

Azure Batch

Azure **Batch Service** is a cloud-based **job scheduling** and **compute management** service that enables users to run **large-scale parallel and high-performance computing (HPC) applications** efficiently. It automates resource provisioning, job scheduling, and workload execution on a **pool of virtual machines (VMs)** in Azure.

How Azure Batch Works

1. **Create a Batch Account** – This acts as a container for managing compute resources.
2. **Provision a VM Pool** – A set of VMs that execute tasks in parallel.
3. **Upload Applications & Data** – Store the necessary application binaries and input files in Azure Storage.
4. **Create & Manage Jobs** – A job consists of multiple tasks that execute the application on VMs.
5. **Run & Monitor Jobs** – Azure Batch distributes workloads across VMs, executing tasks efficiently.
6. **Store & Analyze Results** – The processed output is saved in **Azure Storage** or **Azure SQL Database**.

Use Cases of Azure Batch Service

1. Data Processing & ETL Pipelines

- Used for processing massive datasets in parallel.
- Example: Analyzing **log files** or running **big data analytics**.

2. Media Rendering & Transcoding

- Transcode large video files into multiple formats.
- Example: A **video streaming platform** converting videos into different resolutions.

3. Scientific Simulations & Modeling

- Run **complex simulations in engineering, physics, or finance**.
- Example: **Weather forecasting models** that process large datasets.

4. AI & Machine Learning Model Training

- Train large-scale ML models across multiple compute nodes.
- Example: Processing millions of **images for deep learning**.

5. Financial Risk Analysis

- Compute-intensive tasks such as **Monte Carlo simulations**.
- Example: **Banking & insurance** companies assessing financial risks.

6. Image Processing & OCR

- Process and analyze large image datasets for **object recognition or text extraction**.
- Example: **Digitizing historical documents with OCR (Optical Character Recognition)**.

Benefits of Azure Batch Service

1. Fully Managed Service

- No need to manage infrastructure; **Azure provisions and scales VMs automatically**.

2. Cost-Effective

- Supports **low-priority (spot) VMs**, reducing compute costs significantly.

3. High Scalability

- Automatically scales resources based on workload demands.

4. Parallel Processing

- Distributes tasks across multiple VMs for **faster execution**.

5. Integration with Azure Ecosystem

- Works with **Azure Storage, SQL Database, and AI services** for a complete cloud workflow.

6. Custom VM Images & Software

- Supports **custom VM images** with pre-installed software for specialized tasks.

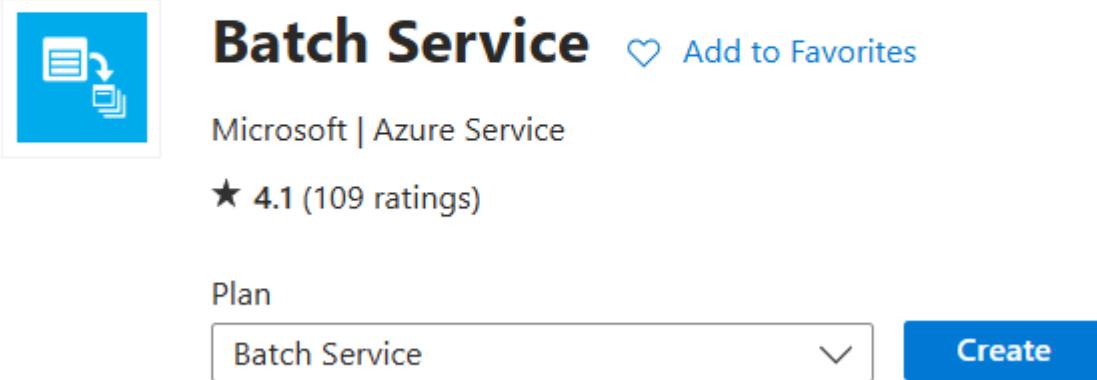
Conclusion

Azure Batch Service is ideal for **compute-intensive workloads** requiring **parallel processing and automation**. It simplifies **job scheduling, resource management, and scaling**, making it a powerful solution for industries like **finance, healthcare, media, AI, and scientific research**.

This process sets up an Azure Batch Service to automate data processing. You create a Batch account, SQL database, and storage account. After building a .NET app in Visual Studio to handle data, you upload the app as a zip file to Azure. You create a Batch Pool, add the app, and run a job with a command line to execute the app. The app reads data from Azure Blob Storage and writes it to the SQL database. The end goal is to automate data ingestion into SQL Server using Azure Batch for scalable, serverless processing. Let me know if you'd like me to refine this!

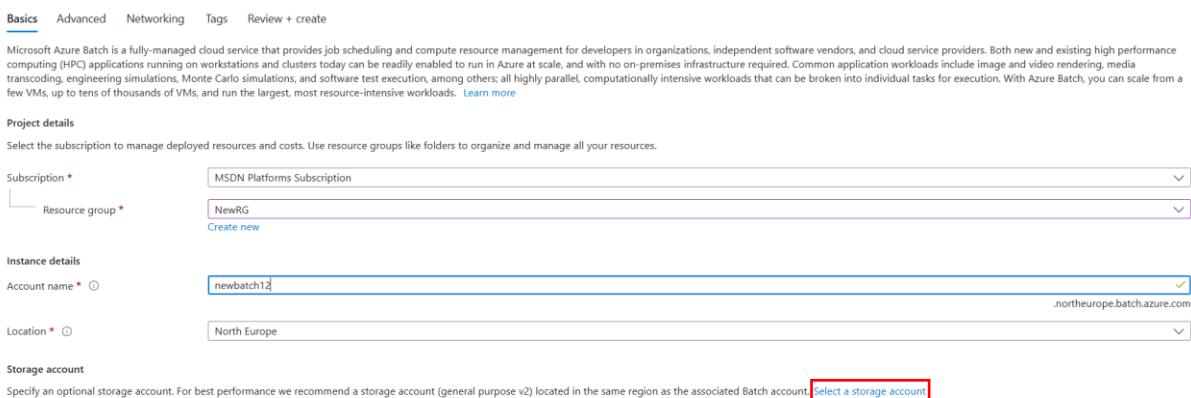
To begin with the Lab

1. Go to the marketplace in the Azure Portal then search for Batch Service. Click on Create.



The screenshot shows the Azure Marketplace page for the 'Batch Service'. At the top, there's a Microsoft logo and a 'Batch Service' title with a gear icon. Below the title are 'Add to Favorites' and 'Microsoft | Azure Service' buttons. A rating of '★ 4.1 (109 ratings)' is displayed. On the left, there's a blue icon representing the service. On the right, there's a 'Plan' section with a dropdown menu set to 'Batch Service' and a large blue 'Create' button.

2. Choose your resource group then give an account name, and choose your location. You will also need a storage account for this service. So, if you don't have a storage account, please create a new one.



The screenshot shows the 'Create a new resource' form for Azure Batch. The 'Basics' tab is selected. In the 'Project details' section, 'Subscription' is set to 'MSDN Platforms Subscription' and 'Resource group' is set to 'NewRG'. In the 'Instance details' section, 'Account name' is 'newbatch12' and 'Location' is 'North Europe'. In the 'Storage account' section, there's a note about specifying a storage account and a red box highlights the 'Select a storage account' link.

3. After that leave advanced and networking settings as they are. Move to the review page and create your Batch service.
4. Once your batch service is created then you have to create an **SQL database and connect it with the SQL Server Management Studio**.

The screenshot shows the Azure portal interface for a database named 'appdb' under the resource group 'NewRG'. The 'Essentials' section displays various properties: Status is 'Online', Location is 'North Europe', and it is part of the 'MSDN Platforms Subscription'. The 'Server name' is 'appdb12311.database.windows.net', and it is associated with a 'No elastic pool'. Connection strings and basic pricing tier are also listed. A note about replicating the database to Microsoft Fabric is present.

- Once you have connected your database in SSMS then create a table.

CREATE TABLE [BlobData]

```
(  
blobname varchar(1000),  
blobsize int,  
accesstier varchar(15)  
)
```

The screenshot shows the SSMS Object Explorer and SQL Query window. The Object Explorer displays the database structure for 'appdb12311.database.windows.net'. The SQL Query window contains the T-SQL code for creating a table named 'BlobData' with three columns: 'blobname' (varchar(1000)), 'blobsize' (int), and 'accesstier' (varchar(15)). The query is executed successfully, as indicated by the message 'Commands completed successfully.' at the bottom of the window.

- After that you have to download a zip file for an application from GitHub and open it in Visual Studio 2022.
- Now on the Azure Portal go to your Storage account create a container with the name scripts and upload some data on it.**

Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
New folder	3/4/2025, 1:27:09 PM			Block blob	62.94 MiB	Available
	3/4/2025, 1:27:32 PM	Hot (Inferred)		Block blob	21.43 MiB	Available
	3/4/2025, 1:27:09 PM	Hot (Inferred)		Block blob	35.1 MiB	Available
	3/4/2025, 1:27:27 PM	Hot (Inferred)		Block blob	2.59 MiB	Available
	3/4/2025, 1:27:00 PM	Hot (Inferred)		Block blob	2.16 MiB	Available
	3/4/2025, 1:27:00 PM	Hot (Inferred)		Block blob	1.44 MiB	Available
	3/4/2025, 1:27:07 PM	Hot (Inferred)		Block blob	18.41 MiB	Available
	3/4/2025, 1:27:01 PM	Hot (Inferred)		Block blob	11.18 MiB	Available

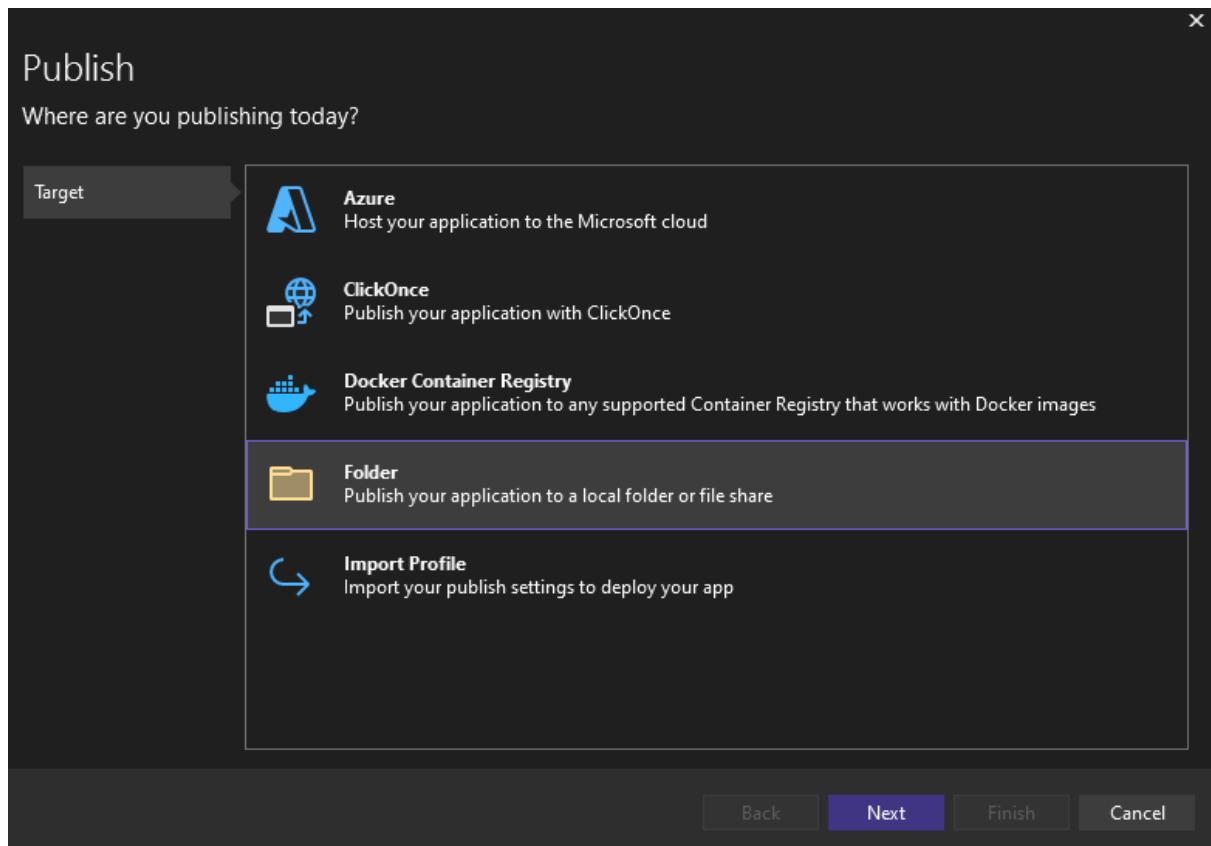
8. Open the Visual Studio and here you can see that on line number 7 we have the container name then on line number 6 you have to give the connection string of your storage account which you can find in the access keys tab.
9. Then you have to change the connection string for SQL Database as well and save the changes.

```

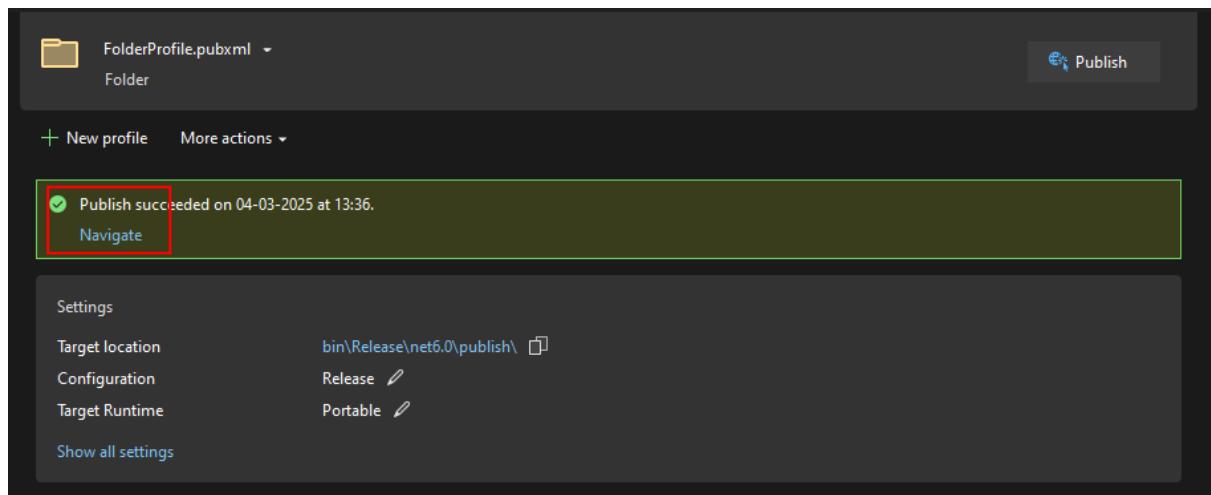
1  using Azure.Storage.Blobs;
2  using Azure.Storage.Blobs.Models;
3  using System.Data;
4  using System.Data.SqlClient;
5
6  string conenctionString = "DefaultEndpointsProtocol=https;AccountName=thestorageaccount1201;Accou
7  string containerName = "scripts";
8  int Id = 1;
9
10 BlobContainerClient blobContainerClient = new BlobContainerClient(conenctionString, containerName)
11 SqlConnection _connection = new SqlConnection("Server=tcp:appserver4434334.database.windows.net,1433;Tran
12
13 SqlParameter paramblobname = new SqlParameter();
14 paramblobname.ParameterName = "@blobname";
15
16 SqlParameter paramblobsize = new SqlParameter();
17 paramblobsize.ParameterName = "@blobsize";
18

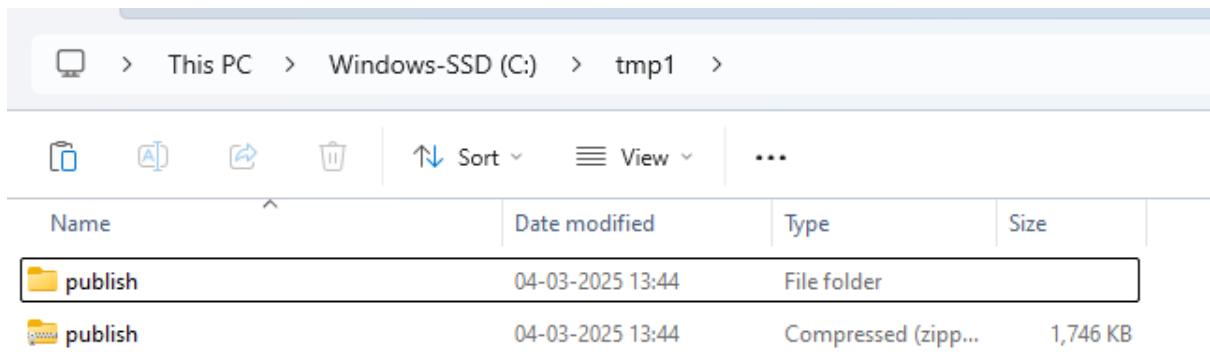
```

10. Then we are going to publish our application in the folder.



11. Once you have published the application open the folder copy the publish folder and paste it into another folder. Then convert the publish folder into a zip file.





12. Now open your Portal and navigate to the Batch service. Here you have to click on Pools from the left pane and then click on Add.

A screenshot of the Azure portal showing the 'newbatch12 | Pools' blade. On the left, there's a sidebar with options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Settings, Features, Applications, Pools (which is selected and highlighted in grey), and Jobs. The main area has a search bar, a filter section with 'State == all' and 'Add filter' buttons, and a pagination limit of 1. Below these are columns for Pool ID, Dedicated nodes, Spot/low-priority nodes, Current vCPUs, and VM size. A message states 'No pools were found for this Batch account'. At the top, there are buttons for '+ Add' (which is highlighted with a red box), 'Add (JSON editor)', 'Columns', and 'Refresh'.

13. First, we have to give a name to our Pool.

A screenshot of the 'Add pool' blade. The title is 'Add pool' and it's associated with the 'newbatch12' account. Under the 'POOL DETAILS' section, there are three fields: 'Pool ID *' with the value 'PoolA', 'Display name' with the placeholder 'Enter a display name (optional)', and 'Identity' with two options: 'None' (selected) and 'User-assigned'.

14. Then we have to choose an image from the market place and we have to choose Microsoft Windows Server 2022 data centre.

Image Type ⓘ	<input type="checkbox"/> Marketplace
Enable unverified image	<input type="checkbox"/>
Publisher *	microsoftwindowsserver
Offer *	windowsserver
Sku *	2022-datacenter
Batch Node Agent SKU ID	batch.node.windows amd64
Security type ⓘ	Standard
Enable Windows automatic updates	<input type="checkbox"/>

15. After that in the network you have to create a new network and then click on ok to add your Pool.

VIRTUAL NETWORK	
Pool endpoint configuration ⓘ	Inbound NAT pool
Virtual network ⓘ	<input type="text"/> batchnetwork Create new
Subnet	<input type="text"/> default
Accelerated Networking ⓘ	<input checked="" type="radio"/> Enabled <input type="radio"/> Disabled
IP address provisioning type ⓘ	<input checked="" type="radio"/> BatchManaged <input type="radio"/> UserManaged <input type="radio"/> NoPublicIPAddresses
Mount configuration	Mount configuration
Save money	
Save up to 40% with a license you already own.	
Already have a Windows Server license? * ⓘ</td <td><input type="radio"/> Yes <input checked="" type="radio"/> No</td>	<input type="radio"/> Yes <input checked="" type="radio"/> No

16. Now we need to open the Application tab to add the application which we published from the visual studio. Click on Add.

ID	Default version	Allow updates
No applications were found for this Batch account		

17. Give an application ID and Version number then the browser for the zip file.

Home > newbatch12 | Applications >

New application

Application Id * ⓘ

blobcopy

Version * ⓘ

1.0

Application package * ⓘ

"publish.zip"

18. Now you can see that our application has been added successfully. Now we need to create a job.

Home > Microsoft.BatchAccount_2025_3_4_15_9_56 | Overview > newbatch123

 **newbatch123 | Applications** ⚡ ⭐ ⋮

Batch account

Search

Add Columns Refresh

Overview Activity log Access control (IAM) Tags Diagnose and solve problems Settings Features

Applications Pools

Application usage: 1 / 200 applications

The application packages feature of Azure Batch provides easy management of task application:

Id	Default version
blobcopy	

19. Click on add to add a job.

newbatch123 | Jobs

Batch account

Search Add Add (JSON editor) Columns Refresh

Overview State == all Add filter

Activity log Filter by ID or pool

Access control (IAM)

Tags

Diagnose and solve problems

Settings

Features

Applications

Pools

Jobs

Job schedules

ID State Pool

No jobs were found for this Batch account

20. Here you need to give a job name then choose your Pool, after that in the Mode choose Custom and click on Job Manager task.

Add job

newbatch123

BASIC INFORMATION

Job ID *

Current pool [\[PoolA\] Change pool](#)

JOB MANAGER, PREPARATION AND RELEASE TASKS

Mode None Custom

Job manager task

Job preparation task

21. In the job manager task give a task ID then put a command line and scroll down.

```
cmd /c dotnet
%AZ_BATCH_APP_PACKAGE_blobcopy#1.0%\publish\BlobCopy.dll
```

Job manager task ...

newbatch123

BASIC INFORMATION

Task ID * ⓘ

TaskA

cmd /c dotnet %AZ_BATCH_APP_PACKAGE_blobcopy#1.0%\publish\BlobCopy.dll

Command line ⓘ

Allow Spot/low-priority node ⓘ

True False

22. Then in the application packages choose your application name and version number.
Just create your Job.

1 application package references

[Application packages](#)

0 environment settings selected

[Environment settings](#)

 To allow tasks to run in containers, the container configuration must be specified for the task's pool. Please e

Select

23. Now you can see that our job has been created.

newbatch123 Jobs ⚡ ⭐ ...			
Batch account			
Overview		Activity log	
State == all		Add filter	
ID	State	Pool	Created
JobA	Active	PoolA	Mar 4, 2025, 15:31:31

24. Also, you can see that our table is currently empty.

The screenshot shows a SQL query window titled "SQLQuery1.sql - ap...pdb (sqladmin (68))". The query is:

```

CREATE TABLE [BlobData]
(
    blobname varchar(1000),
    blobsize int,
    accesstier varchar(15)
)

select * from [BlobData]

```

The results pane below shows a table structure with columns: blobname, blobsize, and accesstier. A message at the bottom indicates the query was executed successfully.

25. You can see that our task has been completed. Go to SQL Server Management Studio and execute the select query again.

The screenshot shows the Azure portal interface for a task named "taska" under "JobA". The task status is "Completed". The details pane shows the following information:

- Batch account: newbatch123
- Pool: PoolA
- Node: /tmpfs/9e0c29b4fe4e37e3cd00d0b4ed03db8b8cd140bab329cdb63c1bc97f2e0d...
- Start time: Tuesday, March 4, 2025 at 15:59:18
- End time: n/a

The task status bar indicates it was active on Mar 4, 2025, running, and completed on Mar 4, 2025 (a few seconds ago). The file listing shows three files: "wd", "stderr.txt", and "stdout.txt".

26. You can see that the data has been loaded to the our SQL Table.

Object Explorer

Connect ▾

- appserver4434334.database.windows.net (SQL Server 12.0.2000.8 - sqladmin)
 - Databases
 - System Databases
 - appdb
 - backupdb
 - Security
 - Integration Services Catalogs
- 74.234.75.3 (SQL Server 16.0.4045.3 - sqladmin)
 - Databases
 - System Databases
 - Database Snapshots
 - appdb
 - Security
 - Server Objects
 - Replication
 - Always On High Availability
 - Management
 - Integration Services Catalogs
 - SQL Server Agent (Agent XPs disabled)
 - XEvent Profiler

SQLQuery3.sql - ap...pdb (sqladmin (69))*

```
SELECT * FROM [BlobData]
```

Results Messages

	blobname	blobsize	acessstier
1	02.sql	477	Hot
2	03.sql	290	Hot
3	04.sql	385	Hot
4	05.sql	567	Hot
5	06.sql	296	Hot
6	07.sql	1533	Hot
7	08.sql	707	Hot
8	Log.csv	6964509	Hot
9	python/01.py	265	Hot
10	python/02.py	251	Hot
11	python/03.py	164	Hot
12	python/04.py	73	Hot
13	python/05.py	143	Hot
14	scala/01.scala	48	Hot
15	scala/02.scala	167	Hot
16	scala/03.scala	92	Hot
17	scala/04.scala	107	Hot
18	scala/05.scala	261	Hot
19	scala/06.scala	136	Hot
20	scala/07.scala	426	Hot