Microsoft

Automate Database Tasks for Azure SQL

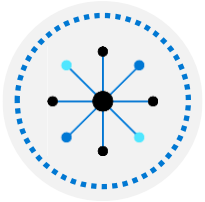
Introduction to Automated Deployment, Scheduled Tasks, Notifications and Alerts, and Azure resources for automation

Objectives

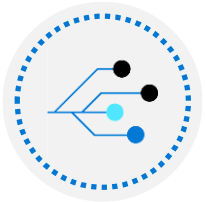
- Automate deployments
- Create Scheduled Tasks in SQL Server
- Create notifications for failures and performance alerts
- Configure automation for PaaS Services

Automate Deployment of Database Resources

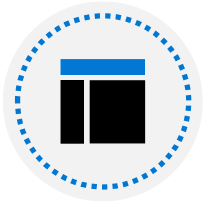
Objectives



How to automate deployment by using ARM templates and Bicep



How to automate deployment by using PowerShell and CLI

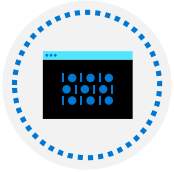


Monitor deployments

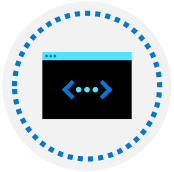
Azure deployment methods

ARM templates and Bicep files	Allow for the most complete, customizable deployment model for Azure resource deployment
Azure PowerShell	Commonly used for resource modifications and status retrieval
Azure CLI	Similar to PowerShell, Azure CLI is commonly used for resource status and modifications. It is built on the bash shell
Azure Portal	Acts as a graphical interface to Azure ARM and can be used to generate ARM templates
Azure DevOps	Deployments are carried out using Azure Pipelines. Azure Pipelines allows you to automate the build, testing, and deployment of your code

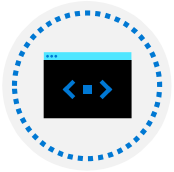
Azure Resource Manager (ARM) templates



An ARM template is a JavaScript Object Notation file that describes the Azure Resources to be deployed within it



ARM templates provide a declarative syntax to interact with the Azure API

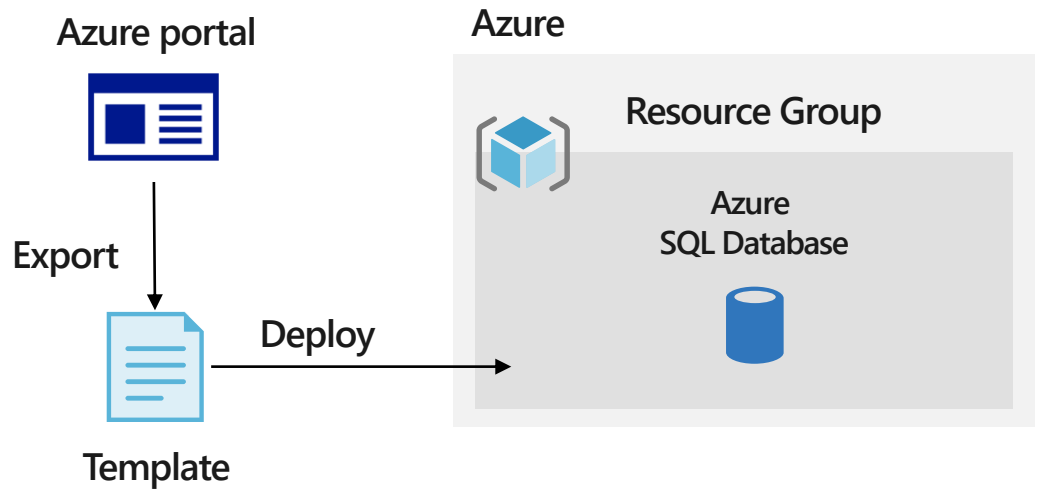


ARM templates can be deployed through PowerShell, Azure CLI, and Azure DevOps pipelines

Microsoft offers pre-built templates for common deployment scenarios
(https://github.com/Azure/azure-quickstart-templates/tree/master/101-sql-vm-new-storage)

Azure Resource Manager (ARM) template benefits

- Improves consistency
- Repeatable and modular
- Reduce error caused by manual mistakes
- Promotes reuse
- Simplifies orchestration



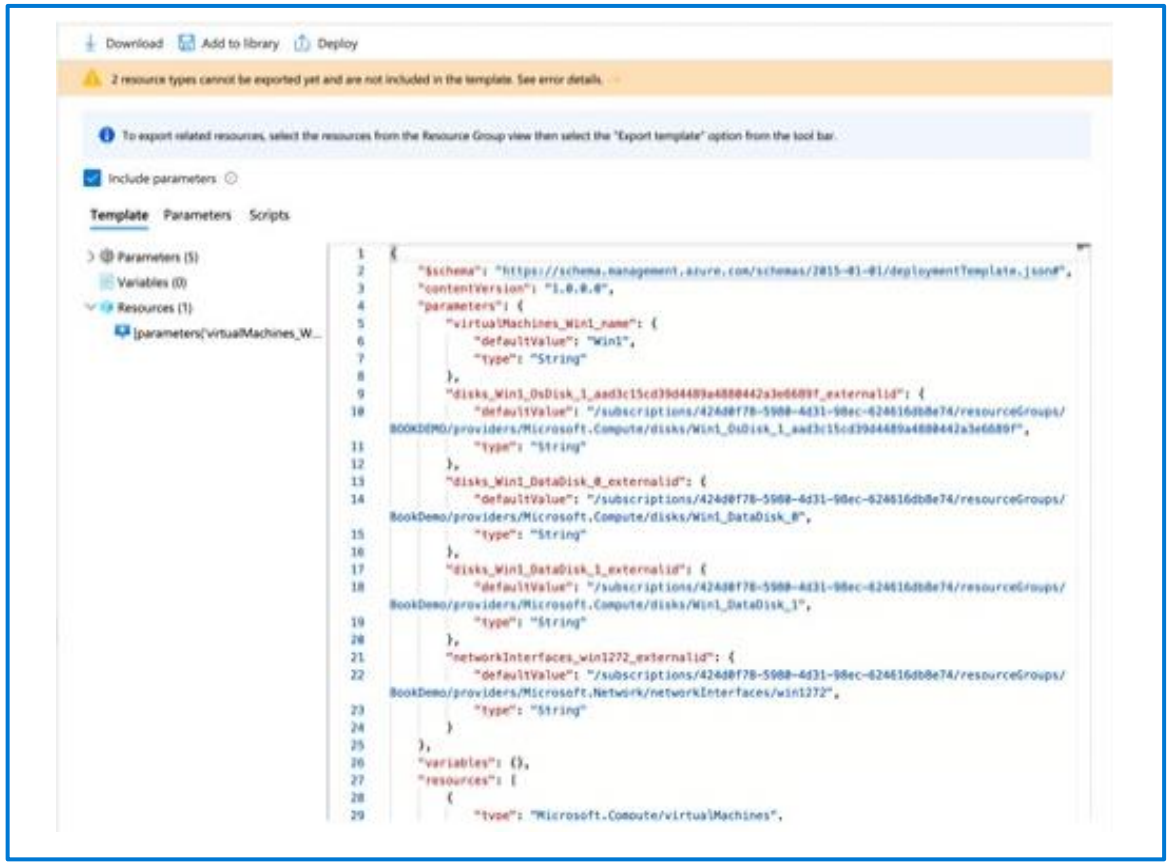
JSON files that automate the deployment of resources

ARM template

Deploys a full set of resources in one single declarative template

Dependencies and parameters can be built

Templates may be exported from the portal



Creating an ARM template

```
{
"$schema": "https://schema.management.azure.com/schemas/2019-04-
01/deploymentTemplate.json#",
"contentVersion": "",
"apiProfile": "",
"parameters": { },
"variables": { },
"functions": [ ],
"resources": [ ],
"outputs": { }
}
```

ARM template deployment

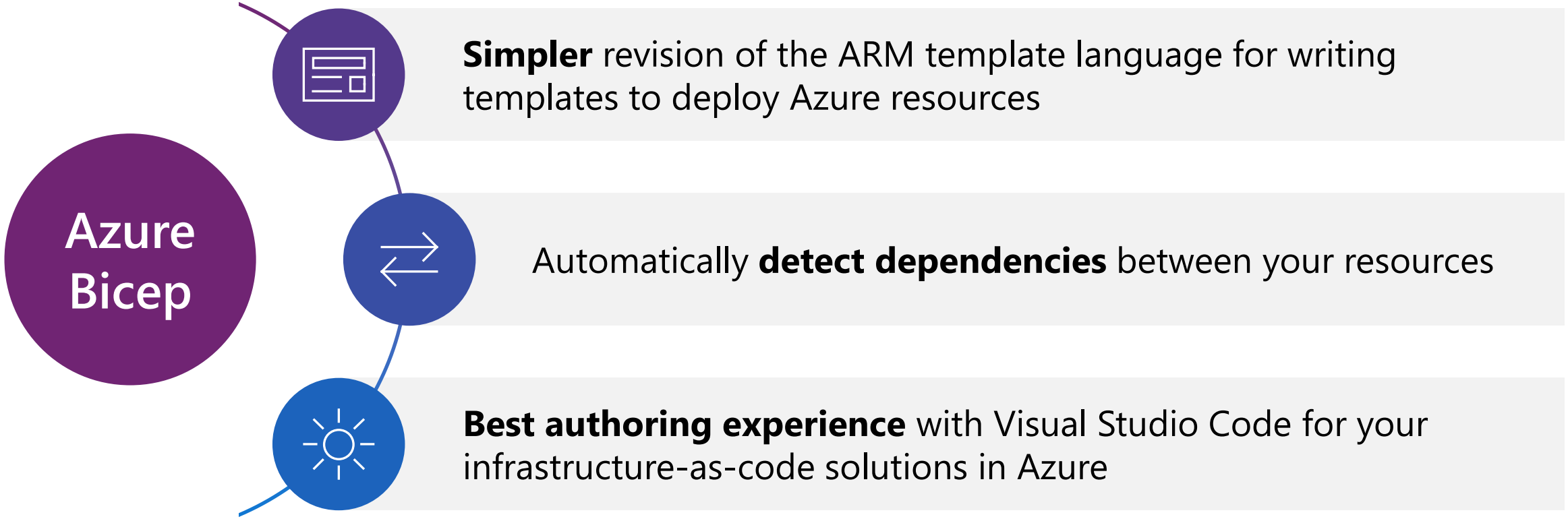
Azure PowerShell

```
New-AzResourceGroupDeployment -Name ExampleDeployment -ResourceGroupName
ExampleResourceGroup `
  -TemplateFile c:\MyTemplates\azuredeploy.json `
  -TemplateParameterFile c:\MyTemplates\storage.parameters.json
```

Azure Command Line Interface (CLI)

```
az deployment group create --resource-group SampleRG --template-file `
‘\path\template.json’
```

What is Azure Bicep?



Azure Bicep

Target your deployment to a resource group, subscription, management group, or tenant by using different options:

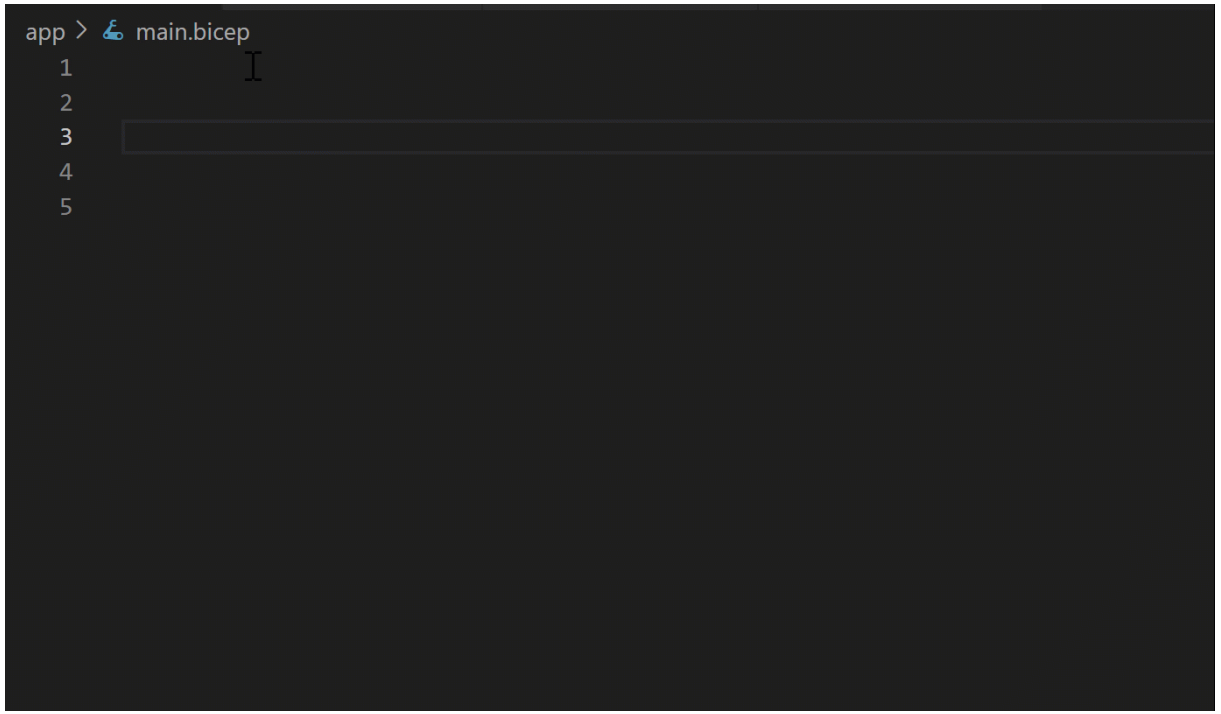
- **Deploy a local Bicep file:** use the `--template-file` parameter in the deployment command

- **Deploy a remote Bicep file:** not supported currently

- **Deploy template specs:** not supported currently by Azure CLI; but you can create a Bicep file with the Microsoft.Resources/templateSpecs resource to deploy

Azure Bicep benefits

- Continuous full support
- Simple syntax
- Easy to use
- Rich editor



Azure Bicep files vs. ARM template

Bicep

```
param location string = resourceGroup().location
param storageAccountName string = 'toylaunch${uniqueString(resourceGroup().id)}'

resource storageAccount 'Microsoft.Storage/storageAccounts@2021-06-01' = {
  name: storageAccountName
  location: location
  sku: {
    name: 'Standard_LRS'
  }
  kind: 'StorageV2'
  properties: {
    accessTier: 'Hot'
  }
}
```

JSON Copy

```
{
  "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "location": {
      "type": "string",
      "defaultValue": "[resourceGroup().location]"
    },
    "storageAccountName": {
      "type": "string",
      "defaultValue": "[format('toylaunch{0}', uniqueString(resourceGroup().id))]"
    }
  },
  "resources": [
    {
      "type": "Microsoft.Storage/storageAccounts",
      "apiVersion": "2021-06-01",
      "name": "[parameters('storageAccountName')]",
      "location": "[parameters('location')]",
      "sku": {
        "name": "Standard_LRS"
      },
      "kind": "StorageV2",
      "properties": {
        "accessTier": "Hot"
      }
    }
  ]
}
```

Azure Bicep deployment

Azure PowerShell

```
New-AzResourceGroupDeployment -ResourceGroupName SampleRG -TemplateFile ./main.bicep
-administratorLogin "<admin-login>"
```

Azure Command Line Interface (CLI)

```
az deployment group create --resource-group SampleRG --template-file –f `
‘\path\template.json’
```

Azure CLI

Create Azure SQL Database

```
az sql server create --name ServerName --resource-group RGName `
--location Location --admin-user $login --admin-password $password
```

Create a firewall rule for Azure SQL Database

```
az sql server firewall-rule create --resource-group RGName `
--server ServerName -n AllowYourIp --start-ip-address 0.0.0.0 --end-ip-address
0.0.0.0
```

Az.Sql PowerShell module

Returns information about an Azure SQL Database

```
Get-AzSqlServer -ResourceGroupName "ResourceGroup01" -ServerName "Server01"
```

Create an Azure SQL Managed Instance database

```
New-AzSqlDatabase -ResourceGroupName "ResourceGroup01" -ServerName "Server01" -
DatabaseName "Database01"
```

Instructor led labs: Deploy an Azure SQL Database using an Azure Resource Manager template

Create and Manage SQL Agent Jobs

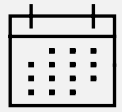
Objectives

What Maintenance Activities you should perform on your databases

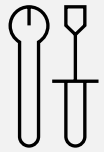
How to configure notifications and alerts on SQL Server Agent jobs and SQL Server

How to configure notifications alerts based on performance monitor values

SQL Server maintenance activities



Databases need regular maintenance activities



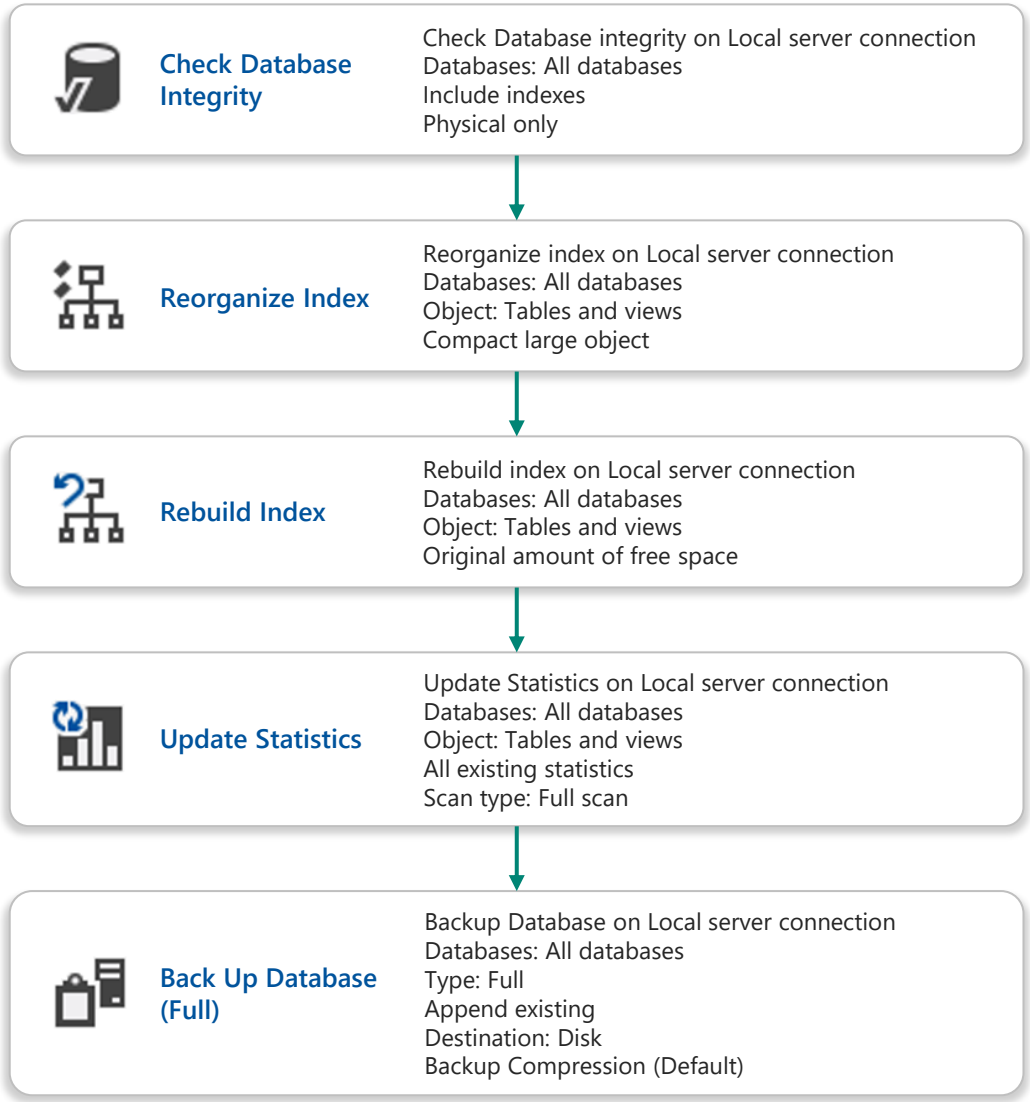
Maintenance activities can be performed using maintenance plans or through open source tools

Some common examples of these activities include:

- Database Backups
- Database Consistency Checks
- Index Maintenance
- Statistics Updates

Best practices for maintenance plans

- Do Not Use the Database Shrink task
- Stagger backups across servers and large databases to avoid potentially overwhelming your storage account or storage device
- Create separate maintenance plans for each set of maintenance activities

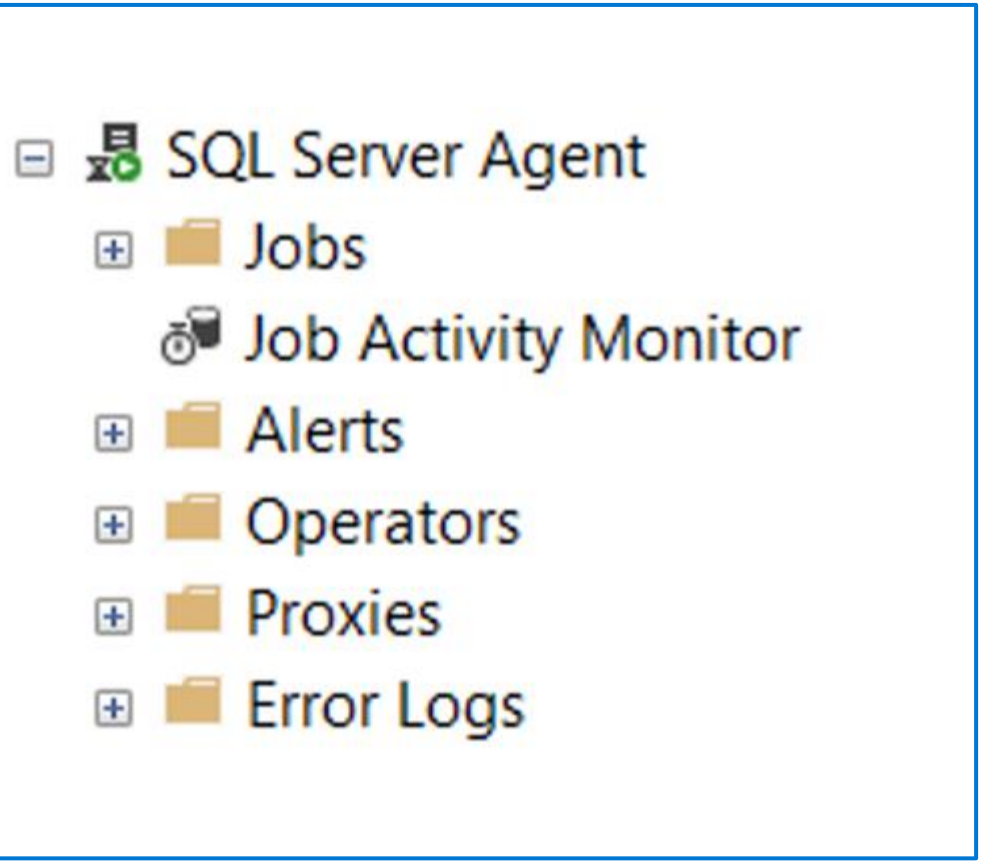


SQL Server Agent

The SQL Server Agent provides automation for SQL Server and Azure SQL Managed Instance

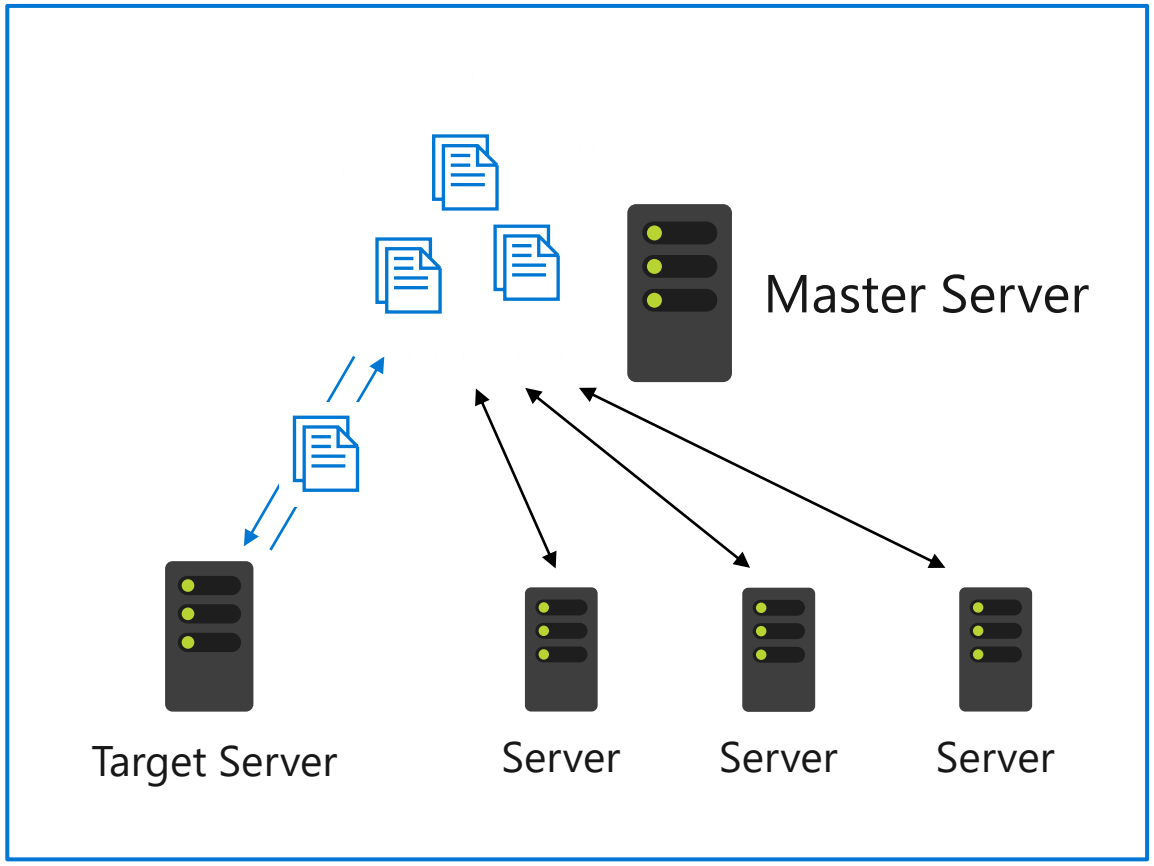
The SQL Server Agent also provides notifications for both job failures and alerts that are written to the error log

The service can also be configured to notify administrators



Multi-server automation

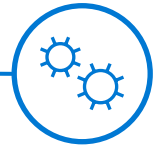
- The SQL Server Agent supports designating one server as a primary to execute jobs on other target servers
- Target servers connect to the primary to ensure job schedules are updated on a regular basis
- This can be used to execute maintenance across your environment



Task status notifications

The SQL Server Agent can be configured to notify an operator for job completion, success, and failure

Usually only enabled for failure to reduce notification volume



To deliver email notifications, the database mail and SQL Server Agent mail profile must be enabled

Can create alerts on performance conditions and error conditions found in the instance's error log

Operators

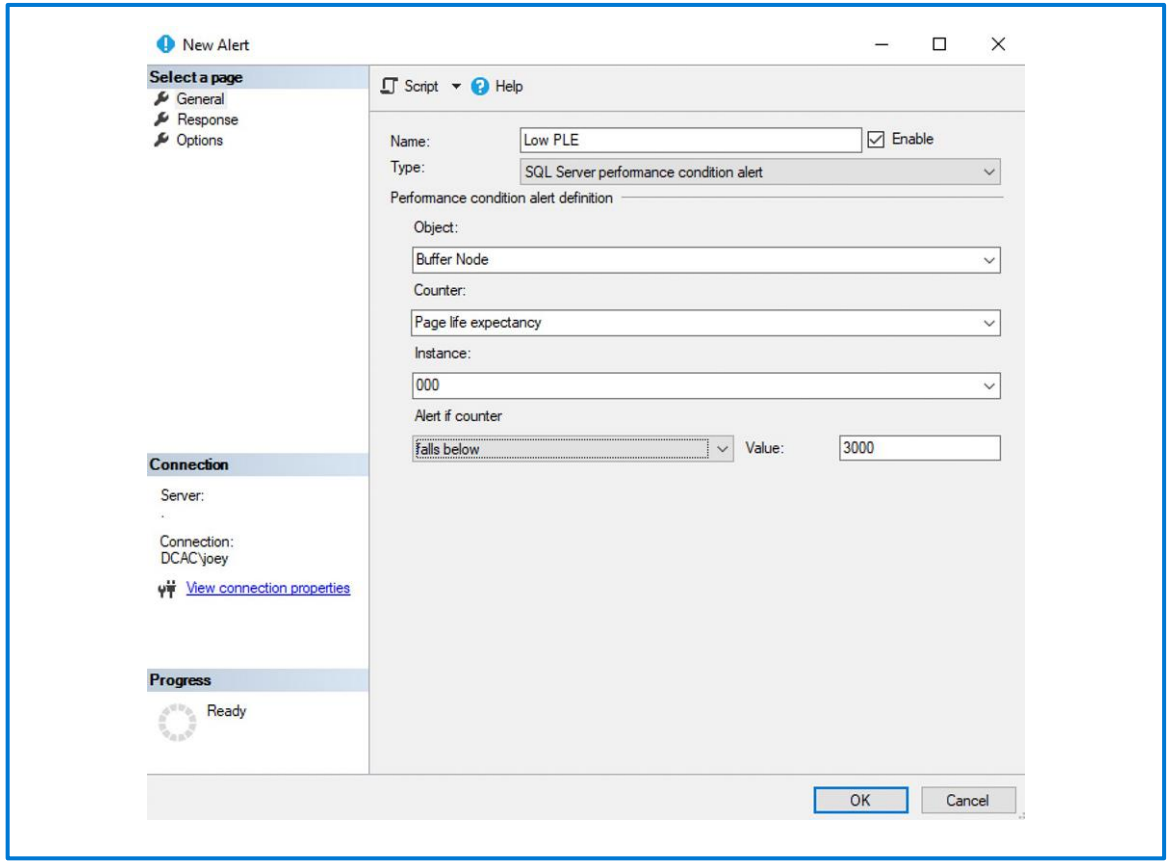
Operators act an alias for a user or group of users that have been configured in the SQL server agent

Typically, an operator will map to a group of administrators who will receive a notification or alert

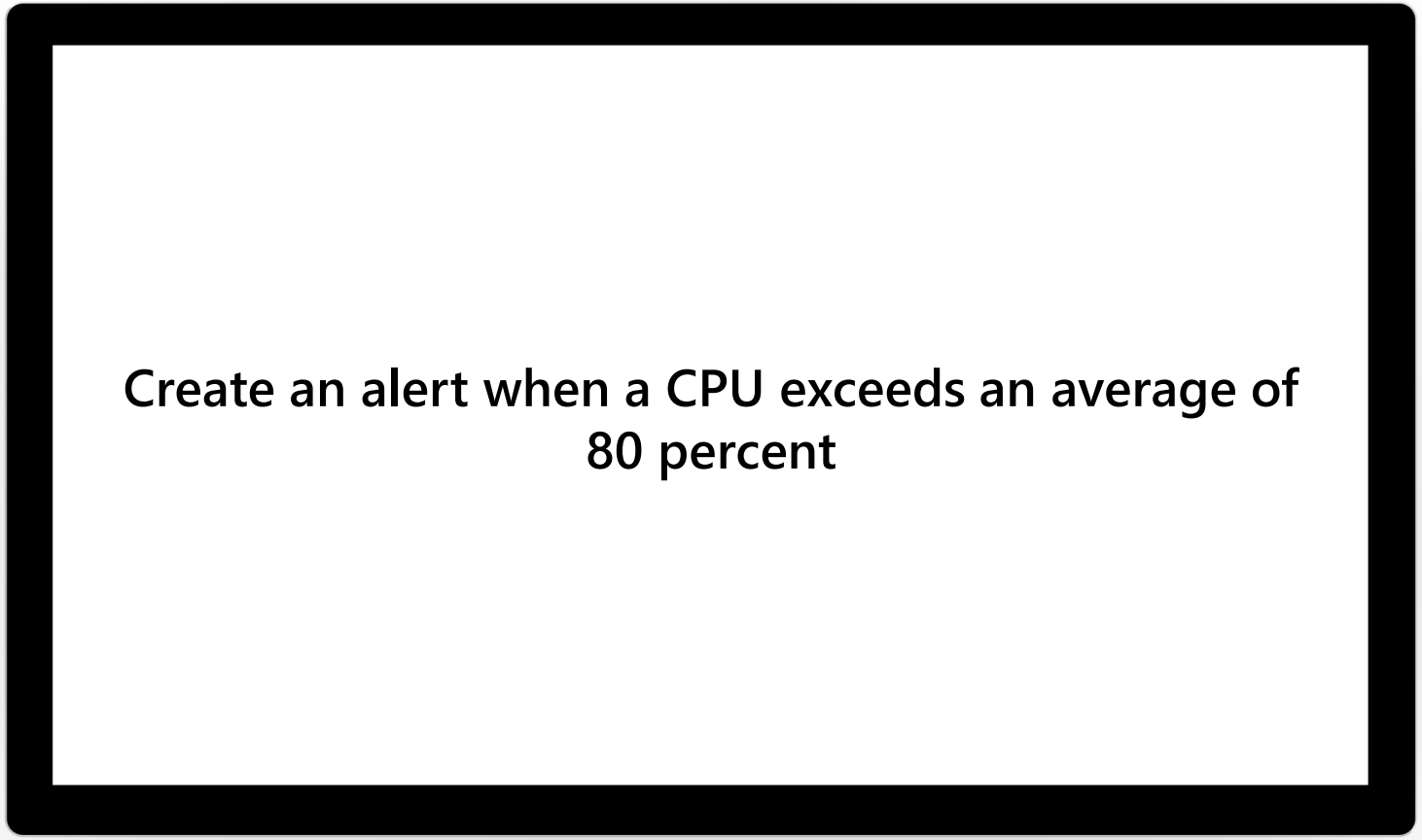
Notifications based on metrics

Alerts are triggered in the event of certain server conditions:

For example, if CPU utilization is over 90% for a period of five minutes, or Page Life Expectancy drops below a certain value



Instructor led labs: Create a CPU status alert for a SQL Server

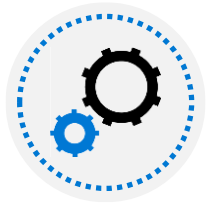


Manage Azure PaaS Tasks by Using Automation

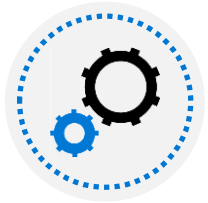
Objectives

The benefits of Azure Policy

The capabilities of Azure Automation



How to use elastic jobs



How to use Logic Apps

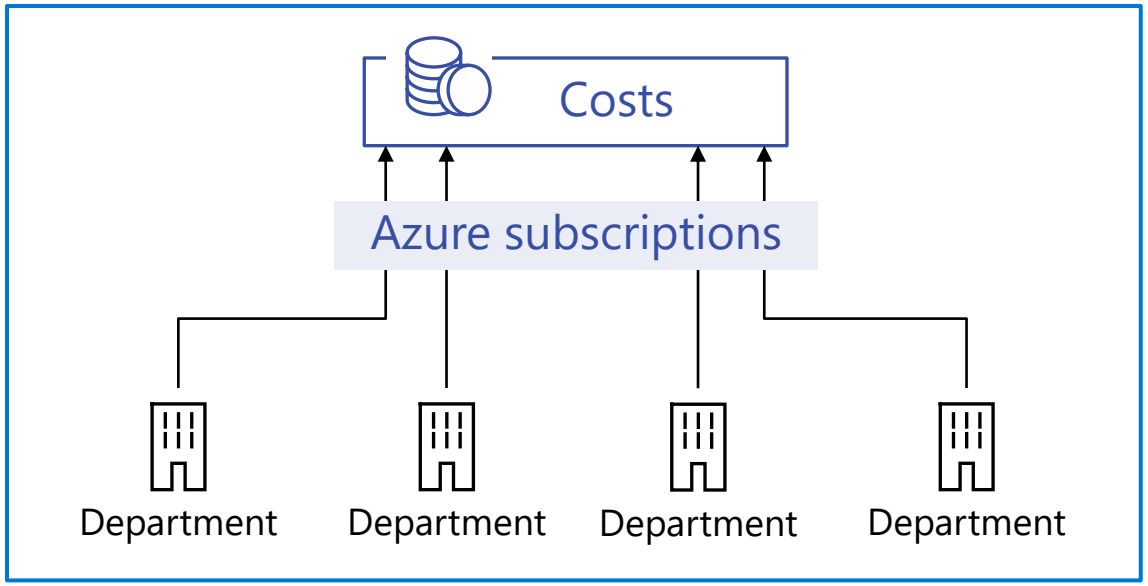
Implementing Azure Policy

Azure Policy is used by administrators to ensure consistency across an Azure environment

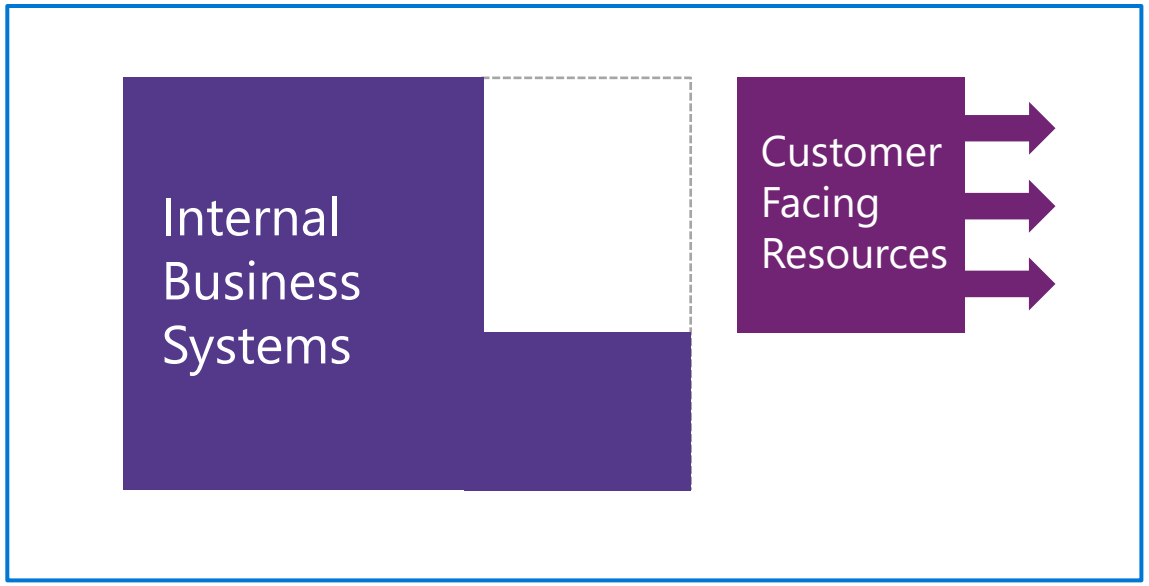
Policy can be used to enforce naming conditions, limit the types of Azure resources deployed in a subscription, or limit the Azure regions where deployment is allowed

Multiple policies can be combined to create an initiative

Why would I have multiple Azure subscriptions?



Many large organizations use Azure subscriptions as a billing boundary so that costs are easily tracked back to a department



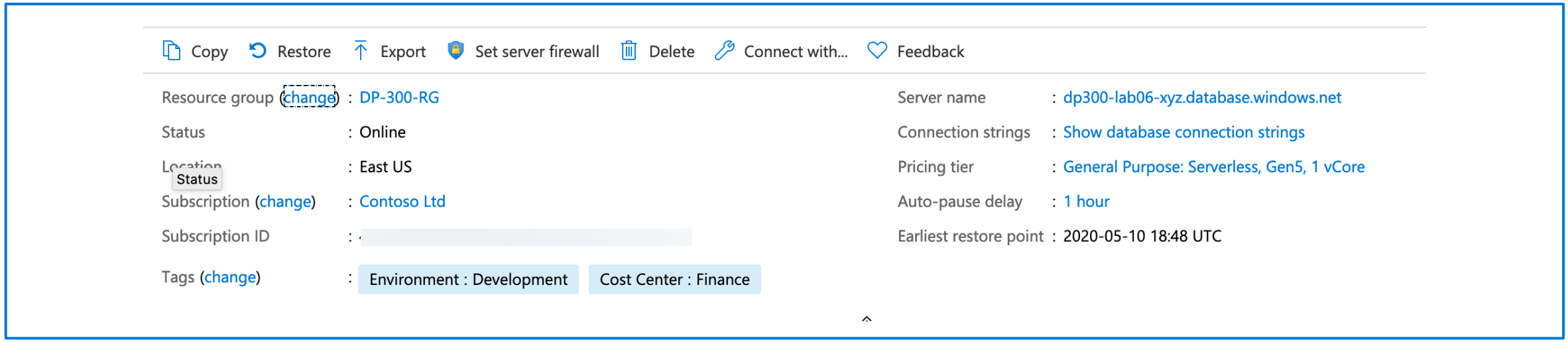
Other uses include separating customer facing resources from internal business systems

Why should I tag Azure resources?

Tags allow you to describe your Azure resources beyond simply the resource name

Tags are key value pairs – each Azure resource can have up to 15 tags

Some sample uses for tags include Cost Center, Environment, and CreateDate



Azure Automation

Azure Automation allows you to create regularly scheduled tasks that run against any Azure resource or even on-premises virtual machines

Automation allows you to perform maintenance activities against Azure SQL Database

Automation can also be used to ensure consistent settings across multiple VMs

Overview of Azure Automation components

Runbooks	Unit of execution in Azure Automation and may be created using PowerShell or Python. You have the option of deploying graphical runbooks, but they are limited in their capabilities
Modules	Used to execute PowerShell cmdlets within your runbooks. Load the modules for the PowerShell cmdlets you need for your runbooks
Credentials	Store sensitive information like passwords for use by runbooks
Schedules	Allow for runbooks to be scheduled for regular execution

Building an Azure Automation runbook

Building a runbook requires you to create automation account

For PowerShell runbooks, you will need to import the PowerShell modules you need in order to execute your runbook (e.g. Az.SQL)

You can also include credentials and run as accounts in your automation account

You can create schedules in your automation account and tie them to your runbooks

Automation includes a test pane you can test your code in the execution context of Azure Automation

Elastic Jobs in Azure SQL Database

Since Azure SQL Database lacks an agent, Elastic Jobs allow for a T-SQL scheduled execution methodology

Elastic Jobs require a dedicated SQL database to hold the metadata for your jobs

Define a target group, a SQL Database server, one or more databases or elastic pools as job targets

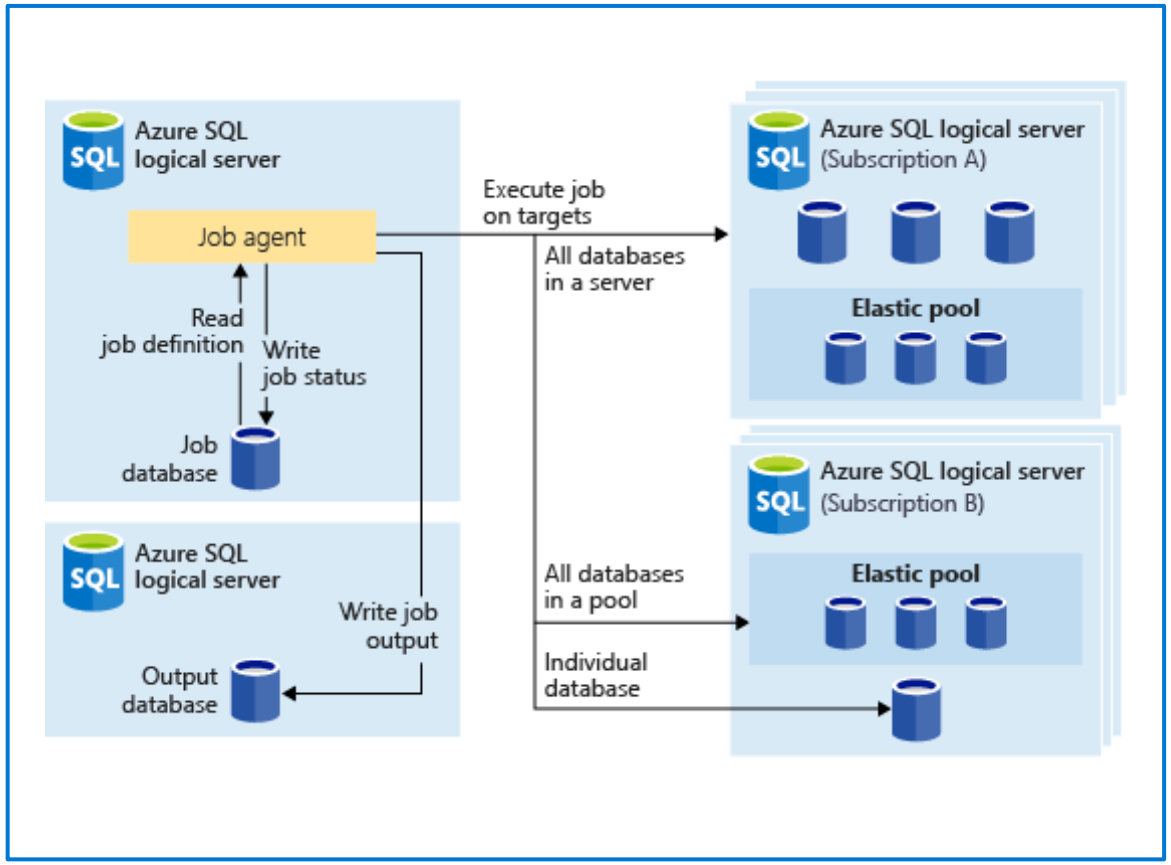
Database resources can run on different Azure subscriptions, and/or regions

Azure SQL Managed Instance doesn't support elastic jobs

Elastic jobs

Components

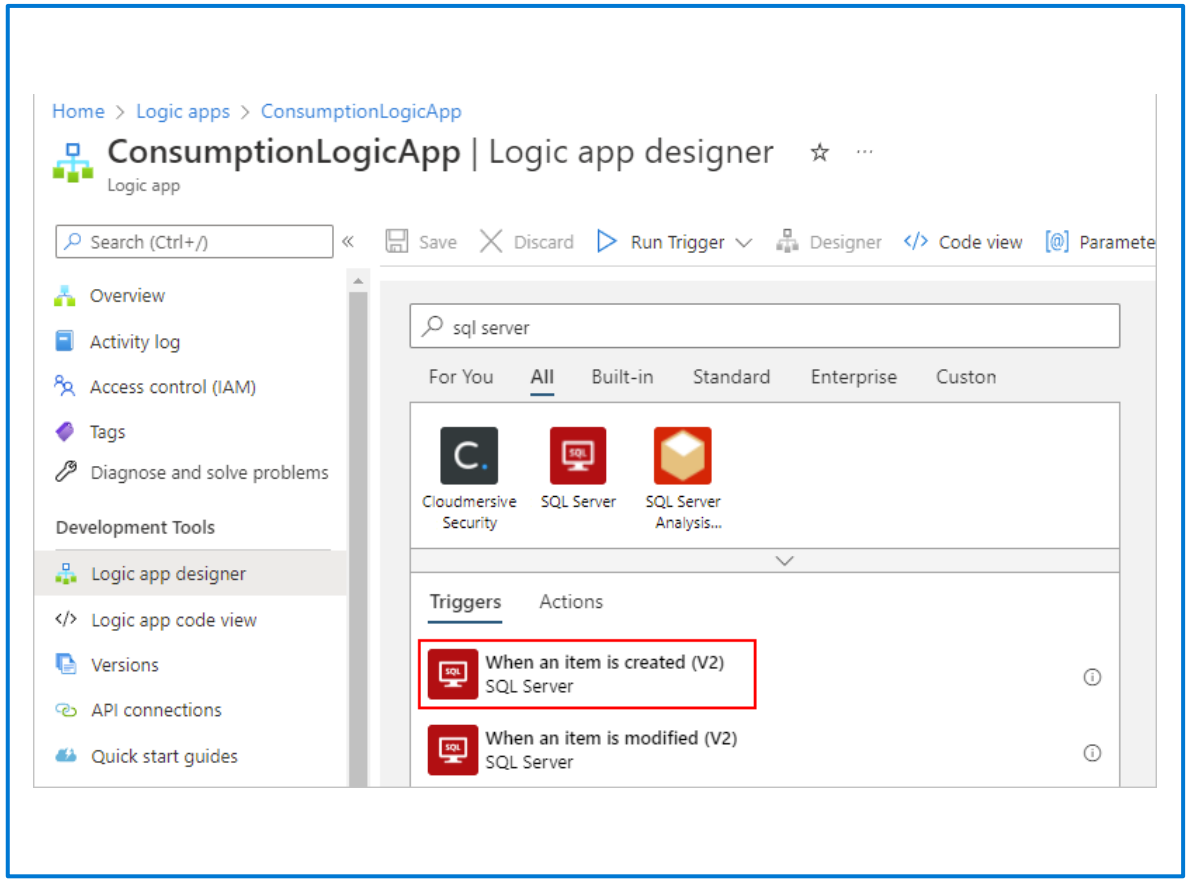
- Elastic Job agent
- Job database
- Target group
- Job



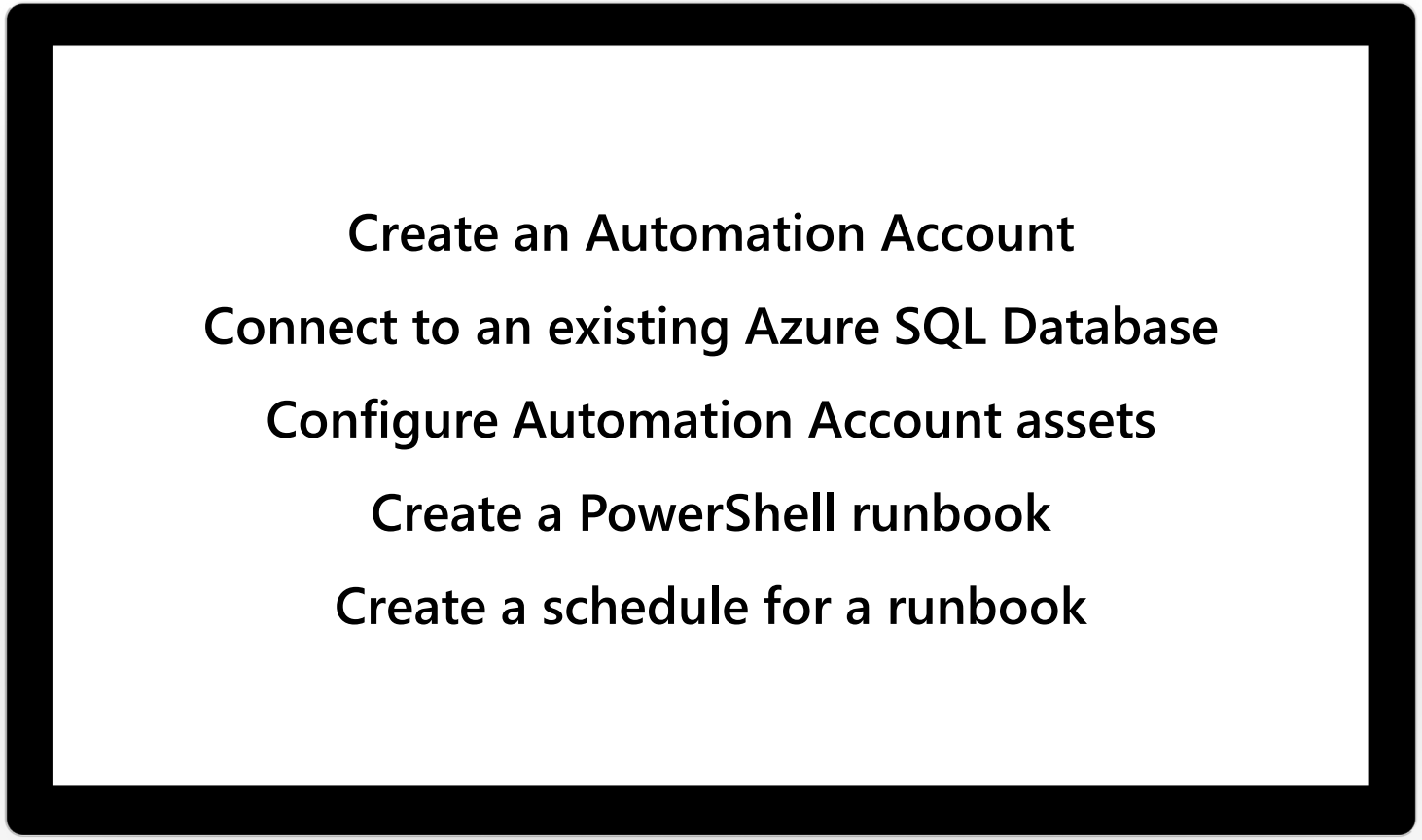
Azure Logic Apps

Platform for creating and running automated workflows that integrate apps, data, services, and systems

- Build your workflow from the components using the design tool.
- SQL Server connector supports the following SQL editions: **SQL Server**, **Azure SQL Database**, and **Azure SQL Managed Instance**
- Built-in SQL Server connector has **no triggers**
- Built-in SQL Server connector has only one operation: **Execute Query**



Instructor led labs: Deploy an automation runbook to automatically rebuild indexes



Summary

Automate deployment of database resources:

- Understand ARM template and Bicep structure
- Deployment options for ARM templates, PowerShell, and Azure CLI
- Using quick start Azure templates

Create and manage SQL Agent jobs:

- Learn the capabilities of the SQL Server Agent
- Configure maintenance tasks for SQL Server
- Creating notifications for failures and performance alerts

Manage Azure PaaS tasks using automation:

- Introduction to Azure Automation, Elastic Jobs, and Azure Logic Apps
- Create an Azure Automation runbook