..Steps to Create and Monitor Azure Resources..

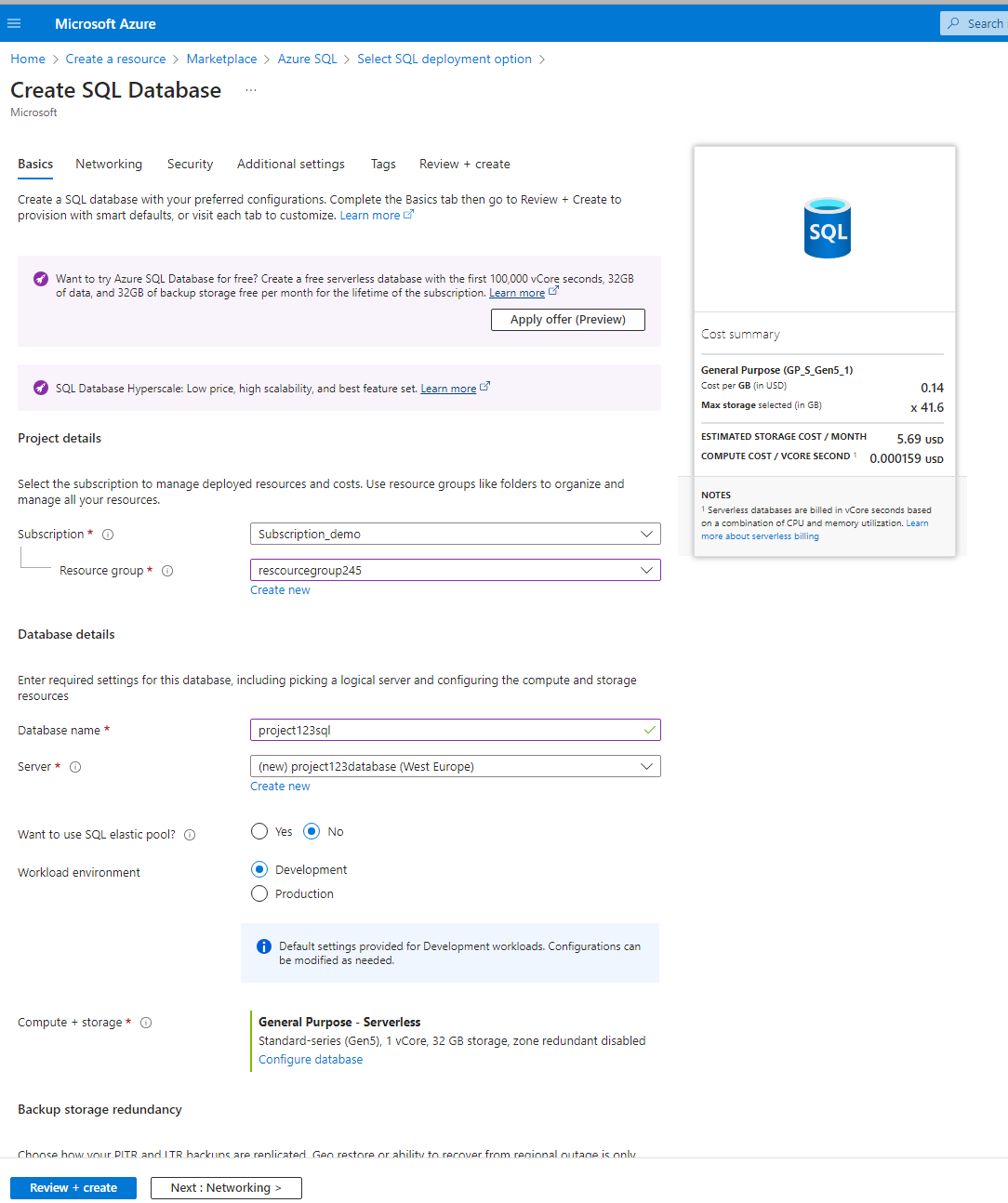
**Creating an Azure SQL Database**

Creating an Azure SQL Database involves several steps, all of which can be done through the Azure Portal. Below is a detailed guide to help you set up your SQL database.

1. In the Azure Portal, click the “Create a resource” button in the homepage's upper left corner. In the new resource page, type “SQL Database’ in the search box and select it from the list of services.

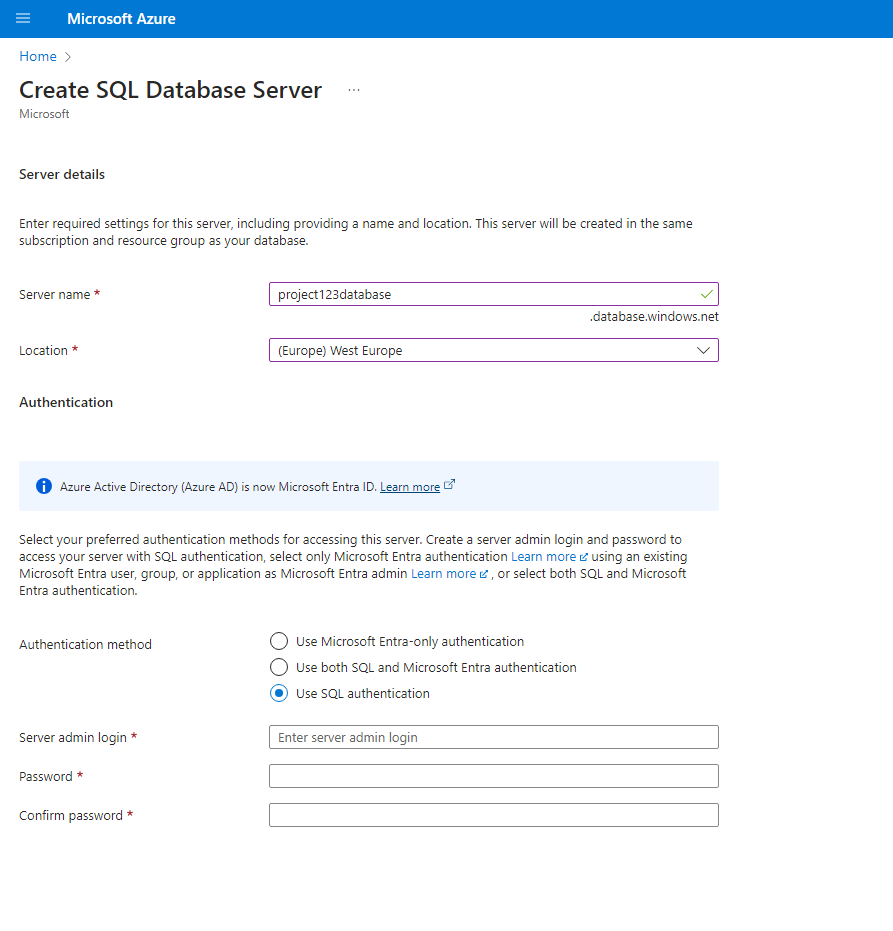
2. Configure the database settings as follows:

* **Subscription:** Choose the appropriate subscription under which you want the database to be billed.
* **Resource group:** Select an existing resource group or create a new one by clicking "Create new" and entering a name.
* **Database name:** Enter a unique name for your new SQL database.



*“Create a SQL database” screen in Azure with a cost summary on the right. Image by author*

3. If you already have a SQL Server, click "Use existing" and select the server from the list. If you do not have an existing SQL Server, click "Create new" to set one up. You’ll need to configure the following settings.



*Creating a new SQL Database Server in Azure. Image by author*

* **Server name:** Enter a unique name for the server.
* **Server admin login:** Create a login name for the server admin.
* **Password:** Create a strong password and confirm it.
* **Location:** Choose the region closest to your users for optimal performance.

4. Next, set up the authentication method and the server admin login:

* **Authentication method:** Choose your preferred authentication method. SQL authentication is common, but you can also use Microsoft Entra ID (previously Azure Active Directory) authentication for added security.
* **Server admin login:** If creating a new server, ensure you have entered the admin login and password as described earlier.

5. Some additional settings to consider:

* **Networking:** Decide whether the database should be accessible over the public internet or only within a private network. Configure the necessary firewall rules and virtual networks.
* **Security:** Enable features like Advanced Data Security, which includes threat detection and vulnerability assessments.
* **Pricing tier:** Choose a tier based on your performance and budget requirements. Options include General Purpose, Business Critical, and Hyperscale tiers. Each tier offers different levels of performance, storage, and features.

6. Finally,review all the configuration settings to ensure everything is set up as desired. Click the "Review + create" button. After the validation process, click "Create" to deploy your SQL database.

If you’re interested in more in-depth knowledge about Azure, check out the [**Azure Architecture and Services course**](https://www.datacamp.com/courses/azure-architecture-and-services).

**Connecting to an Azure SQL Database**

Various tools and methods can be used to connect to an Azure SQL Database. This section covers the primary tools for connection and explains the use of connection strings and authentication methods.

**Tools for connecting to an Azure SQL Database**

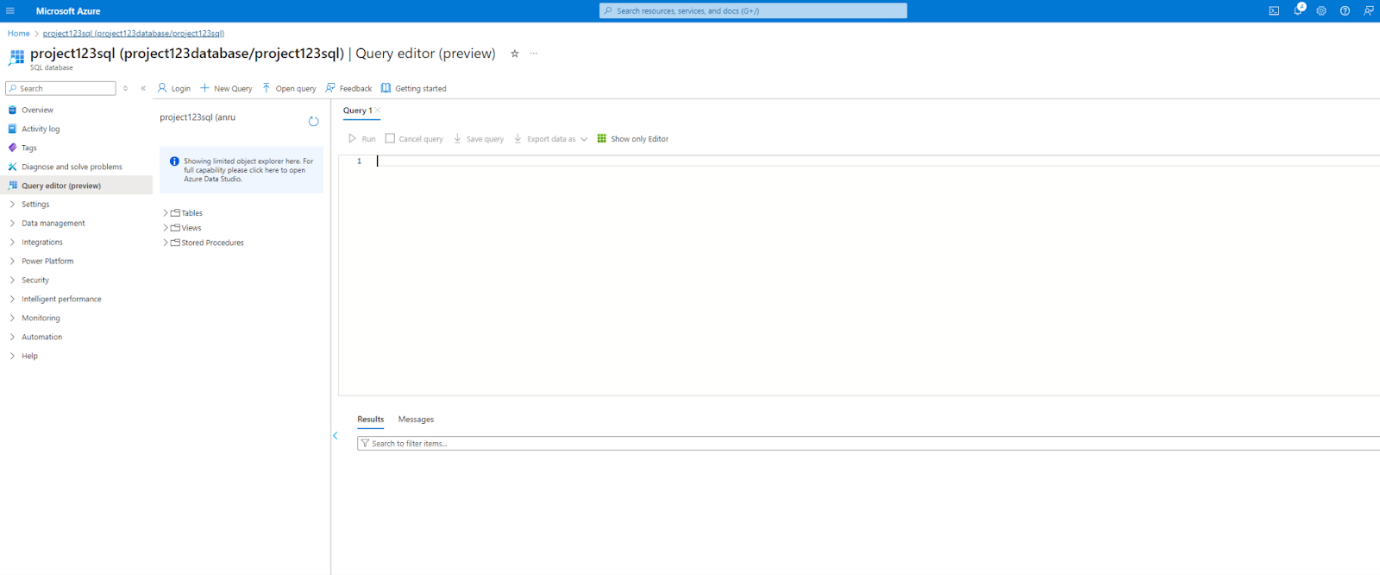
**Connecting via the Azure Portal**

Follow these steps to connect to your SQL database and run queries directly in Azure:

1. Open the Azure Portal and go to your SQL database resource. Click the "Query editor (preview)" in the left-hand menu.

2. Enter the server admin login and password to use SQL authentication, or if configured, use Microsoft Entra ID (previously Azure Active Directory) authentication.

3. You can directly run SQL queries against your database within the Azure Portal’s query editor.



*The Azure Portal query editor. Image by author*

**Connecting via SQL Server Management Studio (SSMS)**

SQL Server Management Studio (SSMS) is a comprehensive, integrated environment for managing SQL Server infrastructure, allowing you to configure, monitor, and administer instances of SQL Server databases.

* **Installation**: Download and install SSMS from the[**official Microsoft website**](https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms).
* **Connecting**: Open SSMS, click "Connect," and select "Database Engine." Enter the server name (e.g., “your\_server\_name.database.windows.net”), the authentication method, and the credentials.

**Connecting via Azure Data Studio**

Azure Data Studio is a cross-platform data management tool for managing SQL Server databases on-premises or in multi-cloud environments.

* **Installation**: Download and install Azure Data Studio from the[**official Microsoft website**](https://docs.microsoft.com/en-us/sql/azure-data-studio/download-azure-data-studio).
* **Connecting**: Open Azure Data Studio, click "New Connection," and enter the server details, including the server name, database name, and authentication details.

**Other tools to connect to your Azure SQL Database**

* **Visual Studio Code**: Install the SQL Server extension for Visual Studio Code, which provides the ability to connect to Azure SQL Databases.
* [**Power BI**](https://www.datacamp.com/courses/introduction-to-power-bi)**:**Connect to Azure SQL Database directly from Power BI Desktop for advanced data analysis and visualization.

**Connection strings and authentication**

**Understanding connection strings**

Connection strings are essential for applications to connect to your Azure SQL Database. They contain the server name, database name, authentication credentials, and other parameters.

Example connection string for SQL authentication:

Server=tcp:your\_server\_name.database.windows.net,1433;Initial Catalog=your\_database\_name;Persist Security Info=False;User ID=your\_username;Password=your\_password;MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;

**Setting up SQL authentication**

In SSMS or Azure Data Studio, connect to your Azure SQL Database and run SQL scripts to create new users and assign roles. Include the username and password in your connection string for authentication.

CREATE LOGIN mylogin WITH PASSWORD = 'StrongPassword!';

CREATE USER myuser FOR LOGIN mylogin;

ALTER ROLE db\_datareader ADD MEMBER myuser;

**Setting up Microsoft Entra ID (previously Azure Active Directory) authentication**

Follow these steps to configure the Entra ID admin and assign users/roles:

1. In the Azure Portal, navigate to your SQL Server resource and select "Active Directory admin" to set an Entra ID user or group as the administrator.

2. Use SQL scripts to create users and assign them roles. This requires SSMS or Azure Data Studio with Entra ID authentication enabled. For example:

CREATE USER [aad\_user@domain.com] FROM EXTERNAL PROVIDER;

ALTER ROLE db\_datareader ADD MEMBER [aad\_user@domain.com];

3. Modify your connection string to include Entra ID credentials. This often involves setting up token-based authentication:

Server=tcp:your\_server\_name.database.windows.net,1433;Initial Catalog=your\_database\_name;Persist Security Info=False;User ID=aad\_user@domain.com;Authentication="Active Directory Password";Password=your\_password;MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;

**Advanced database management features**

Now, let’s look at more sophisticated features you can use in your Azure SQL Database, particularly in production environments.

**Scaling databases with elastic pools**

Elastic pools in Azure SQL Database enable you to manage and scale multiple databases with varying and unpredictable usage demands. They provide cost efficiency by sharing resources among databases within the pool. Follow these steps to create an elastic pool:

1. Navigate to the Azure Portal, select your SQL Database server, and click "Create new" under Elastic Pool.

2. Configure the pool by setting the number of DTUs (Database Transaction Units) and adding databases.

**Implementing high availability and disaster recovery solutions**

Azure SQL Database provides several high availability and disaster recovery options to ensure your database remains operational and data is protected:

* **Built-in high availability:** There is built-in high availability with a 99.99% uptime SLA. This is achieved through data replication across multiple nodes in different physical locations.
* **Geo-replication:** You can enable geo-replication to create readable secondary replicas in different Azure regions. This allows for failover to a secondary region in a regional outage. To do so, navigate to your SQL database from the Azure Portal, select “Geo-Replication,” and configure the secondary replicas.
* **Automatic back-ups:** Azure SQL Database automatically performs full, differential, and transaction log backups. You can configure backup retention policies to meet your recovery point objectives.
* **Point-in-time restore:** In case of accidental data loss or corruption, you can restore the database to any point within the retention period. You can do this directly from the Azure Portal by navigating to your database and selecting the "Restore" option.

**Securing Your Azure SQL Database**

When working with data, security is of utmost importance. This section discusses the main security features of the Azure SQL Database service.

**Security best practices**

To help protect your Azure SQL Database from unauthorized access and ensure secure, private connectivity within your Azure environment, you can implement these best practices:

* **Configure IP firewall rules** to limit access to the database from specific IP addresses or ranges. This minimizes the risk of unauthorized access by restricting database connectivity to known, trusted sources.
* **Use virtual network service endpoints** to restrict database access to specific virtual networks. This allows you to lock down access to the Azure SQL Database from specific VNets, enhancing security by limiting exposure to the public internet.
* **Integrate the VNet with the SQL database** using service or private endpoints for secure connectivity. This reduces the risk of interception or attack and exposure to the public Internet.

**Using advanced data security features**

To help protect your data in Azure SQL databases, consider these advanced configurations:

* **Advanced threat protection** is used to detect unusual activities.
* **Transparent data encryption (TDE)** encrypts data at rest.
* **Dynamic data masking** hides sensitive data in query results.

**Regularly auditing and monitoring your database**

To further support your SQL database's security and data protection, you can set up auditing processes to track database activities and store logs in an Azure Storage account, Log Analytics workspace, or Event Hub.

You can also use [**Azure Monitor**](https://www.datacamp.com/tutorial/getting-started-with-azure-monitor) to track performance and security metrics and set up alerts for specific conditions.

If you’d like to learn more about data governance in Azure, check out DataCamp’s [**Azure Management and Governance**](https://www.datacamp.com/courses/azure-management-and-governance) course.