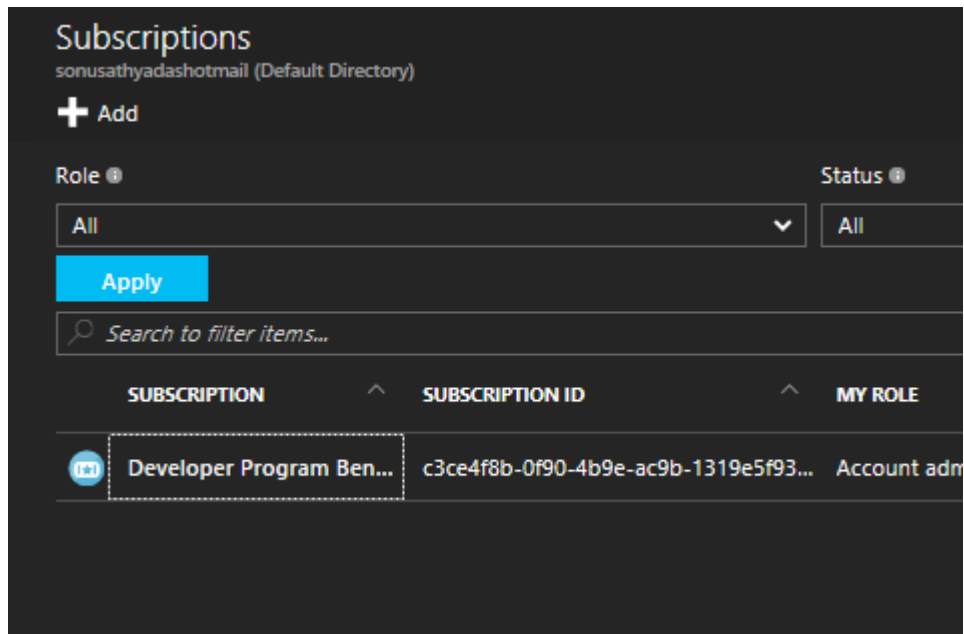
The screenshot shows the 'Add access' dialog for 'DeliveryGroup'. It has two main sections: 'Add access' on the left and 'Select a role' on the right. The 'Add access' section has two steps: '1 Select a role' (None selected) and '2 Add users' (None selected). The 'Select a role' section shows a list of roles: Owner (selected), Contributor, Reader, API Management Service Contri..., API Management Service Operat..., API Management Service Reader..., Application Insights Component..., and Automation Operator. An 'OK' button is at the bottom left.



The screenshot shows the 'Add access' dialog for 'DeliveryGroup' with the 'Add users' step selected. The 'Add access' section shows '1 Select a role' (Contributor) and '2 Add users' (None selected). The 'Add users' section has an 'Invite' button, a search bar, and a list of users: Engineers, Henry Olunga, James Bell, Sonu Sathyadas (selected), and Trainers. A 'Selected' section at the bottom shows 'Sonu Sathyadas and 1 other'. A 'Select' button is at the bottom right.

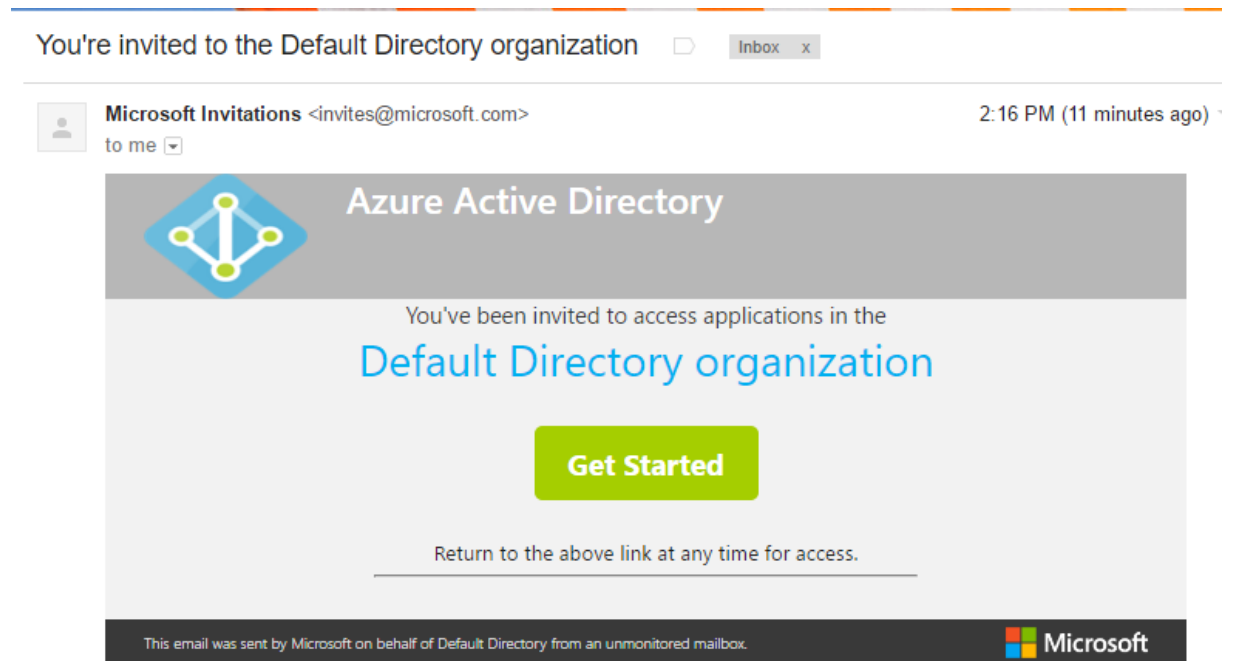
RBAC- Add external users to subscription

- Open Azure Portal and Navigate to subscriptions.
- Choose your subscription.



RBAC- Add external users to subscription

- Click on Add and choose role and username(email id).
- An email will be sent to the user.
- Click on it to add a Microsoft account.
- Verify the account after creating it.
- Login to Azure and start using it.



DEMO: RBAC using Powershell

Organize resources using Tags

- Logically organize resources.
- Key/value pairs that identify resources.
- Same tag for resources that belongs to same category.
- Not limited to only resources in the same resource group, across multiple resource groups.
- Organize resources for billing or management.
- Each resource or resource group can have a maximum of 15 tags.
- Tag name is limited to 512 characters, and the tag value is limited to 256 characters.

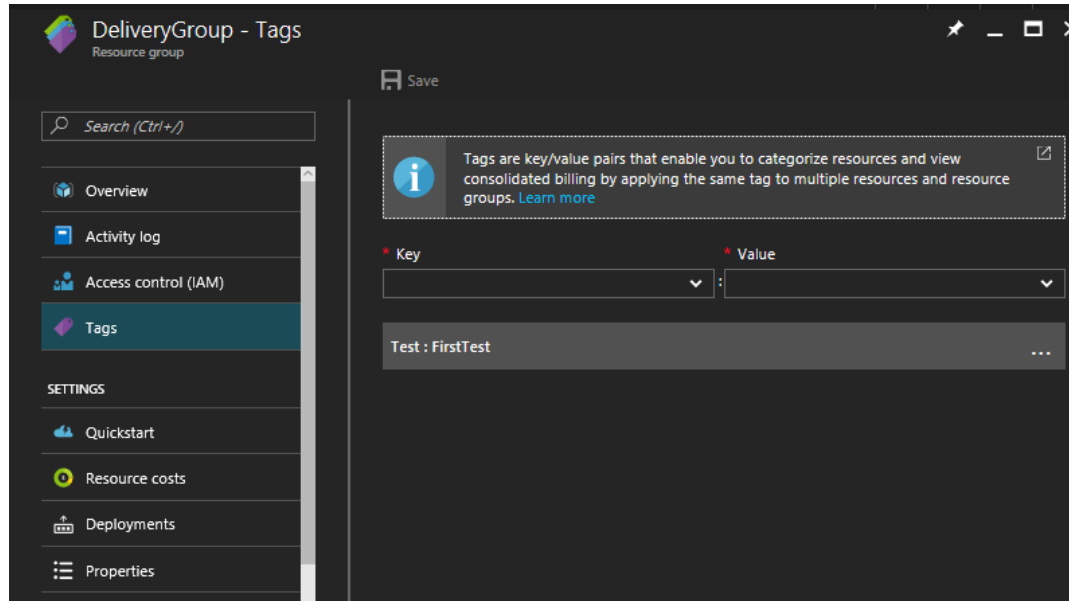
Tag a resource using template

- Add the **tags** element to the resource

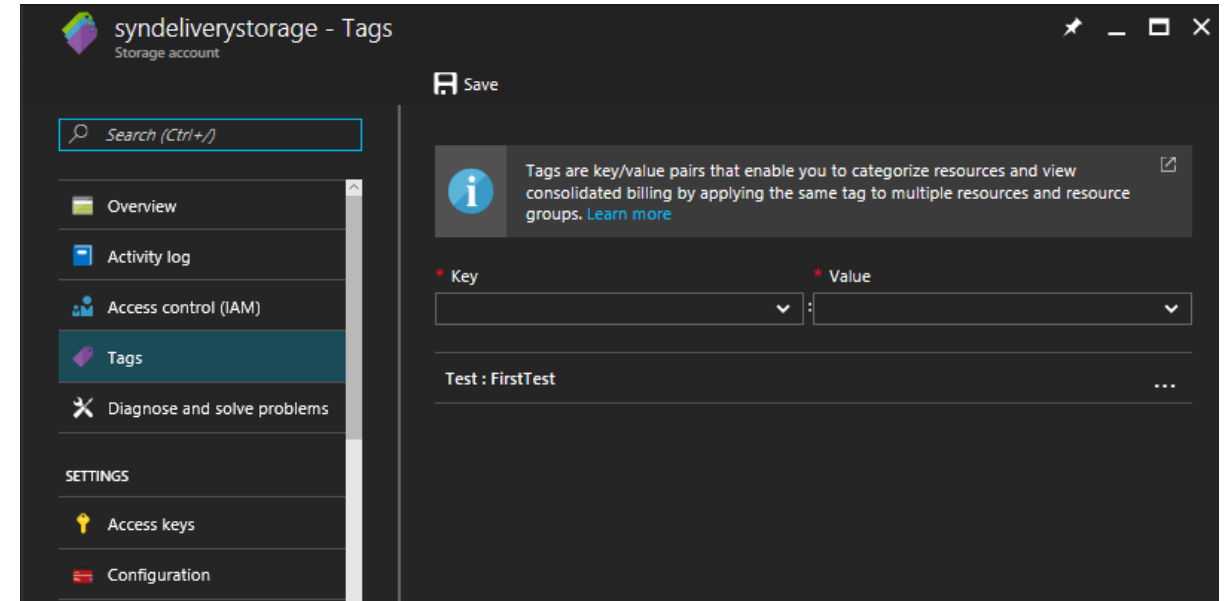
```
"resources": [  
  {  
    "type": "Microsoft.Storage/storageAccounts",  
    "apiVersion": "2015-06-15",  
    "name": "[concat('storage', uniqueString(resourceGroup().id))]",  
    "location": "[resourceGroup().location]",  
    "tags": {  
      "dept": "Finance"  
    },  
    "properties": {  
      "accountType": "Standard_LRS"  
    }  
  }  
]
```

Tag a resource using Portal

- Select the resource group and Click on the Tags option.
- Enter key and value for the tag.



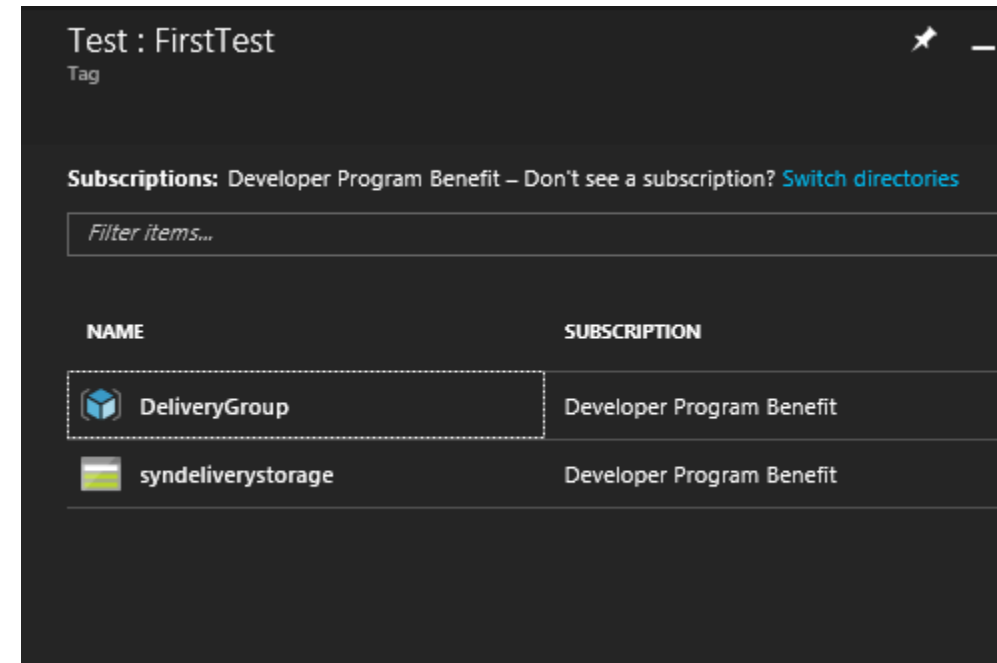
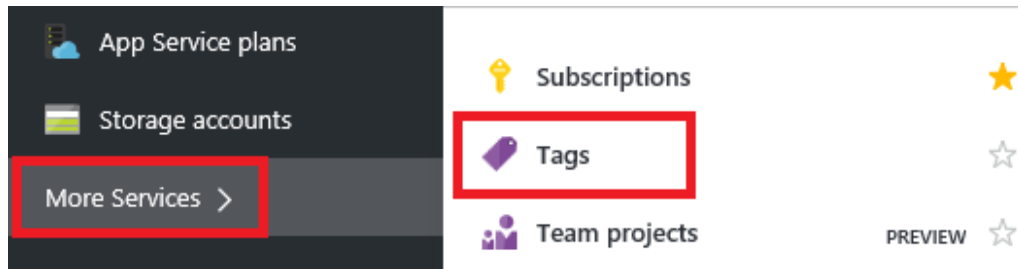
For a Resource Group



For a storage account

Tags using Portal

- To view your taxonomy of tags in the portal, select **More Services** and **Tags**.
- Select any of the tags to display the resources and resource groups with that tag



Tags using PowerShell

Get-Module -ListAvailable -Name AzureRm.Resources | Select Version

Old : For powershell version <3.0.0

```
New-AzureRmResourceGroup -Tags @{ Name = "testtag"; Value = "testval" } `
    -Name DeliveryGroup `
    -Location 'SouthEast Asia'
```

New : For powershell version >3.0.0

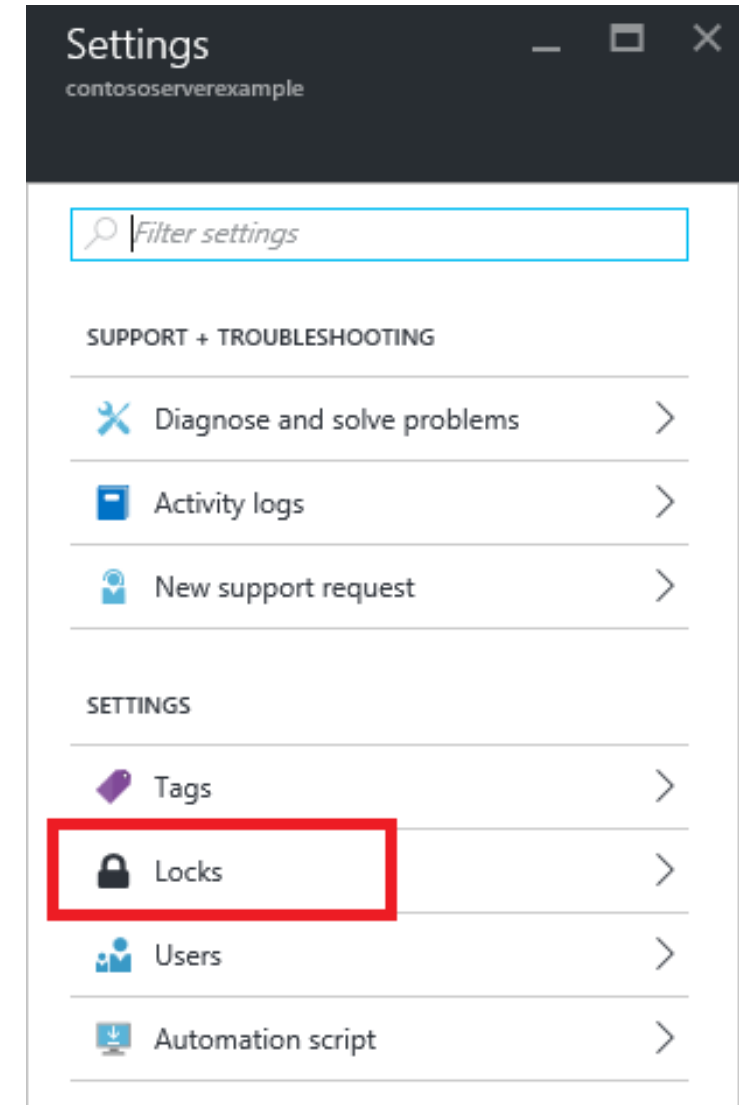
```
New-AzureRmResourceGroup -Tag @{ testtag = "testval" } `
    -Name DeliveryGroup `
    -Location 'SouthEast Asia'
```

Lock resources

- Lock a subscription, resource group, or resource to prevent other users from accidentally deleting or modifying critical resources.
- Set the lock level to **CanNotDelete** or **ReadOnly**.
 - **CanNotDelete** :authorized users can still read and modify a resource, but they can't delete the resource.
 - **ReadOnly** :authorized users can read a resource, but they can't delete or update the resource.

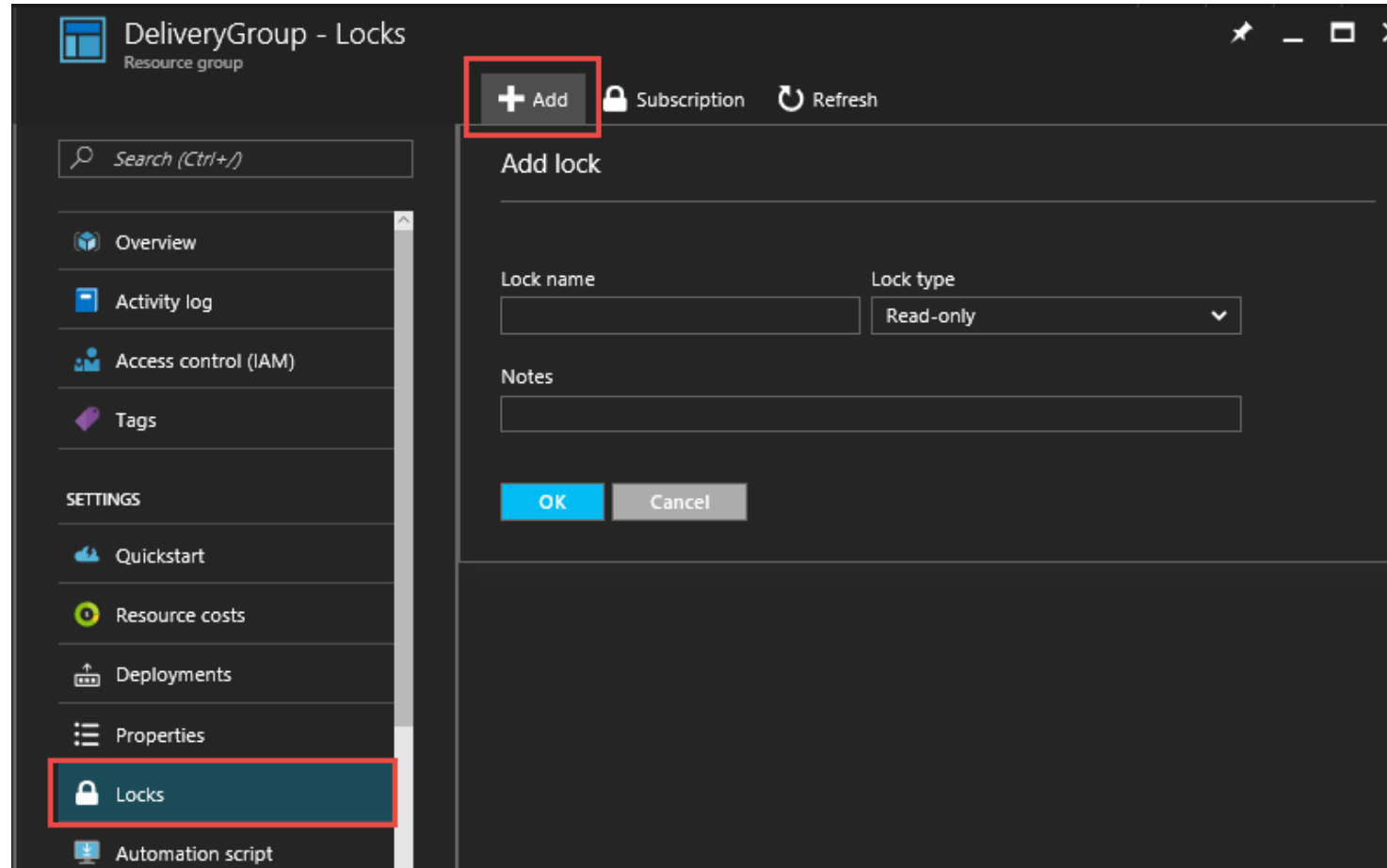
Applying locks using Portal

- In the Settings blade for the resource, resource group, or subscription that you wish to lock, select **Locks**.



Applying locks using Portal

- Click on the Add, specify the name and lock type.



Applying/Removing locks using PowerShell

```
New-AzureRmResourceLock -LockLevel CanNotDelete `
    -LockName LockStorage `
    -ResourceName mystoragename `
    -ResourceType Microsoft.Storage/storageAccounts `
    -ResourceGroupName TestRG1
```

```
Remove-AzureRmResourceLock -LockName LockStorage `
    -ResourceName mystoragename `
    -ResourceType Microsoft.Storage/storageAccounts `
    -ResourceGroupName TestRG1
```

Export templates

- RM enables you to export a RM template from existing resources.
- Exported template can be used for redeployment.
- Two types of templates can be exported
 - You can export the actual template that you used for a deployment.
 - You can export a template that represents the current state of the resource group.

Export template using Portal

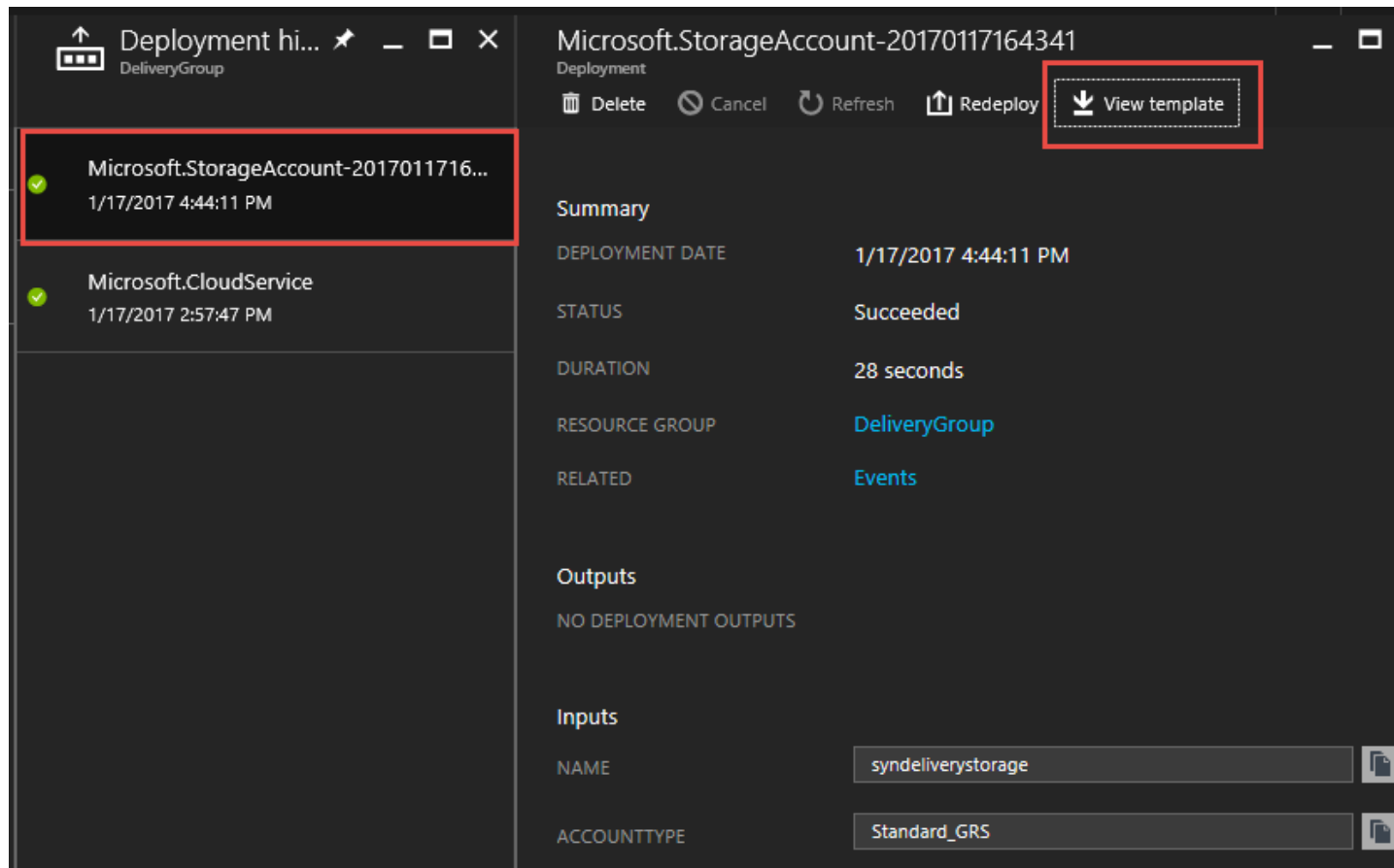
- Navigate to the resource group and click on the Deployments .
- It shows the deployment history.

The screenshot displays the Azure Portal interface. On the left, the 'Essentials' sidebar shows the 'Deployments' link highlighted with a red box, indicating '2 Succeeded'. The main area shows the 'Deployment history' for a resource group, also highlighted with a red box. It lists two successful deployments: 'Microsoft.StorageAccount-2017011716...' at 1/17/2017 4:44:11 PM and 'Microsoft.CloudService' at 1/17/2017 2:57:47 PM. Below this, a table lists the resources: 'syn-cloudsvc' (Cloud service (classic)) and 'syndeliverystorage' (Storage account), both located in Southeast Asia.

NAME	TYPE	LOCATION
syn-cloudsvc	Cloud service (classic)	Southeast Asia
syndeliverystorage	Storage account	Southeast Asia

Export template using Portal

- Select any of the deployment from the history, click on the view template.



The screenshot shows the Azure Portal interface. On the left, the 'Deployment history' pane for the 'DeliveryGroup' is visible, listing two successful deployments. The first deployment, 'Microsoft.StorageAccount-2017011716...', is selected and highlighted with a red box. On the right, the details for this deployment are shown. The 'View template' button in the top right corner of the details pane is also highlighted with a red box. Below the summary, the 'Outputs' section shows 'NO DEPLOYMENT OUTPUTS', and the 'Inputs' section shows 'NAME' as 'syndeliverystorage' and 'ACCOUNTTYPE' as 'Standard_GRS'.

Deployment history (DeliveryGroup)

- Microsoft.StorageAccount-2017011716...
1/17/2017 4:44:11 PM
- Microsoft.CloudService
1/17/2017 2:57:47 PM

Microsoft.StorageAccount-20170117164341
Deployment

Delete Cancel Refresh Redeploy View template

Summary

DEPLOYMENT DATE 1/17/2017 4:44:11 PM

STATUS Succeeded

DURATION 28 seconds

RESOURCE GROUP DeliveryGroup

RELATED Events

Outputs

NO DEPLOYMENT OUTPUTS

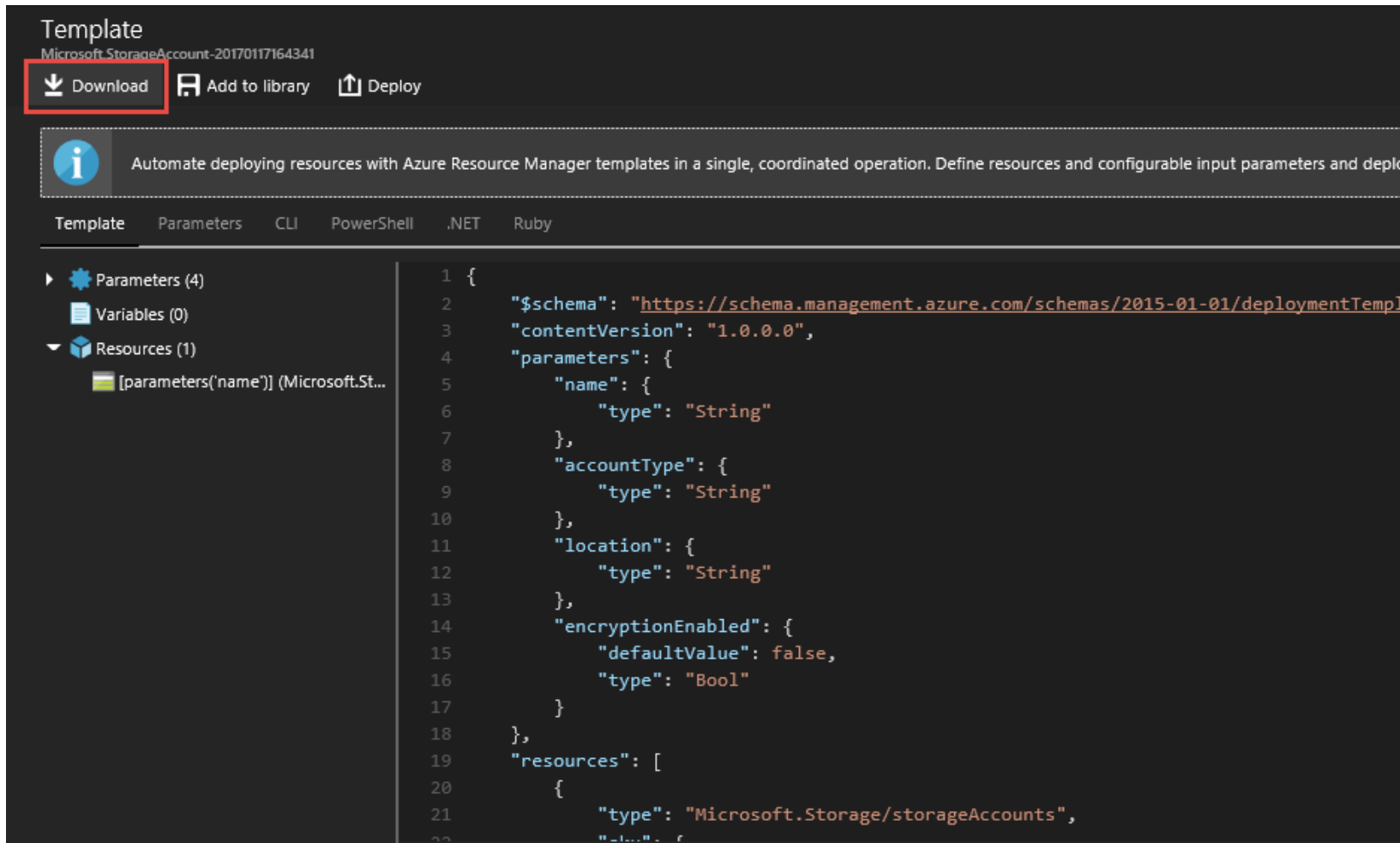
Inputs

NAME syndeliverystorage

ACCOUNTTYPE Standard_GRS

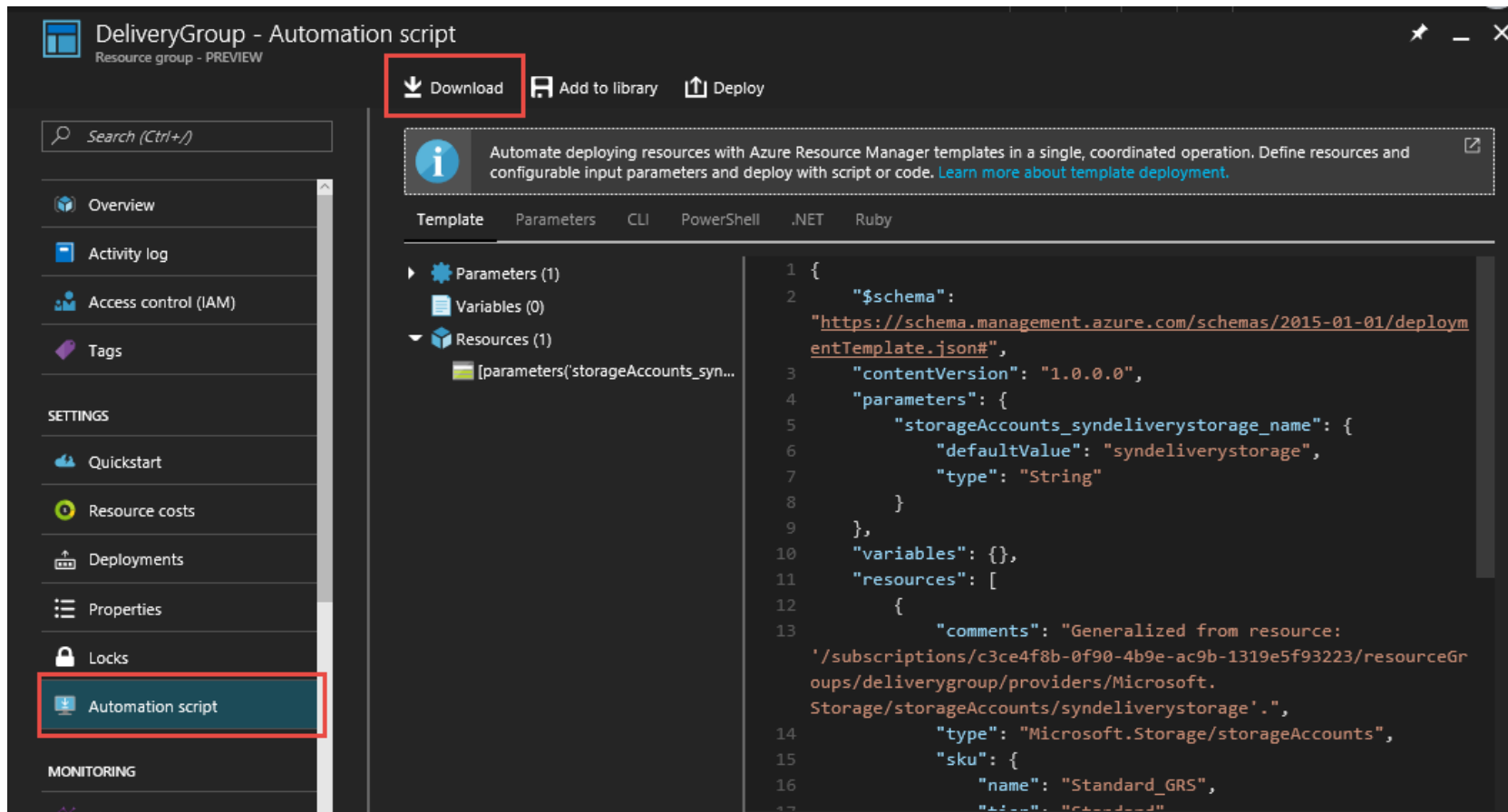
Export template using Portal

- Click the download button to download the template.



Export template for resource group

- Select the Resource group and click on the **Automation script** option.
- Click on the download button to download the template.



The screenshot shows the Azure portal interface for a resource group named 'DeliveryGroup'. The 'Automation script' tab is selected, and the 'Download' button is highlighted with a red box. The left sidebar shows the 'Automation script' option selected. The main area displays the JSON template for a storage account resource.

Automation script

Resource group - PREVIEW

Download Add to library Deploy

Automate deploying resources with Azure Resource Manager templates in a single, coordinated operation. Define resources and configurable input parameters and deploy with script or code. [Learn more about template deployment.](#)

Template Parameters CLI PowerShell .NET Ruby

Parameters (1)
Variables (0)
Resources (1)
[parameters('storageAccounts_syn...]

```
1 {  
2   "$schema":  
3     "https://schema.management.azure.com/schemas/2015-01-01/deploym  
4     entTemplate.json#",  
5   "contentVersion": "1.0.0.0",  
6   "parameters": {  
7     "storageAccounts_syndeliverystorage_name": {  
8       "defaultValue": "syndeliverystorage",  
9       "type": "String"  
10    }  
11  },  
12  "variables": {},  
13  "resources": [  
14    {  
15      "comments": "Generalized from resource:  
16      '/subscriptions/c3ce4f8b-0f90-4b9e-ac9b-1319e5f93223/resourceGr  
17      oups/deliverygroup/providers/Microsoft.  
Storage/storageAccounts/syndeliverystorage'.",  
18      "type": "Microsoft.Storage/storageAccounts",  
19      "sku": {  
20        "name": "Standard_GRS",  
21        "tier": "Standard"
```

Export template using PowerShell

```
Export-AzureRmResourceGroup -ResourceGroupName TestRG1 `
    -Path c:\Azure\Templates\Downloads\TestRG1.json
```


Remove resources or resource group

Removing Resources/Resource group

- You can remove a resource or resource group. When you remove a resource group, you also remove all the resources within that resource group.

Removing Resources/Resource group using PowerShell



Microsoft Partner
Silver Cloud Platform
Silver Learning

```
Remove-AzureRmResource -ResourceName mystoragename `
    -ResourceType Microsoft.Storage/storageAccounts `
    -ResourceGroupName TestRG1
```

```
Remove-AzureRmResourceGroup -Name TestRG1
```

THANK YOU