AZURE FUNDAMENTALS

Control and organize Azure resources with Azure Resource Manager

-Microsoft Learning

Disclaimer: The information contained in this document is completely owned and published by Microsoft Learning. I claim no credits to any of the content as I have just accumulated the available information for ease of read and learning. This document is/will/should not be used for any revenue systems.

Contents

Introduction

Imagine you've joined a company who has been moving to the cloud. This movement happened organically across different departments, and resulted in a lack of awareness of what's already been created and where everything is. There's no ability to easily determine who owns which resources. There's no enforcement of standards for things like resource names, resource sizes, and geographic locations. There's also been several instances where critical resources were inadvertently deleted, causing business-critical outages.

Your manager has asked you to head up an effort to put some order into the chaos, but you're new to Azure and aren't entirely sure what you can do to make this better.

Azure Resource Manager has a number of features that you can use to organize resources, enforce standards, and protect critical Azure resources from accidental deletion. We'll take a tour through these features, and show how you can use them to your advantage.

Learning objectives

In this module, you will:

- Use resource groups to organize Azure resources
- Use tags to organize resources
- Apply policies to enforce standards in your Azure environments
- Use resource locks to protect critical Azure resources from accidental deletion

Note

For this module, you will need to use your own subscription to follow along. We'll be working with resources that will have no cost associated with them, so a trial subscription or a subscription you already have access to will work to follow along with these exercises.

Principles of resource groups

In your first week on your new job, you've looked through the existing resources in your company's Azure subscription. There are a number of resource groups that contain many different resources, but they aren't organized into a coherent structure. You've worked on Azure before, but aren't entirely sure how resource groups work and what their role is. You've guessed (correctly) that they can play a role in how you organize your resources. Let's look at what they are, and how they can be used.

Note

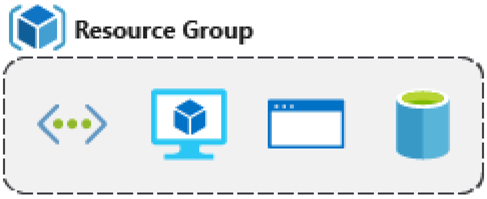
If you don't have an Azure subscription, create a free account before you begin.

What are resource groups?

Resource groups are a fundamental element of the Azure platform. A resource group is a logical container for resources deployed on Azure. These resources are anything you create in an Azure subscription like virtual machines, Application Gateways, and CosmosDB instances. All resources must be in a resource group and a resource can only be a member of a single resource group. Many resources can be moved between resource groups with some services having specific limitations or requirements to move. Resource groups can't be nested. Before any resource can be provisioned, you need a resource group for it to be placed in.

Logical grouping

Resource groups exist to help manage and organize your Azure resources. By placing resources of similar usage, type, or location, you can provide some order and organization to resources you create in Azure. Logical grouping is the aspect that we're most interested in here, since there's a lot of disorder among our resources.



Life cycle

If you delete a resource group, all resources contained within are also deleted. Organizing resources by life cycle can be useful in non-production environments, where you might try an experiment, but then dispose of it when done. Resource groups make it easy to remove a set of resources at once.

Authorization

Resource groups are also a scope for applying role-based access control (RBAC) permissions. By applying RBAC permissions to a resource group, you can ease administration and limit access to allow only what is needed.

Create a Resource Group

Resource groups can be created by using the following methods:

- Azure portal
- Azure PowerShell
- Azure CLI
- Templates
- Azure SDKs (like .NET, Java)

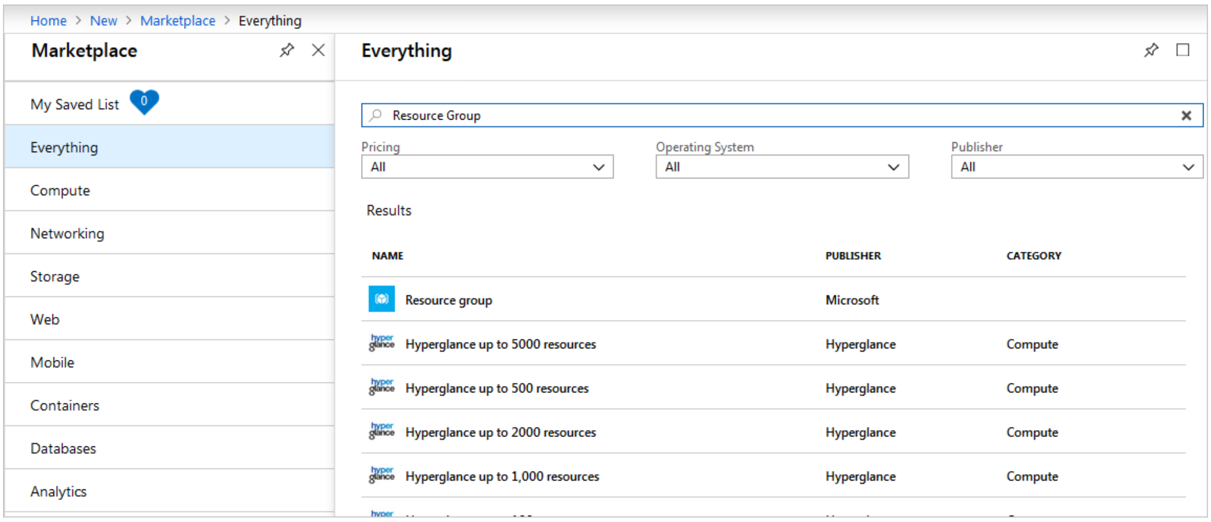
Let's walk through the steps you'd take to create a resource group in the Azure portal. If you'd like to follow along in your own subscription, you may.

1. Open a web browser and sign into the Azure portal .

   Important

   Make sure to use your *own* subscription. When you are in the free sandbox environment, it will not allow you to create resource groups. You can tell which subscription you are using by looking at the tenant name under your profile picture. You can switch tenants by clicking on your profile picture and selecting **Switch Directory** from the options menu.

2. On the left pane, select **+ Create a resource**
3. Type **Resource Group** in the search box and hit Enter.
4. The first item in the list should be the resource group resource. Select it and then click the **Create** button.

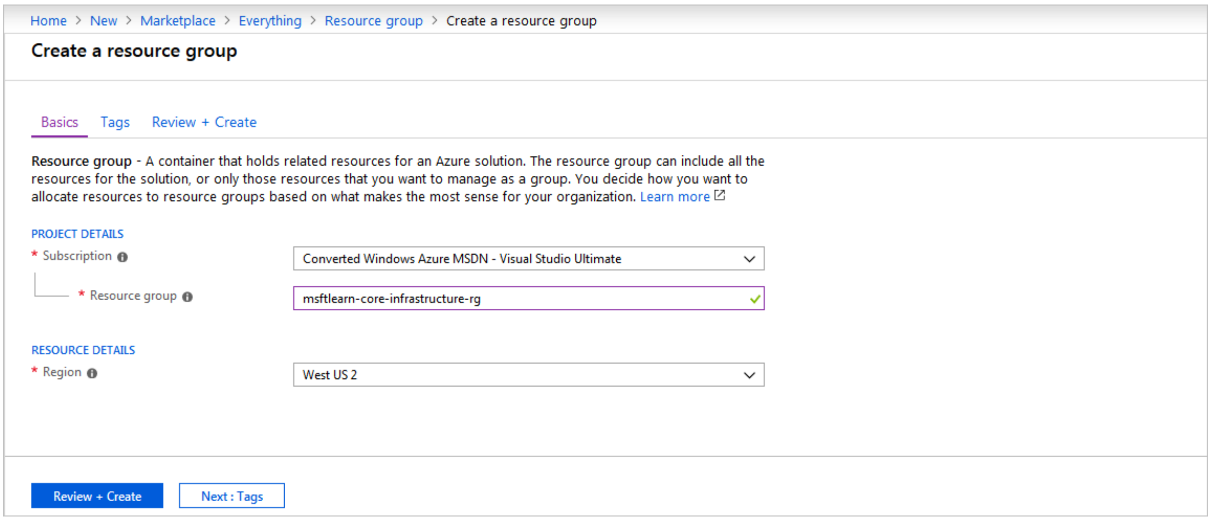


Resource Group

Pricing: All
Operating System: All
Publisher: All

NAME	PUBLISHER	CATEGORY
Resource group	Microsoft	
Hyperglance up to 5000 resources	Hyperglance	Compute
Hyperglance up to 500 resources	Hyperglance	Compute
Hyperglance up to 2000 resources	Hyperglance	Compute
Hyperglance up to 1,000 resources	Hyperglance	Compute

5. Enter your resource group name, let's use **msftlearn-core-infrastructure-rg**. Select the subscription it should be in, and select the region for the resource group. Click **Review + Create** and then **Create** to create the resource group.



Create a resource group

Resource group - A container that holds related resources for an Azure solution. The resource group can include all the resources for the solution, or only those resources that you want to manage as a group. You decide how you want to allocate resources to resource groups based on what makes the most sense for your organization.

* Subscription: Converted Windows Azure MSDN - Visual Studio Ultimate

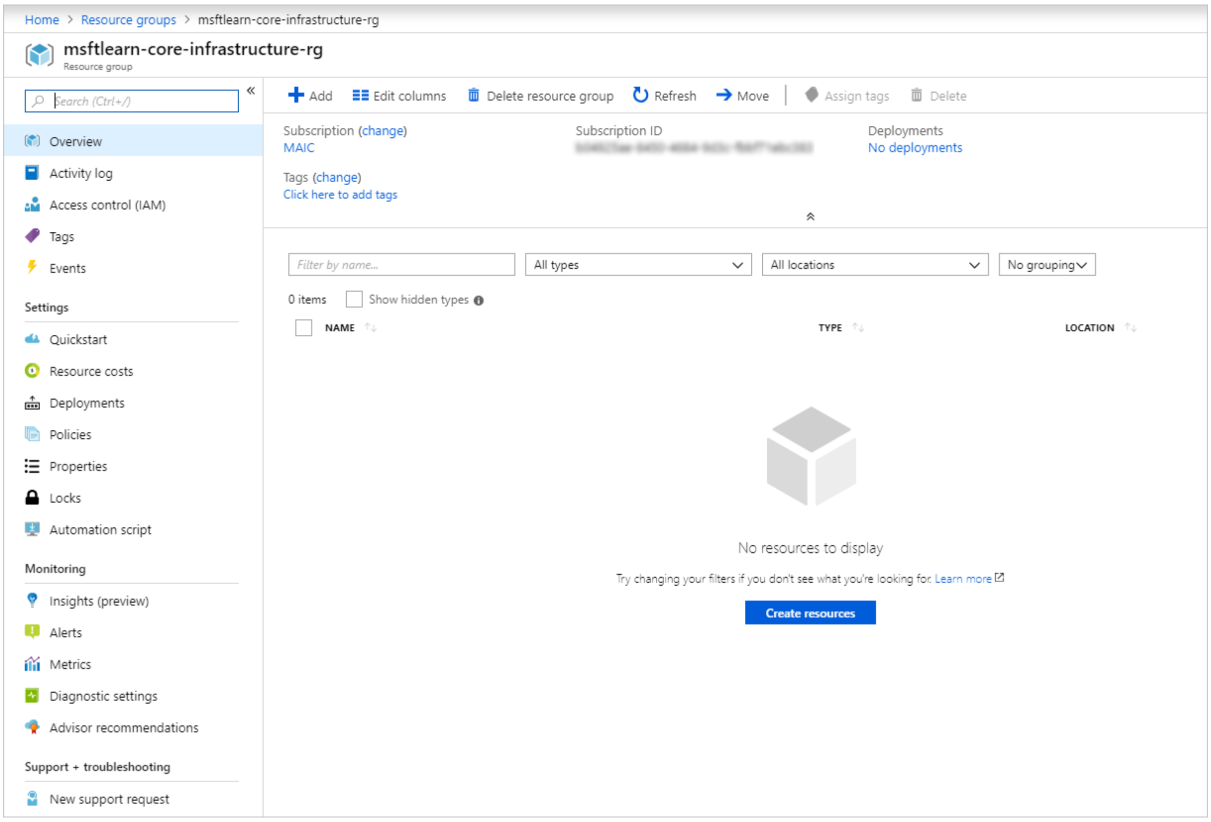
* Resource group: msftlearn-core-infrastructure-rg

* Region: West US 2

That's it, you've created a resource group that you can now use when you deploy Azure resources. Let's take a closer look at this resource group and some important things to consider.

Explore a resource group and add a resource

In the portal, select **Resource groups** on the left menu, and select your newly created resource group. Note that you may also see a resource group called **NetworkWatcherRG**. You can ignore this resource group, it's created automatically to enable Network Watcher in Azure virtual networks.



msftlearn-core-infrastructure-rg
Resource group

Subscription (change): MAIC

Subscription ID: [redacted]

Deployments: No deployments

Tags (change): Click here to add tags

0 items □ Show hidden types

No resources to display

Try changing your filters if you don't see what you're looking for.

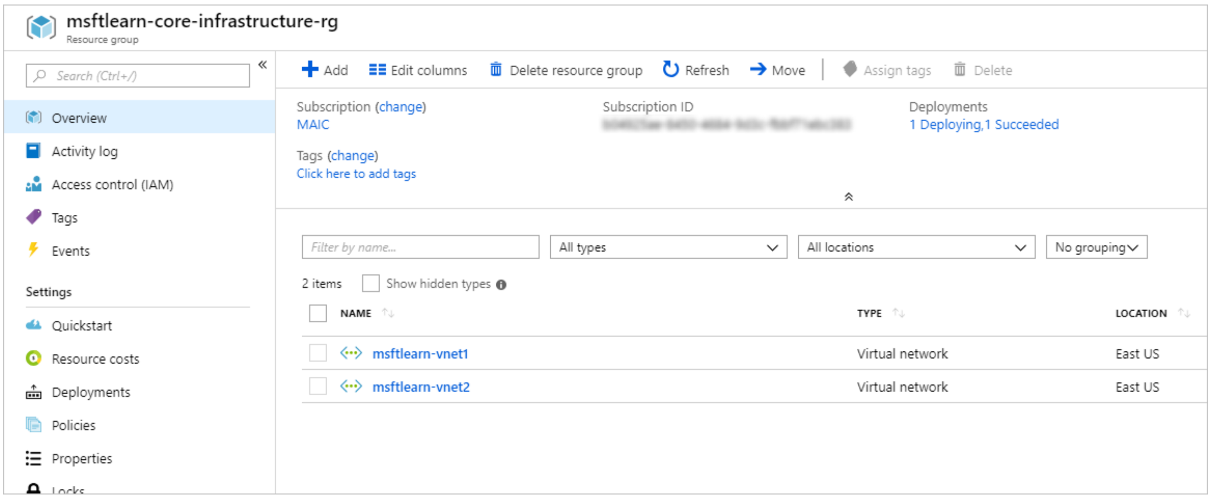
On the Overview panel, there's the basic information about the resource group like the subscription it's in, the subscription ID, any tags that are applied, and a history of the deployments to this resource group. We'll cover tags in the next unit. The deployments link takes you to a new panel with the history of all deployments to this resource group. Anytime you create a resource, it's a deployment, and you see that history for the resource group here.

Across the top you can add more resources, change the columns in the list, move the resource group to another subscription, or delete it entirely.

On the left menu, there are a number of options

We don't have any resources in this resource group yet, so the list at the bottom is empty. Let's create a couple resources inside the resource group.

1. Click **+ Add** at the top or click the **Create resources**, either will work.
2. Search for **Virtual network**. The first result should be the virtual network resource. Click it, and on the next screen, make sure **Select a deployment model** is set to **Resource Manager**. Click **Create**.
3. Name the virtual network **msftlearn-vnet1**. For the **Resource group** drop-down, select the resource group that you created earlier. Enter **192.168.0.0/24** for both the **Address space** and subnet **Address range**. Leave the defaults for all other options, and click **Create**.
4. Repeat the steps again to create one more VNet, where both the **Address space** and subnet **Address range** are for a different network than your previous network, (e.g. **192.168.100.0/24**). Name it **msftlearn-vnet2**, and make sure to place it in the resource group that you created earlier.
5. Go back to your resource group, and on the **Overview** panel you should see the two VNets you created.



Subscription (change)
MAIC

Subscription ID
[redacted]

Deployments
1 Deploying,1 Succeeded

Tags (change)
Click here to add tags

2 items

NAME	TYPE	LOCATION
msftlearn-vnet1	Virtual network	East US
msftlearn-vnet2	Virtual network	East US

Our resource group now contains two virtual network resources because we specified in our deployment (when we created the resources) which resource group we wanted the VNet to be placed in. We could create additional resources inside this resource group, or we could create additional resource groups in the subscription to deploy resources into.

When creating resources, you usually have the option to create a new resource group as an alternative to using an existing resource group. This simplifies the process a bit, but as you see in your new organization, can lead to resources spread across resource groups with little thought as to how to organize them.

Use resource groups for organization

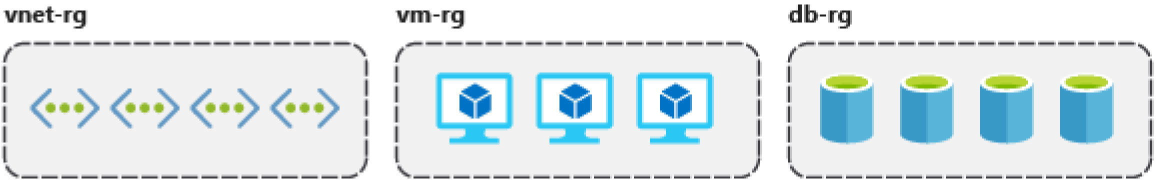
So how can you use resource groups to your advantage in your new organization? There are some guidelines and best practices that can help with the organization.

Consistent naming convention

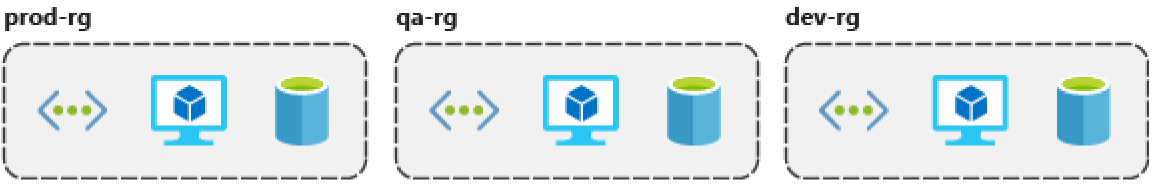
You can start with using an understandable naming convention. We named our resource group **msftlearn-core-infrastructure-rg**. We've given some indication of what it's used for (**msftlearn**), the types of resources contained within (**core-infrastructure**), and the type of resource it is itself (**rg**). This descriptive name gives us a better idea of what it is. If we had named it **my-resource-group** or **rg1**, we have no idea on a glance of what the usage may be. In this case, we can deduce that there are probably core pieces of infrastructure contained within. If we created additional VNets, storage accounts, or other resources the company may consider *core infrastructure*, we could place them here as well, to improve the organization of our resources. Naming conventions can vary widely between and even within companies, but some planning can help.

Organizing principles

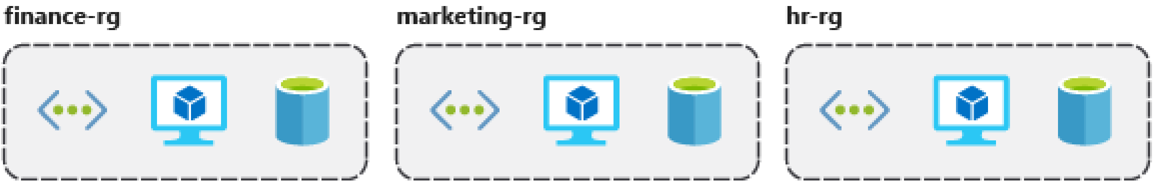
Resource groups can be organized in a number of ways, let's take a look at a few examples. We might put all resources that are *core infrastructure* into this resource group. But we could also organize them strictly by resource type. For example, put all VNets in one resource group, all virtual machines in another resource group, and all Azure Cosmos DB instances in yet another resource group.



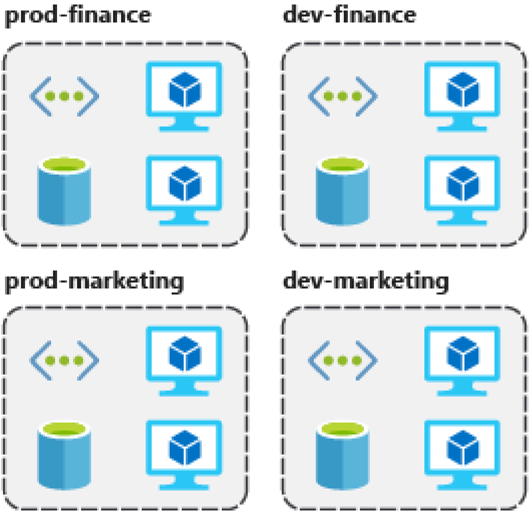
We could organize them by environment (prod, qa, dev). In this case, all production resources are in one resource group, all test resources are in another resource group, and so on.



We could organize them by department (marketing, finance, human resources). Marketing resources go in one resource group, finance in another resource group, and HR in a third resource group.



We could even use a combination of these strategies and organize by environment and department. Put production finance resources in one resource group, dev finance resources in another, and the same for the marketing resources.



There are a few factors that can play into the strategy you use to organize resources: authorization, resource life cycle, and billing.

Organizing for authorization

Since resource groups are a scope of RBAC, you can organize resources by *who* needs to administer them. If your database administration team is responsible for managing all of your Azure SQL Database instances, putting them in the same resource group would simplify administration. You could give them the proper permissions at the resource group level to administer the databases within the resource group. Similarly, the database administration team could be denied access to the resource group with virtual networks, so they don't inadvertently make changes to resources outside the scope of their responsibility.

Organizing for life cycle

We mentioned earlier that resource groups serve as the life cycle for the resources within it. If you delete a resource group, you delete all the resources in it. Use this to your advantage, especially in areas where resources are more disposable, like non-production environments. If you deploy 10 servers for a project that you know will only last a couple of months, you might put them all in a single resource group. One resource group is easier to clean up than 10 or more resource groups.

Organizing for billing

Lastly, placing resources in the same resource group is a way to group them for usage in billing reports. If you're trying to understand how your costs are distributed in your Azure environment, grouping them by resource group is one way to filter and sort the data to better understand where costs are allocated.

Summary

The bottom line is that you have flexibility in how to organize resources in your resource groups. Put some thought into it so that you have a coherent approach to how you use resource groups in your Azure environment.

Use tagging to organize resources

You've gone through your resources and moved them into resource groups that are more organized than before. But what if resources have multiple uses? How do you better search, filter, and organize these resources? Tags can be helpful as you look to improve organization of your Azure resources.

What are tags?

Tags are name/value pairs of text data that you can apply to resources and resource groups. Tags allow you to associate custom details about your resource, in addition to the standard Azure properties a resource has:

- department (like finance, marketing, and more)
- environment (prod, test, dev),
- cost center
- life cycle and automation (like shutdown and startup of virtual machines).

A resource can have up to 15 tags. The name is limited to 512 characters for all types of resources except storage accounts, which have a limit of 128 characters. The tag value is limited to 256 characters for all types of resources. Tags aren't inherited from parent resources. Not all resource types support tags, and tags can't be applied to classic resources.

Tags can be added and manipulated through the Azure portal, Azure CLI, Azure PowerShell, Resource Manager templates, and through the REST API. For example, to add a resource tag to a virtual network using the Azure CLI, you could use the following command:

Azure CLI

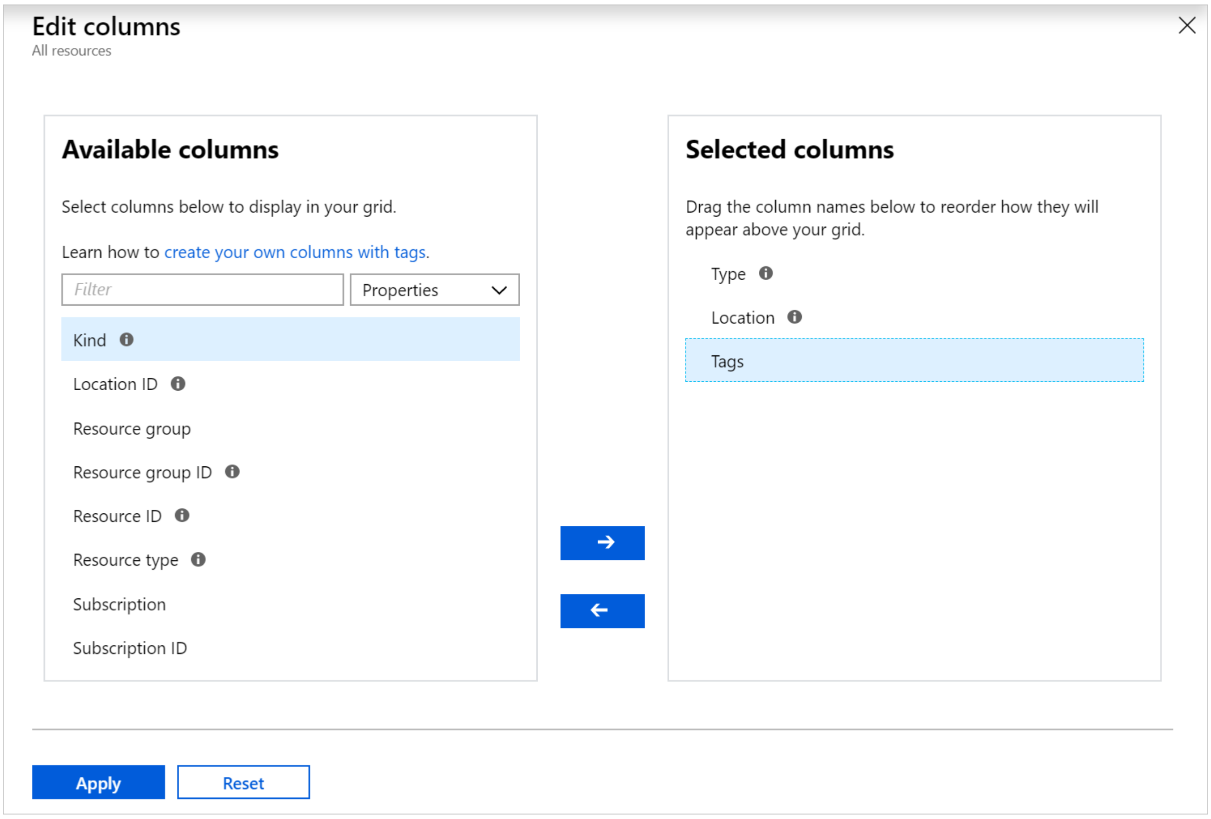
```
az resource tag --tags Department=Finance \
    --resource-group msftlearn-core-infrastructure-rg \
    --name msftlearn-vnet1 \
    --resource-type "Microsoft.Network/virtualNetworks"
```

You can use Azure Policy to automatically add or enforce tags for resources your organization creates based on policy conditions that you define. For example, you could require that a value for the Department tag is entered when someone in your organization creates a virtual network in a specific resource group.

Apply tags to resources

Let's apply some tags to the resources you created. Recall that we created a resource group **msftlearn-core-infrastructure-rg** and two VNets inside that resource group, **msftlearn-vnet1** and **msftlearn-vnet2**. The names of the VNets are relatively generic, so we'd like to associate the VNets with services from different departments.

1. Open the Azure portal , and navigate to your **msftlearn-core-infrastructure-rg** resource group.
2. On the **Overview** tab of your resource group, you should see your two VNets listed. The default view doesn't display the tags column, so let's add that to the display. Select **Edit columns** at the top. In the **Available columns** list, select **Tags** and click **->** to add it to the **Selected columns** list. Click **Apply** to apply your changes.



Edit columns
All resources

Available columns

Select columns below to display in your grid.

Learn how to create your own columns with tags.

Filter | Properties

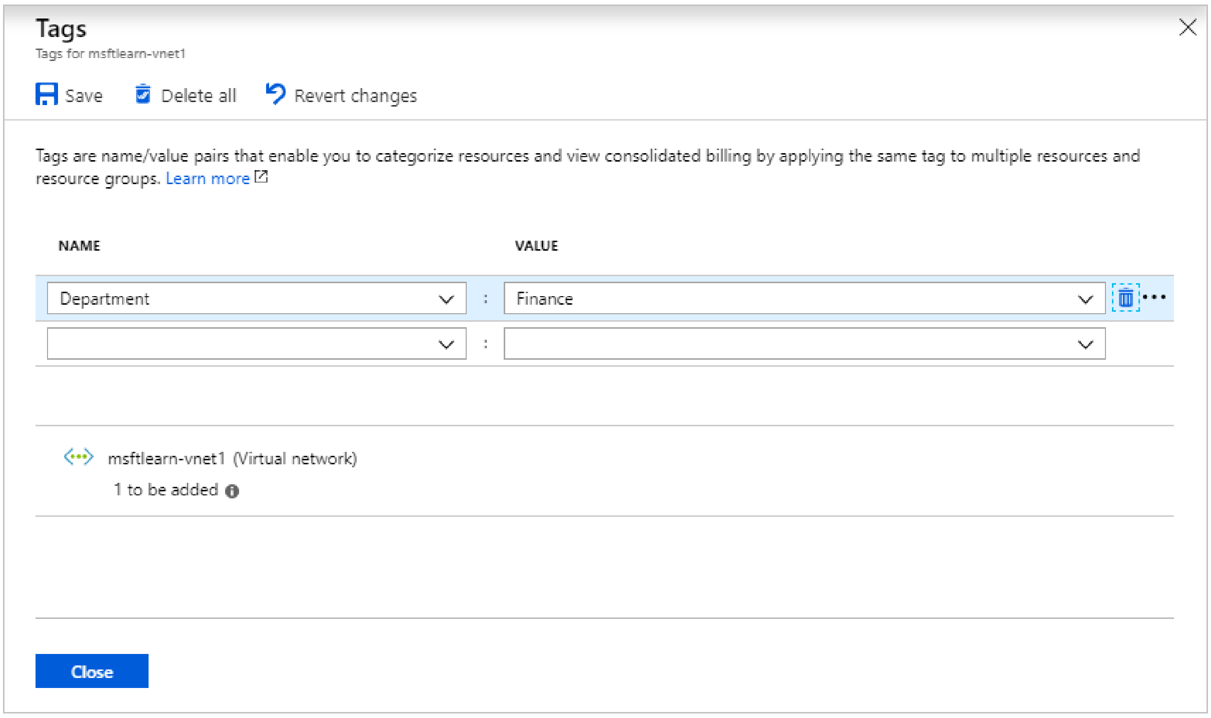
- Kind
- Location ID
- Resource group
- Resource group ID
- Resource ID
- Resource type
- Subscription
- Subscription ID

Selected columns

Drag the column names below to reorder how they will appear above your grid.

- Type
- Location
- Tags

3. You should now see the tags column, but it will be empty since we haven't added any tags yet. We'll add the tags directly here. You can also add tags to any resource that supports it on the resource's **Tags** panel. In the list of resources, you should see a pencil in the **TAGS** column where you can directly edit the tags. Click the pencil for the **msftlearn-vnet1** resource.
4. This will display the dialog to edit the Tags. Let's add a couple tags to this VNet. In the **NAME** box type **Department**, and in the **VALUE** box type **Finance**. Click **Save** to save your changes, then click **Close** to close the dialog.



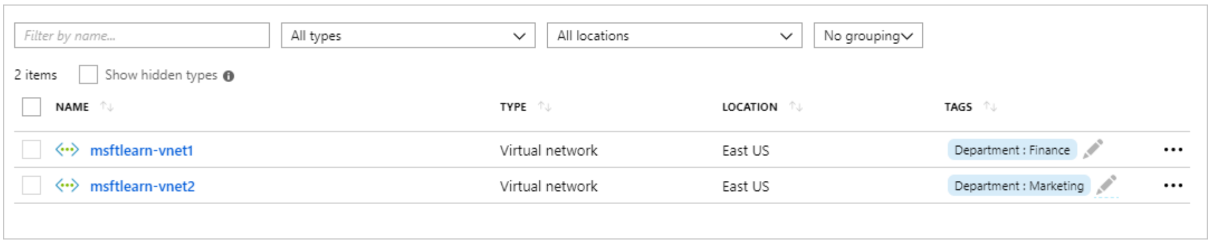
Tags are name/value pairs that enable you to categorize resources and view consolidated billing by applying the same tag to multiple resources and resource groups. Learn more

NAME		VALUE
Department	:	Finance
	:	

msftlearn-vnet1 (Virtual network)
1 to be added

5. Let's do the same steps for the **msftlearn-vnet2** VNet. For this VNet, add a **Department:Marketing** tag to the resource.

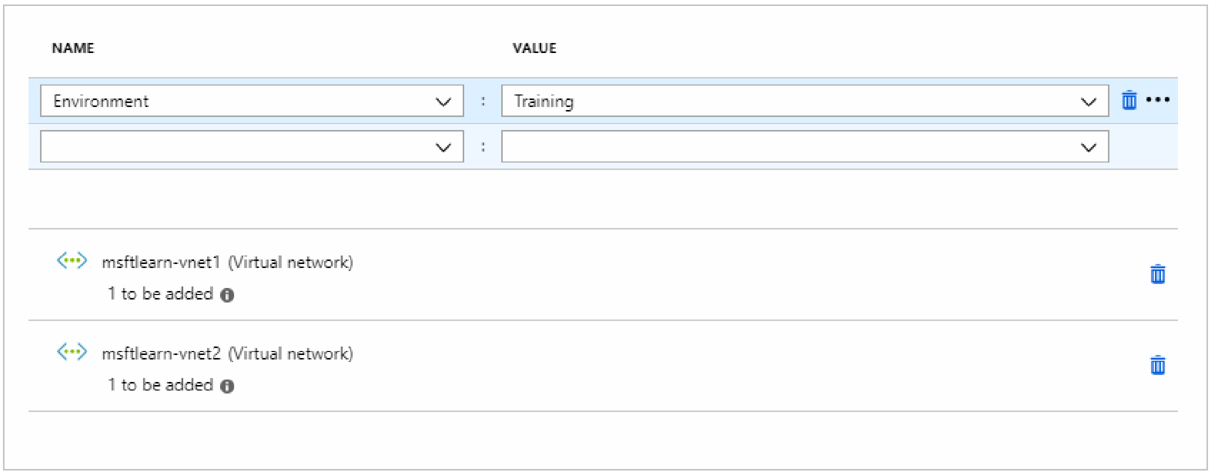
   You should now see your tags applied to each resource.



NAME	TYPE	LOCATION	TAGS
msftlearn-vnet1	Virtual network	East US	Department : Finance
msftlearn-vnet2	Virtual network	East US	Department : Marketing

6. Let's add tags to both of these resources in bulk. Select the checkbox on the left for each of the VNets and click **Assign tags** in the top menu. By selecting multiple resources, we can add a tag to them in bulk, making it easy if we have multiple resources we want to apply the same tag to.

   Add the **Environment:Training** tag to the resources. You should see in the dialog that the tag will be applied to each of the VNets.



NAME	VALUE
Environment	Training

msftlearn-vnet1 (Virtual network)
1 to be added

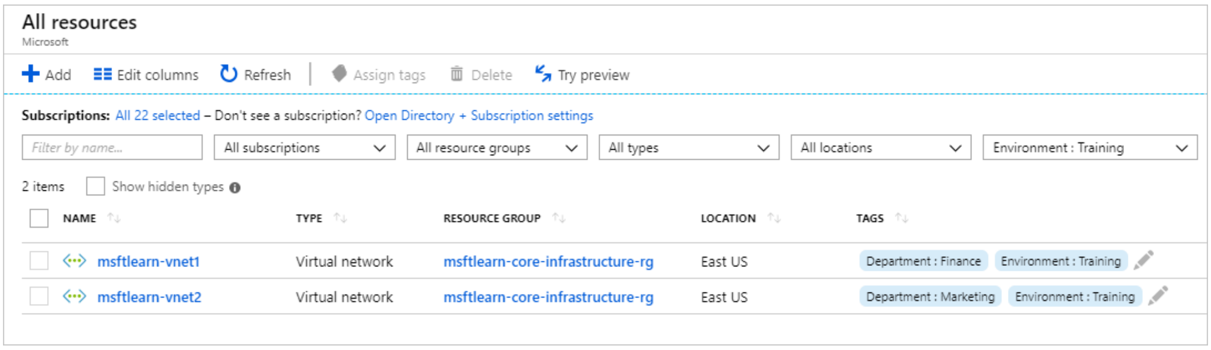
msftlearn-vnet2 (Virtual network)
1 to be added

Back in the resource list you'll now see a **2** displayed as we now have two tags applied to each resource.

7. Let's take a look at how we can use tags to filter your resources. On the main Azure menu on the left, select **All resources**.
8. In the **All tags** drop down, under **Environment** select **Training**. You should see only your two VNets displayed, since we tagged those resources with the **Environment:Training** tag.

   Note that there is currently a preview of the filtering capabilities that you may see in your portal. If you are in this preview, you will need to instead select **Add filter**. In the **Tags**, select **Environment**, then select **Training**.

   You can also join or leave this preview at any time by selecting **Try preview** or **Leave preview**.



All resources
Microsoft

Subscriptions: All 22 selected – Don't see a subscription? Open Directory + Subscription settings

Filter by name... | All subscriptions | All resource groups | All types | All locations | Environment : Training

2 items ☐ Show hidden types

NAME	TYPE	RESOURCE GROUP	LOCATION	TAGS
msftlearn-vnet1	Virtual network	msftlearn-core-infrastructure-rg	East US	Department : Finance Environment : Training
msftlearn-vnet2	Virtual network	msftlearn-core-infrastructure-rg	East US	Department : Marketing Environment : Training

9. We can further filter these resources by additionally filtering on the **Department:Finance** or **Department:Marketing** tags.

Use tags for organization

The above example is just one example of where you can use tags to organize your resources. With their flexibility, there are several ways you can use tags to your advantage.

You can use tags to group your billing data. For example, if you're running multiple VMs for different organizations, use the tags to group usage by cost center. You can also use tags to categorize costs by runtime environment, such as the billing usage for VMs running in the production environment. When exporting billing data or accessing it through billing APIs, tags are included in that data and can be used to further slice your data from a cost perspective.

You can retrieve all the resources in your subscription with a specific tag name or value. Tags enable you to retrieve related resources from different resource groups. This approach is helpful when you need to organize resources for billing or management.

Tagging resources can also help in monitoring to track down impacted resources. Monitoring systems could include tag data with alerts, giving you the ability to know exactly who is impacted. In our example above, we applied the **Department:Finance** tag to the **msftlearn-vnet1** resource. If an alarm was thrown on **msftlearn-vnet1** and the alarm included the tag, we'd know that the finance department may be impacted by the condition that triggered the alarm. This contextual information can be valuable if an issue occurs.

It's also common for tags to be used in automation. If you want to automate the shutdown and startup of virtual machines in development environments during off-hours to save costs, you can use tags to assist in this. Add a **shutdown:6PM** and **startup:7AM** tag to the virtual machines, then create an automation job that looks for these tags, and shuts them down or starts them up based on the tag value. There are several solutions in the Azure Automation Runbooks Gallery that use tags in a similar manner to accomplish this.

Use policies to enforce standards

You're organizing your resources better in resource groups, and you've applied tags to your resources to use them in billing reports and in your monitoring solution. Resource grouping and tagging have made a difference in the existing resources, but how do you ensure that new resources follow the rules? Let's take a look at how policies can help you enforce standards in your Azure environment.

What is Azure Policy?

Azure Policy is a service you can use to create, assign, and manage policies. These policies apply and enforce rules that your resources need to follow. These policies can enforce these rules when resources are created, and can be evaluated against existing resources to give visibility into compliance.

Policies can enforce things such as only allowing specific types of resources to be created, or only allowing resources in specific Azure regions. You can enforce naming conventions across your Azure environment. You can also enforce that specific tags are applied to resources. Let's take a look at how policies work.

Create a policy

We'd like to ensure that all resources have the **Department** tag associated with them and block creation if it doesn't exist. We'll need to create a new policy definition and then assign it to a scope; in this case the scope will be our **mslearn-core-infrastructure-rg** resource group. Policies can be created and assigned through the Azure portal, Azure PowerShell, or Azure CLI. Let's walk through how to do create a policy in the portal.

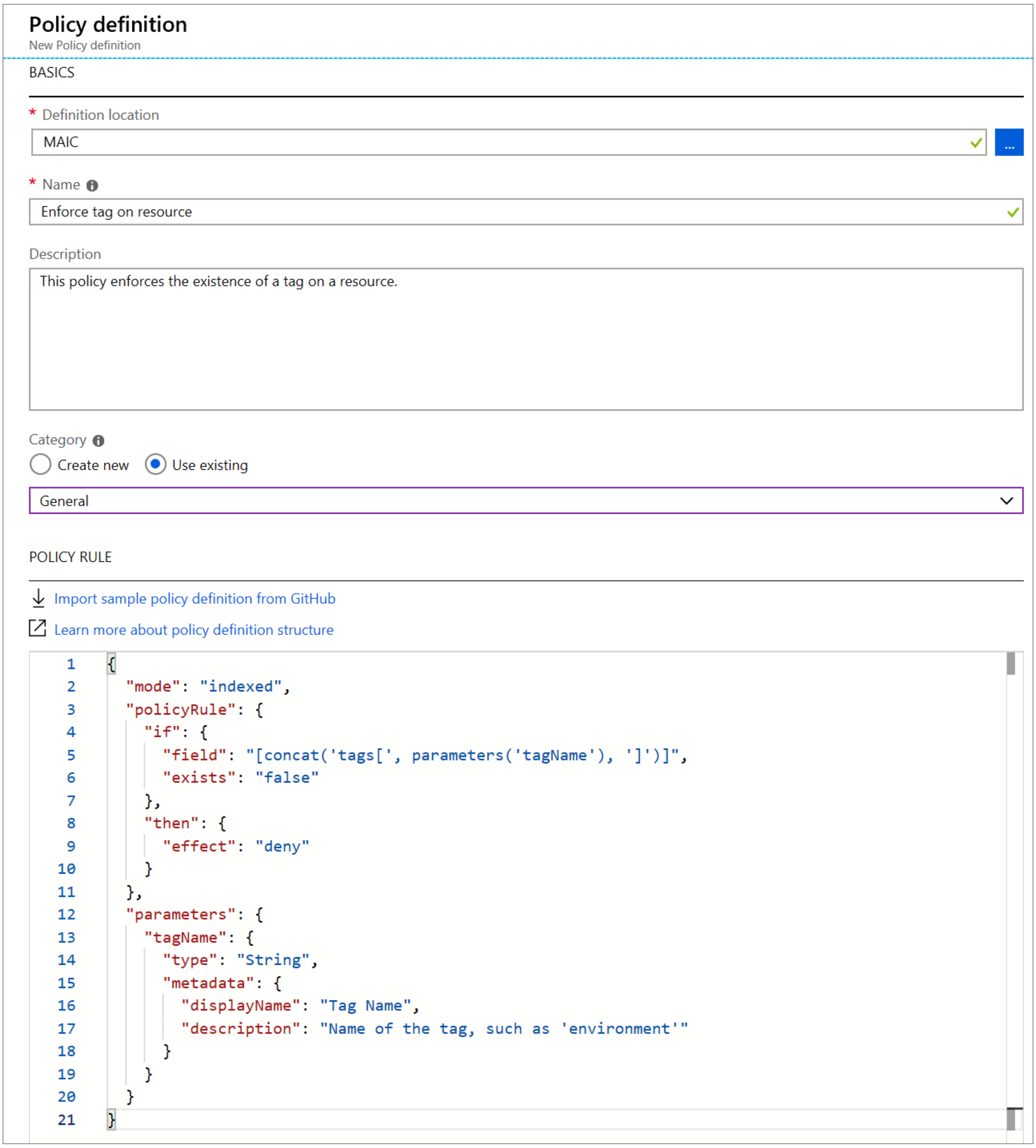
Create the policy definition

1. Go ahead and pull up the Azure portal in a web browser if you haven't already. In the search box in the top navigation bar, search for **Policy** and select the **Policy** service.
2. In **Authoring** section in the left menu, select **Definitions**.
3. You should see a list of built-in policies that you can use. In this case, we're going to create our own custom policy. Click **+ Policy definition** in the top menu.
4. This brings up the **New policy definition** dialog. To set the **Definition location**, click the blue **...**. Select the subscription for the policy to be stored in, which should be the same subscription as our resource group. Click **Select**.
5. Back on the **New policy definition** dialog, for **Name** give your policy a name of **Enforce tag on resource**.
6. For the **Description**, enter This policy enforces the existence of a tag on a resource.
7. For **Category** select **Use existing** and then select the **General** category.
8. For the **Policy rule**, delete all text in the box and paste in the following JSON.

   JSON

```
{
  "mode": "Indexed",
  "policyRule": {
    "if": {
      "field": "[concat('tags[', parameters('tagName'), ']')]",
      "exists": "false"
    },
    "then": {
      "effect": "deny"
    }
  },
  "parameters": {
    "tagName": {
      "type": "String",
      "metadata": {
        "displayName": "Tag Name",
        "description": "Name of the tag, such as 'environment'"
      }
    }
  }
}
```

29. Your policy definition should look like below. Click **Save** to save your policy definition.



Policy definition
New Policy definition

BASICS

* Definition location: MAIC

* Name: Enforce tag on resource

Description: This policy enforces the existence of a tag on a resource.

Category: ○ Create new ● Use existing

General

POLICY RULE

Import sample policy definition from GitHub
Learn more about policy definition structure

```
{
  "mode": "indexed",
  "policyRule": {
    "if": {
      "field": "[concat('tags[', parameters('tagName'), ']')]",
      "exists": "false"
    },
    "then": {
      "effect": "deny"
    }
  },
  "parameters": {
    "tagName": {
      "type": "String",
      "metadata": {
        "displayName": "Tag Name",
        "description": "Name of the tag, such as 'environment'"
      }
    }
  }
}
```

Create a policy assignment

We've created the policy, but we haven't actually put it into effect yet. To enable the policy, we need to create an assignment. In this case, we'll assign it to the scope of our **msftlearn-core-infrastructure-rg** resource group, so that it applies to anything inside the resource group.

1. In the policy pane, in the **Authoring** section on the left, select **Assignments**.
2. Select **Assign policy** at the top.

3. In the **Assign policy** pane, we'll assign our policy to our resource group. For **Scope**, click the blue **...**. Select your subscription and the **msftlearn-core-infrastructure-rg** resource group, then click **Select**.
4. For **Policy definition**, click the blue **...**. In the **Type** drop-down, select **Custom**, select the **Enforce tag on resource** policy you created, then click **Select**.
5. In the **Parameters** section, for **Tag name** enter **Department**. Click **Assign** to assign the policy.

Test out the policy

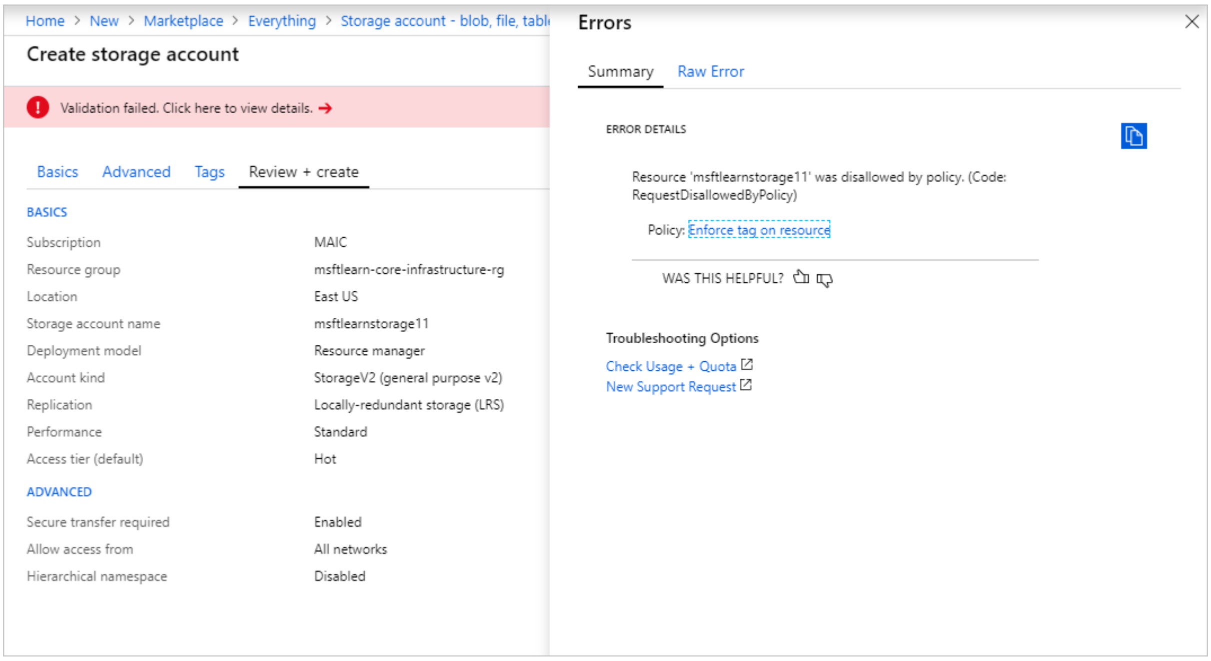
Now that we have assigned the policy to our resource group, any attempts to create a resource without the **Department** tag should fail. Let's try this out.

Important

Please note that the policy assignment may take up to 30 minutes to take effect. Because of this delay, in the following steps the policy validation may succeed but the deployment will still fail. If this happens, allow for additional time and retry your deployment.

1. Click **+ Create a resource** in the top left of the portal.
2. Search for **Storage Account** and select **Storage account** in the results. Click **Create**.
3. Select your subscription, and the **msftlearn-core-infrastructure-rg** resource group.
4. For **Storage account name**, give it any name of your choice, but note that it does have to be a globally unique name.
5. Leave the rest of the options at their default, click **Review + create**.

   Validation of your resource creation will fail because we don't have a **Department** tag applied to the resource.



Home > New > Marketplace > Everything > Storage account - blob, file, tabl

Create storage account

Validation failed. Click here to view details. →

Basics Advanced Tags Review + create

BASICS

Subscription	MAIC
Resource group	msftlearn-core-infrastructure-rg
Location	East US
Storage account name	msftlearnstorage11
Deployment model	Resource manager
Account kind	StorageV2 (general purpose v2)
Replication	Locally-redundant storage (LRS)
Performance	Standard
Access tier (default)	Hot

ADVANCED

Secure transfer required	Enabled
Allow access from	All networks
Hierarchical namespace	Disabled

Errors

ERROR DETAILS

Resource 'msftlearnstorage11' was disallowed by policy. (Code: RequestDisallowedByPolicy)

Policy: Enforce tag on resource

WAS THIS HELPFUL?

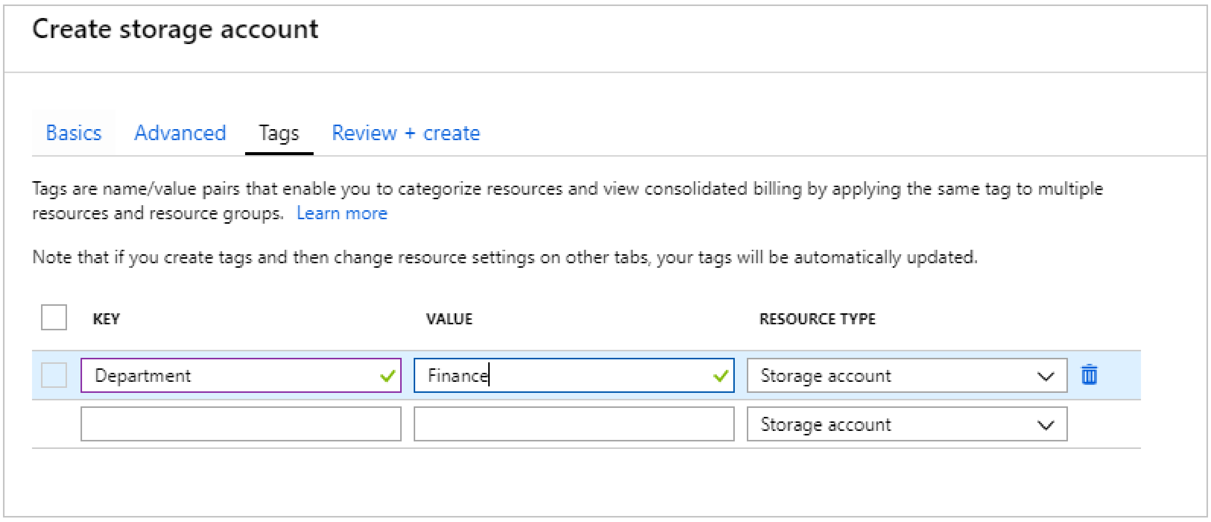
Troubleshooting Options

Check Usage + Quota

New Support Request

Let's fix the violation so we can successfully deploy the storage account.

6. Select **Tags** at the top of the **Create storage account** pane.
7. Add a **Department:Finance** tag to the list.



Create storage account

Tags are name/value pairs that enable you to categorize resources and view consolidated billing by applying the same tag to multiple resources and resource groups. Learn more

Note that if you create tags and then change resource settings on other tabs, your tags will be automatically updated.

□	KEY	VALUE	RESOURCE TYPE
□	Department ✓	Finance ✓	Storage account
			Storage account

8. Now click **Review + create**. Validation should now pass, and if you click **Create** your storage account will be created.

Use policies to enforce standards

We've seen how we could use policies to ensure that our resources have the tags that organize our resources. There are other ways policies can be used to our benefit.

We could use policy to restrict which Azure regions we can deploy resources to. For organizations that are heavily regulated or have legal or regulatory restrictions on where data can reside, policies help to ensure that resources aren't provisioned in geographic areas that would go against these requirements.

We could use policy to restrict which types of virtual machine sizes can be deployed. You may want to allow large VM sizes in your production subscriptions, but maybe you'd like to ensure that you keep costs minimized in your dev subscriptions. By denying the large VM sizes through policy in your dev subscriptions, you can ensure they don't get deployed in these environments.

We could also use policy to enforce naming conventions. If our organization has standardized on specific naming conventions, using policy to enforce the conventions helps us to keep a consistent naming standard across our Azure resources.

Secure resources with role-based access control

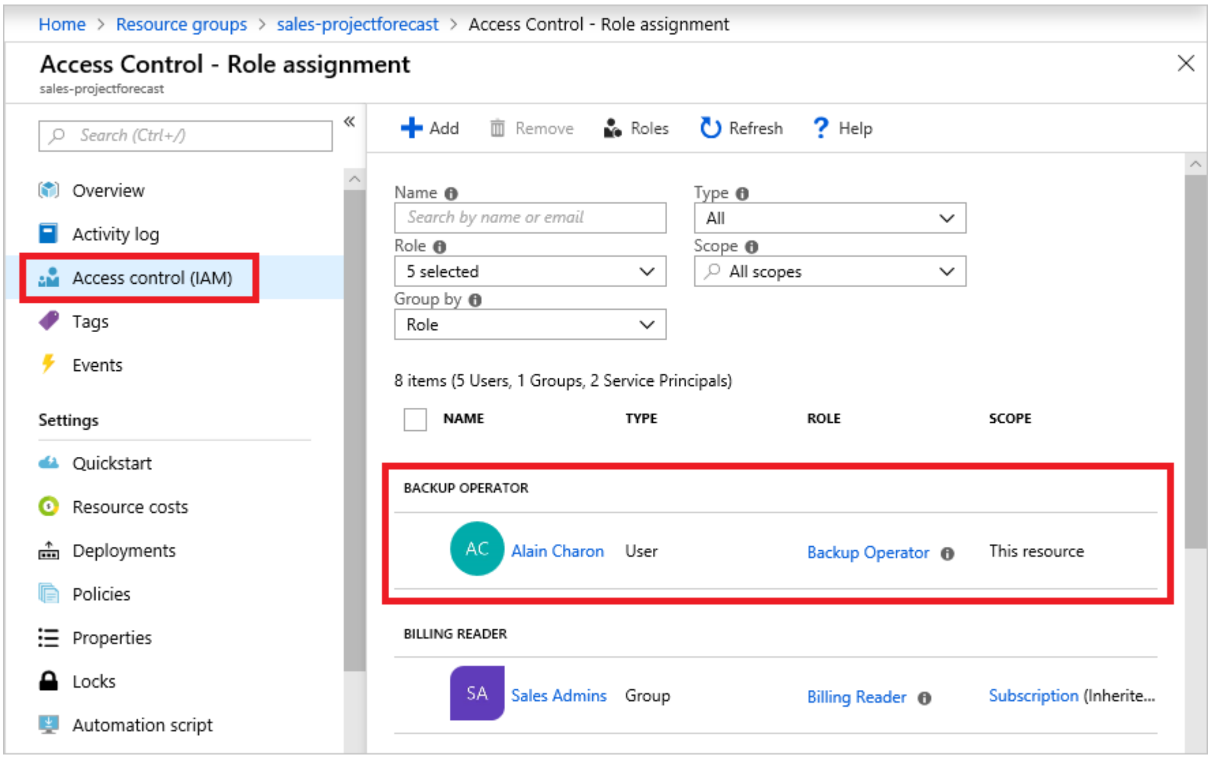
Implementing Azure Policy ensured that all our employees with Azure access are following our internal standards for creating resources, but we have a second issue we need to solve: how do we protect those resources once they are deployed? We have IT personnel that need to manage settings, developers that need to have read-only access, and administrators that need to be able to control them completely. Enter Role-Based Access Control (RBAC).

RBAC provides fine-grained access management for Azure resources, enabling you to grant users the specific rights they need to perform their jobs. RBAC is considered a core service and is included with all subscription levels at no cost.

Using RBAC, you can:

- Allow one user to manage VMs in a subscription, and another user to manage virtual networks.
- Allow a database administrator (DBA) group to manage SQL databases in a subscription.
- Allow a user to manage all resources in a resource group, such as VMs, websites, and virtual subnets.
- Allow an application to access all resources in a resource group.

To view access permissions, use the **Access Control** (IAM) blade in the Azure portal. On this blade, you can see who has access to an area and their role. Using this same blade, you can also grant or remove access.



Name: Search by name or email

Type: All

Role: 5 selected

Scope: All scopes

Group by: Role

8 items (5 Users, 1 Groups, 2 Service Principals)

NAME	TYPE	ROLE	SCOPE
BACKUP OPERATOR			
AC Alain Charon	User	Backup Operator	This resource
BILLING READER			
SA Sales Admins	Group	Billing Reader	Subscription (Inherite...

In the above screenshot, **Alain Charon** has been assigned the **Backup Operator** role for this resource group.

How RBAC defines access

RBAC uses an **allow model** for access. When you are assigned to a role, RBAC *allows* you to perform specific actions, such as read, write, or delete. Therefore, if one role assignment grants you read permissions to a resource group, and a different role assignment grants you write permissions to the same resource group, you will have write permissions on that resource group.

Best Practices for RBAC

Here are some best practices you should use when setting up resources.

- Segregate duties within your team and grant only the amount of access to users that they need to perform their jobs. Instead of giving everybody unrestricted permissions in your Azure subscription or resources, allow only specific actions at a particular scope.
- When planning your access control strategy, grant users the lowest privilege level that they need to do their work.
- Use **Resource Locks** to ensure critical resources aren't modified or deleted (more on that next!)

Use resource locks to protect resources

In a recent conversation, your manager mentioned that there had been instances where critical Azure resources had been mistakenly deleted. Since there was disorganization across their Azure environment, some good intentions of cleaning up unnecessary resources resulted in accidental deletion. You've heard of resource locks on Azure. You mention to your manager that you think you can help prevent this type of incident from happening in the future. Let's take a look at how you could use resource locks to solve this problem.

What are resource locks?

Resource locks are a setting that can be applied to any resource to block modification or deletion. Resource locks can set to either **Delete** or **Read-only**. Delete will allow all operations against the resource but block the ability to delete it. **Read-only** will only allow read activities to be performed against it, blocking any modification or deletion of the resource. Resource locks can be applied to subscriptions, resource groups, and to individual resources, and are inherited when applied at higher levels.

Note

Applying **Read-only** can lead to unexpected results because some operations that seem like read operations actually require additional actions. For example, placing a Read-only lock on a storage account prevents all users from listing the keys. The list keys operation is handled through a POST request because the returned keys are available for write operations.

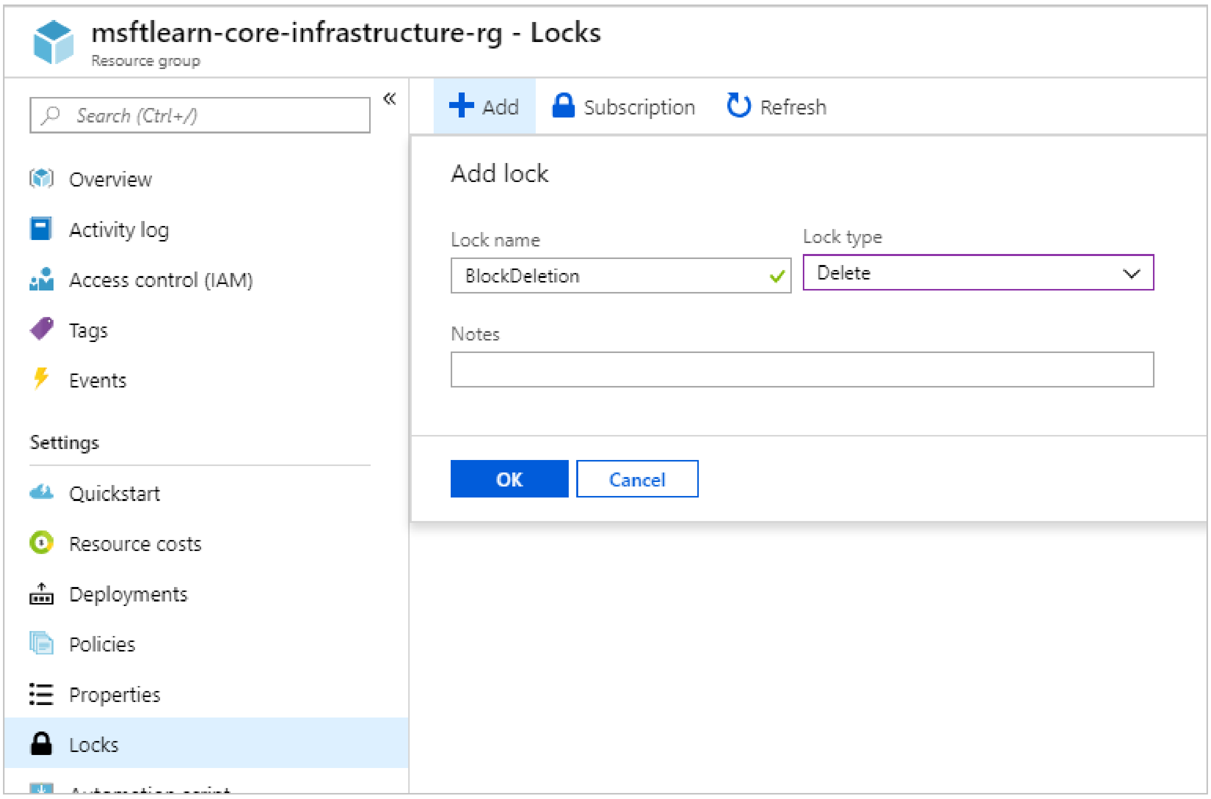
When a resource lock is applied, you must first remove the lock in order to perform that activity. By putting an additional step in place before allowing the action to be taken on the resource, it helps protect resources from inadvertent actions, and helps protect your administrators from doing something they may not have intended to do. Resource locks apply regardless of RBAC permissions. Even if you are an owner of the resource, you must still remove the lock before you'll actually be able to perform the blocked activity.

Let's take a look at how a resource lock works in action.

Create a resource lock

Recall our **msftlearn-core-infrastructure-rg** resource group. We've now got two VNets and a storage account in them. We consider these resources to be critical pieces of our Azure environment, and want to ensure that they aren't mistakenly deleted. Let's apply a resource lock to the resource group to prevent the resource group and its contained resources from being deleted.

1. Go ahead and pull up the [Azure portal](https://portal.azure.com)in a web browser if you haven't already. In the search box in the top navigation bar, search for **msftlearn-core-infrastructure-rg** and click on the resource group.
2. In the **Settings** section in the left menu, select **Locks**. You should see that the resource currently has no locks. Let's add one.
3. Click **+ Add**. Name the lock **BlockDeletion** and select a **Lock type** of **Delete**. Click **OK**.



msftlearn-core-infrastructure-rg - Locks
Resource group

Add lock

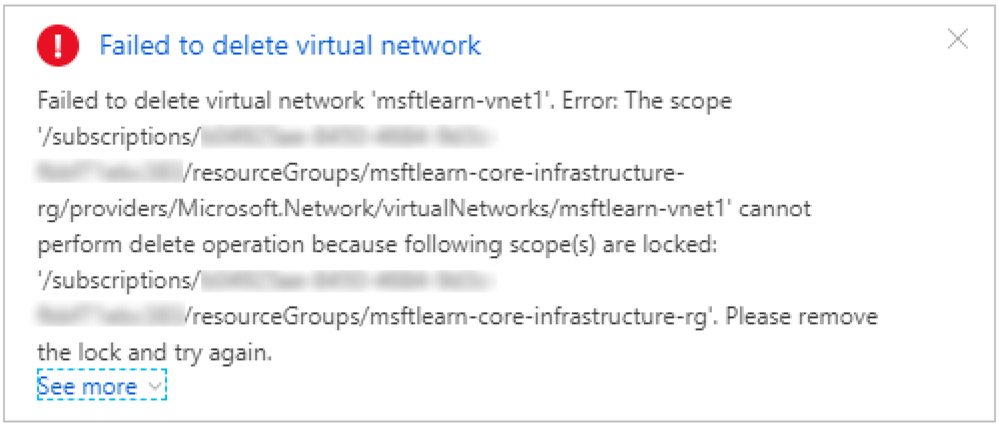
Lock name: BlockDeletion

Lock type: Delete

Notes:

You now have a lock applied to the resource group that will prevent deletion of the resource group, and is inherited by all resources within the resource group. Let's try to delete one of the VNets to see what happens.

4. Go back to **Overview**, and click on **msftlearn-vnet1** to view the resource.
5. In the **Overview** pane for **msftlearn-vnet1**, click **Delete** at the top, then **Yes** to confirm. You should receive an error, stating that there is a lock on the resource preventing its deletion.



Failed to delete virtual network

Failed to delete virtual network 'msftlearn-vnet1'. Error: The scope '/subscriptions/[redacted]/resourceGroups/msftlearn-core-infrastructure-rg/providers/Microsoft.Network/virtualNetworks/msftlearn-vnet1' cannot perform delete operation because following scope(s) are locked: '/subscriptions/[redacted]/resourceGroups/msftlearn-core-infrastructure-rg'. Please remove the lock and try again.

6. In the **Settings** section in the left menu, select **Locks**. You should see here that our **msftlearn-vnet1** has a lock that is inherited by from the resource group.
7. Navigate back to the **msftlearn-core-infrastructure-rg** resource group, and bring up the **Locks** pane. Let's remove our lock so we can clean up. Click the **...** on the **BlockDeletion** lock and select **Delete**.

Using resource locks in practice

We've seen how resource locks can protect from accidental deletion. In order to delete the virtual network, we needed to remove the lock. This concerted action helps ensure that you really intend to delete or modify the resource in question.

Use resource locks to protect those key pieces of Azure that could have a large impact if they were removed or modified. Some examples are ExpressRoute circuits, and virtual networks, critical databases, and domain controllers. Evaluate your resources, and apply locks where you'd like to have an extra layer of protection from accidental actions.

Summary

We've taken a look at several features you can use to put organization and control around your Azure resources.

We talked about how resource groups worked, and some ways you can use them to organize your resources.

We looked at how tags allow you to add custom contextual information to your resources, for use in areas such as billing and filtering.

We saw how we could use policies to enforce standards across our Azure resources.

We used resource locks to prevent accidental deletion of critical resources.

By using these tools throughout your Azure environment, you'll have greater organization across your Azure resources.

Clean up

Let's clean up the resources that we created. Since we deployed everything in a single resource group, cleaning up is easy.

1. Go ahead and pull up the Azure portal in a web browser if you haven't already. In the search box in the top navigation bar, search for **msftlearn-core-infrastructure-rg** and click on the resource group.
2. In the **Overview** pane, click **Delete resource group**. Enter the **msftlearn-core-infrastructure-rg** resource group name to confirm, and click **Delete**.