

Computer Science

Ken Gamradt

Docker SQL Server

02-22-24

SQL Server – Microsoft Artifact Registry

The screenshot shows a web browser window displaying the Microsoft Artifact Registry page for the Microsoft SQL Server - Ubuntu based images. The page has a blue header with the Microsoft logo and the text "Microsoft Artifact Registry". Below the header, there's a large image of a blue cylinder with the word "SQL" on it and a gear icon. To the left of the image, there are several stats: "Last Modified" (02/15/2024), "Pulls" (46.1M+), "Categories" (Databases), "Project" (links to Project Website and License), and "Help" (links to Documentation and Support). The main content area has tabs for "About" (which is selected) and "Tags". The "About" tab contains the text "Official container images for Microsoft SQL Server- Ubuntu based for Docker Engine". Below this, there's a section titled "Featured Tags" with three items: "2022-latest" (with the command "docker pull mcr.microsoft.com/mssql/server:2022-latest"), "2019-latest" (with the command "docker pull mcr.microsoft.com/mssql/server:2019-latest"), and "2017-latest" (with the command "docker pull mcr.microsoft.com/mssql/server:2017-latest"). At the bottom, there's a section titled "Related Repositories" with a note about the Beta program being suspended.

Microsoft Artifact Registry

Back to catalog

 Microsoft SQL Server - Ubuntu based images

Microsoft

Last Modified

02/15/2024

Pulls

46.1M+

Categories

Databases

Project

[Project Website](#)

[License](#)

Help

[Documentation](#)

[Support](#)

About

Tags

About

Official container images for Microsoft SQL Server- Ubuntu based for Docker Engine

Featured Tags

- 2022-latest
docker pull mcr.microsoft.com/mssql/server:2022-latest
- 2019-latest
docker pull mcr.microsoft.com/mssql/server:2019-latest
- 2017-latest
docker pull mcr.microsoft.com/mssql/server:2017-latest

Related Repositories

The Beta program for SQL Server Windows Container was suspended in the year 2021 as updated here: [Update on Beta program for SQL Server on Windows Container](#), thus it is not supported for Production workload. For development purpose, if you still need references for a custom SQL Server container image on

Database Software

SQL Server

- [SQL Server](#) is a database management system (DBMS) that is used to store and manage data
- The Developer edition is a full-featured free edition

Azure Data Studio

- [Azure Data Studio](#) is a free, open-source graphical tool for managing and administering SQL Server and Azure databases
- It allows you to easily create and maintain databases, run SQL queries, and view and edit data

Required Software

docker

- [Docker](#) is a tool designed to make it easier to create, deploy, and run applications
- It does this by using containers, which are essentially self-contained environments that contain all of the necessary parts for an application to run, such as libraries, dependencies, and runtime
- Docker helps to simplify the process of building and deploying applications

Install docker

- First, make sure you have Docker installed on your computer
- If you don't have it, you can download it from the Docker [website](#)

Installation

Run the docker compose file

- A docker compose file is a way to define and run multi-container Docker applications
- It's like a blueprint for your application, telling Docker what images to use and how to connect them together
- This will save time and hassle because you can set everything up in one place and start everything with a single command
- It's written in yaml format and has sections for defining the version and the different services that make up the application
- Each service can be customized with options like images, environment variables, volumes, ports, and links to other services

docker-compose.yaml (.yml) file

```
services:  
  mssql:  
    container_name: mssql-db  
    hostname: mssql-db  
    image: mcr.microsoft.com/mssql/server:2022-latest  
    environment:  
      ACCEPT_EULA: 'Y'                      # required  
      MSSQL_PID: 'Developer'                 # required – default  
      # MSSQL_SA_USER: 'sa'  
      MSSQL_SA_PASSWORD: 'Pa55w0rd'        # required  
      MSSQL_TCP_PORT: 1433                  # default  
      MSSQL_DATA_DIR: /var/opt/mssql/data  
    ports:  
      - "1433:1433"  
    restart: always  
    volumes:  
      - ./data:/var/opt/mssql/data  
      - ./log:/var/opt/mssql/log  
      - ./secrets:/var/opt/mssql/secrets
```

docker-compose.yaml (.yml) file

Version

- Specifies the version of the docker compose file format

Services

- Defines the services (i.e., containers) that make up the application
- In this case, we have one service: mssql

Images

- Specifies the docker image to be used for the container
- For the mssql service, we are using the following image
 - mcr.microsoft.com/mssql/server:2022-latest

docker-compose.yaml (.yml) file

Environments

- Sets environment variables for the container
- In the postgres service, we are setting the following variables used to configure the SQL Server database
 - MSSQL_SA_USER
 - MSSQL_SA_PASSWORD
 - ...

Volumes

- Mounts a volume (i.e., persistent storage) for the container
- In the mssql service, we are mounting volumes to store SQL Server data
 - data, log, secrets
- This allows the data to persist even if the container is stopped or removed

`docker-compose.yaml (.yml)` file

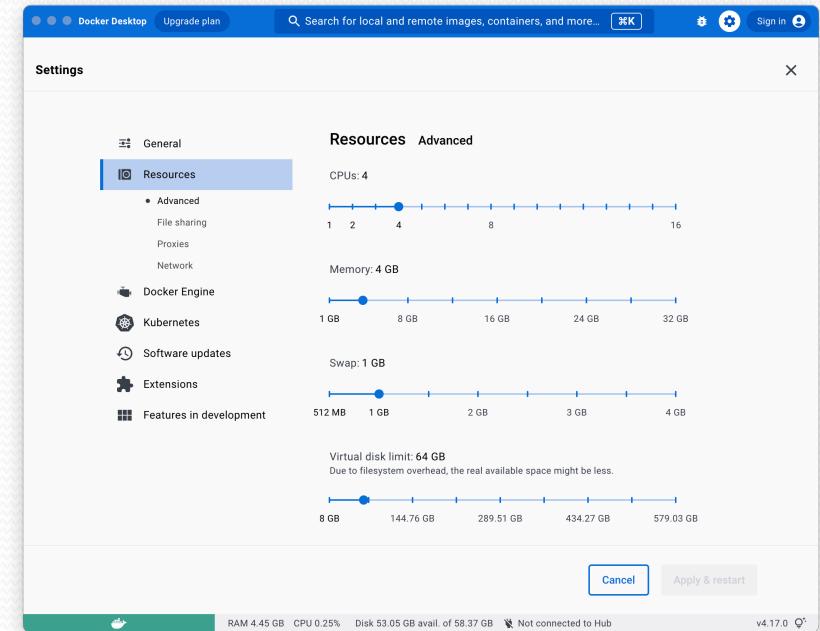
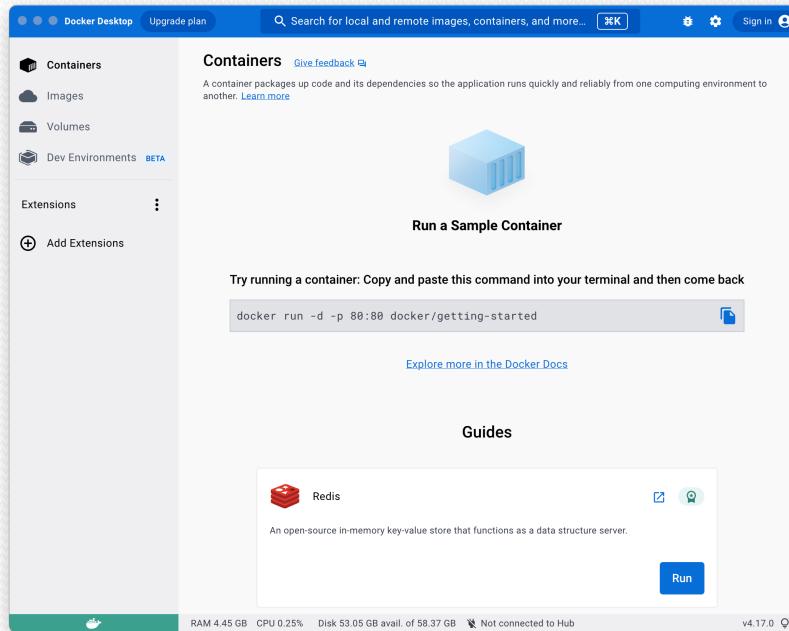
Ports

- Exposes a container's port to the host
- In the postgres service, we are exposing the default SQL Server port (1433) on the host

Depends on (not used here)

- Specifies that this service requires other services to already be running

Setting up Docker and SQL Server



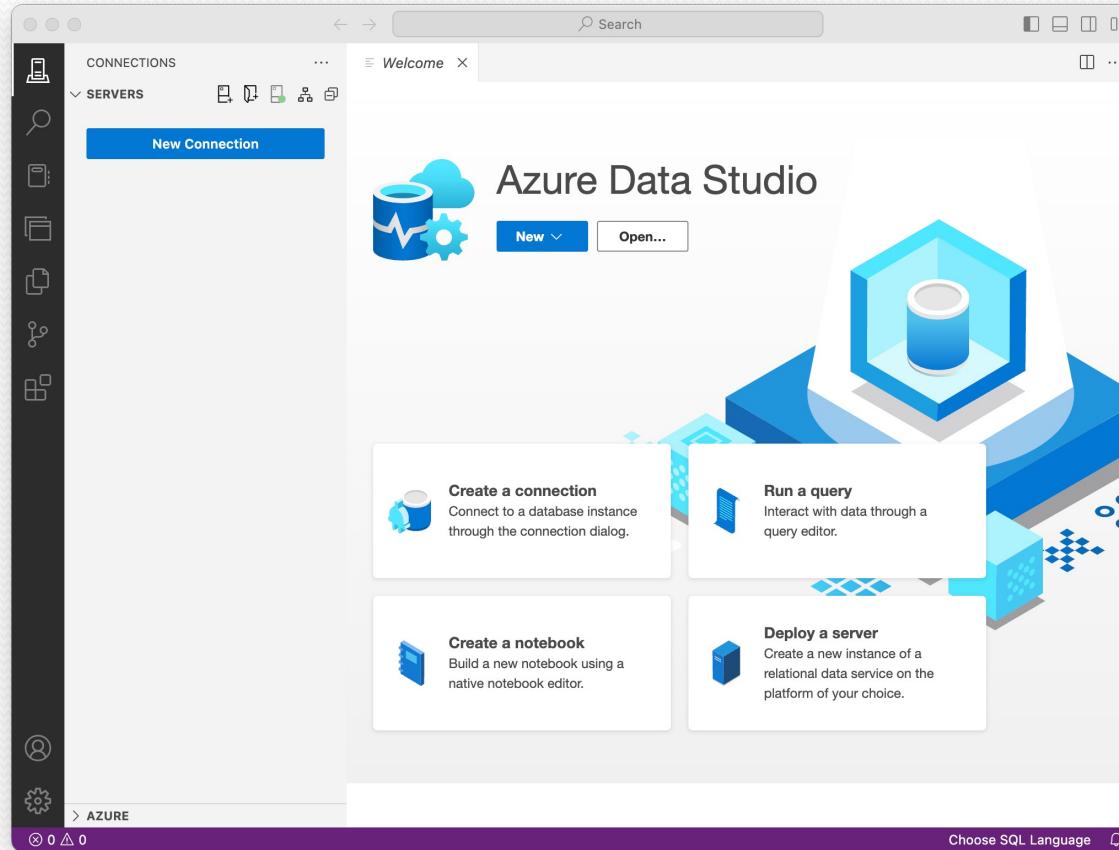
Setting up Docker and SQL Server

```
ken@sdeh136-08c4af SQLServer % pwd
/Users/ken/Documents/Databases/SQLServer
ken@sdeh136-08c4af SQLServer % ls
docker-compose.yaml
ken@sdeh136-08c4af SQLServer % docker compose up -d
[+] Running 4/4
✓ mssql 3 layers [■■■]    0B/0B      Pulled          42.7s
  ✓ 25fa6962a0ca Pull complete                         3.9s
  ✓ 22fbfac82290 Pull complete                         29.9s
  ✓ f57446ead6b1 Pull complete                         10.4s
[+] Running 1/2
  :: Network sqlserver_default   Created          0.5s
  ✓ Container mssql-db          Started          0.5s
ken@sdeh136-08c4af SQLServer % ls
data           log
docker-compose.yaml secrets
ken@sdeh136-08c4af SQLServer % █
```

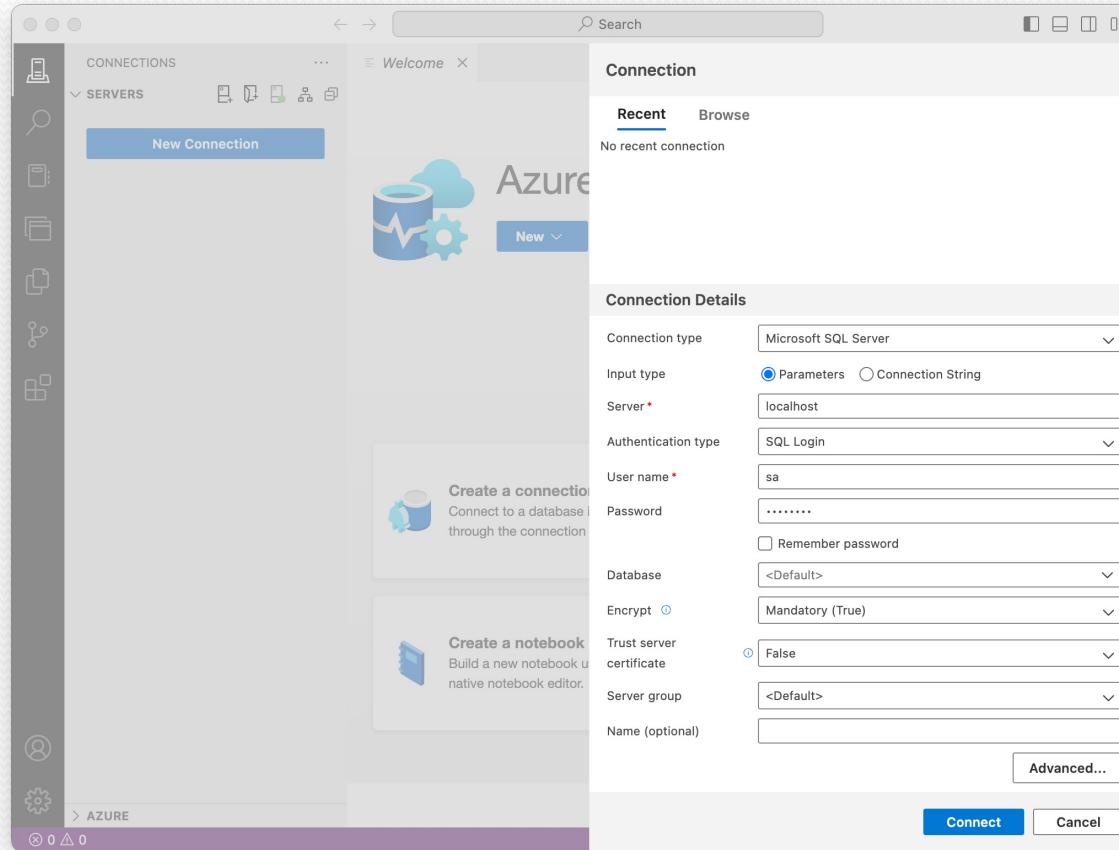
Setting up Docker and SQL Server

```
ken@sdeh136-08c4af SQLServer % docker image ls -a
REPOSITORY          TAG      IMAGE ID      CREATED        SIZE
mcr.microsoft.com/mssql/server    2022-latest   ffdd6981a89e   3 months ago   1.5
8GB
ken@sdeh136-08c4af SQLServer % docker container ls -a
CONTAINER ID        IMAGE               COMMAND
CREATED            STATUS              PORTS
...                NAMES
fa920d5a56dd   mcr.microsoft.com/mssql/server:2022-latest   "/opt/mssql/bin/perm
..."   About a minute ago   Up About a minute   0.0.0.0:1433->1433/tcp   mssql-db
ken@sdeh136-08c4af SQLServer %
```

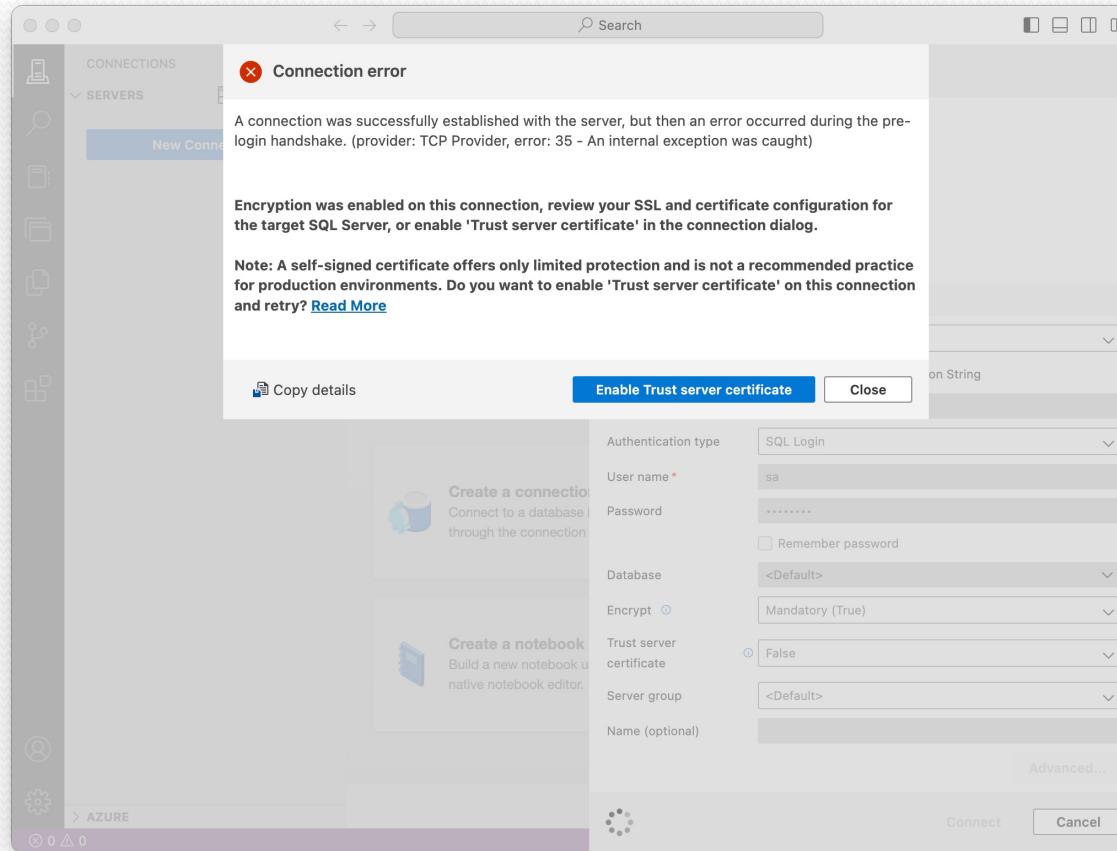
Setting up Azure Data Studio and SQL Server



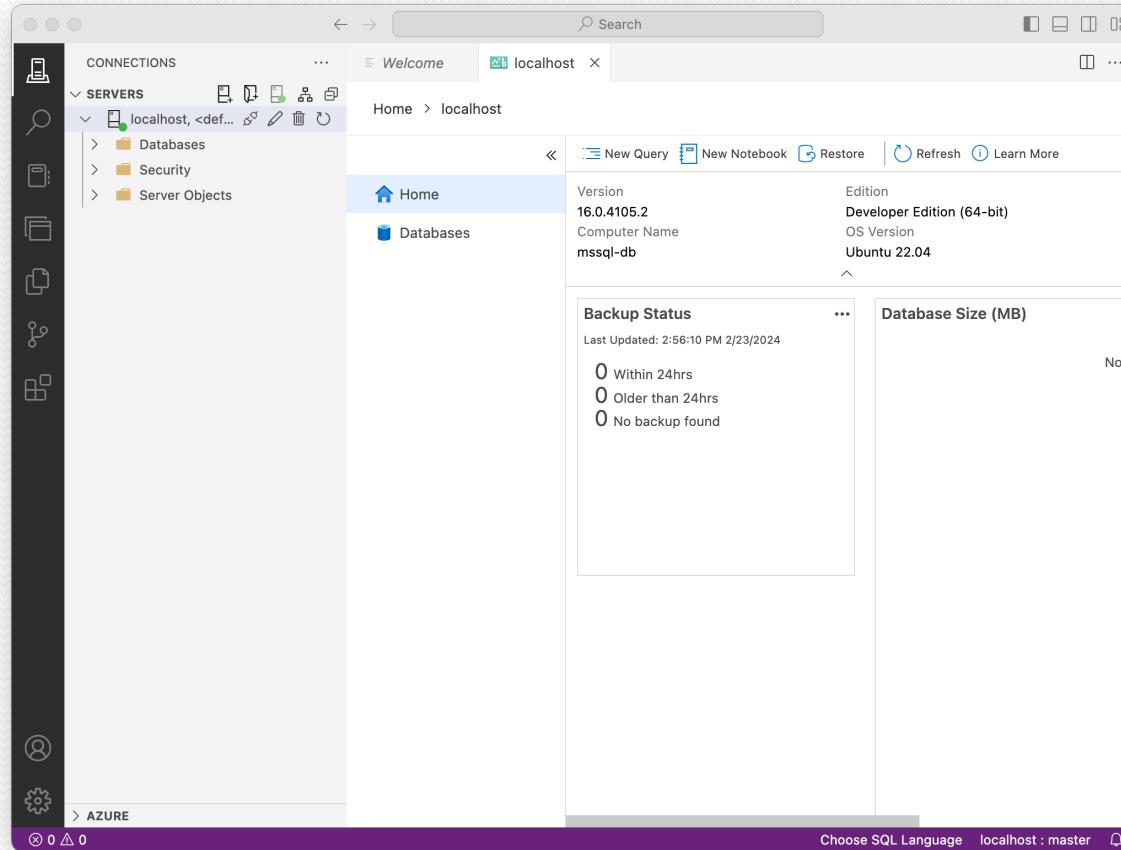
Setting up Azure Data Studio and SQL Server



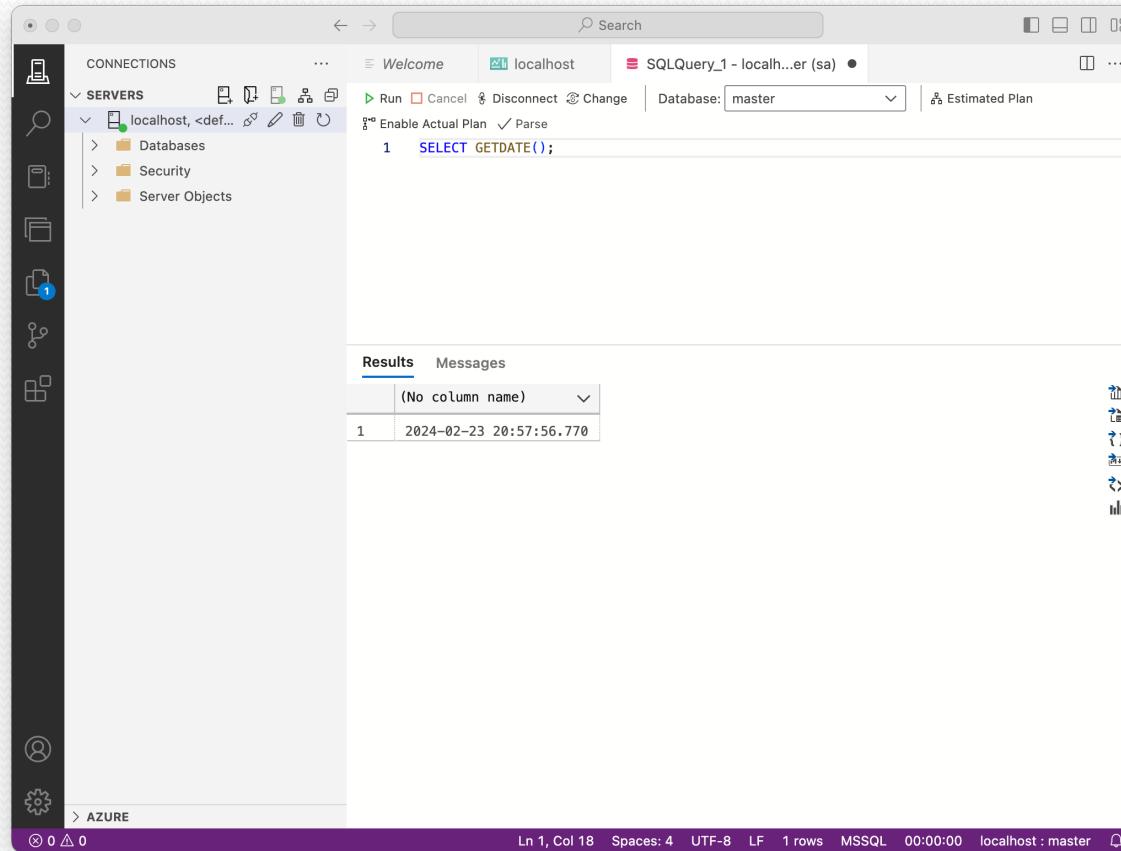
Setting up Azure Data Studio and MS SQL



Setting up Azure Data Studio and MS SQL



Setting up Azure Data Studio and MS SQL



Database Information

```
CREATE DATABASE DatabaseAPI
GO

USE DatabaseAPI
GO

CREATE SCHEMA SchemaAPI
GO

CREATE TABLE SchemaAPI.[User](
    UserId INT IDENTITY(1, 1) PRIMARY KEY,
    FirstName NVARCHAR(50),
    LastName NVARCHAR(50),
    Email NVARCHAR(50),
    Gender NVARCHAR(50),
    Active BIT
);
```

- Connection String:

```
"Server=localhost;
Database=DatabaseAPI;
Trusted_Connection=false;
TrustedServerCertificate=true;
User Id=sa;
Password=Pa55w0rd;"
```

- All one string – 1 line

Database Creation

The screenshot shows the SQL Server Management Studio (SSMS) interface. On the left, the Object Explorer tree view displays the following structure:

- CONNECTIONS**: A list of available connections.
- SERVERS**:
 - localhost, <default> (sa)**: The selected connection.
 - Databases**: A folder containing:
 - System Databases**
 - DatabaseAPI**: The currently selected database.
- AZURE**: A section for Azure integration.

The **DatabaseAPI** node is expanded, showing:

- Tables**:
 - SchemaAPI.User**: Expanded, showing columns: UserId, FirstName, LastName, Email, Gender, Active.
 - Keys**:
 - PK_User_1788CC4C1CA40F99**: Primary key constraint.
 - Constraints**
 - Triggers**
 - Indexes**
 - Statistics**
 - Dropped Ledger Tables**
- Views**
- Synonyms**
- Programmability**
- External Resources**

The **SQLQuery_1** window on the right contains the T-SQL script for creating the database and its schema:

```
1 CREATE DATABASE DatabaseAPI
2 GO
3
4 USE DatabaseAPI
5 GO
6
7 CREATE SCHEMA SchemaAPI
8 GO
9
10 CREATE TABLE SchemaAPI.[User](
11     UserId INT IDENTITY(1, 1) PRIMARY KEY,
12     FirstName NVARCHAR(50),
13     LastName NVARCHAR(50),
14     Email NVARCHAR(50),
15     Gender NVARCHAR(50),
16     Active BIT
17 );
18
```

The **Messages** pane at the bottom shows the execution log:

Time	Message
1:27:32 PM	Started executing query at Line 1 Commands completed successfully.
1:27:32 PM	Started executing query at Line 3 Commands completed successfully.
1:27:32 PM	Started executing query at Line 6 Commands completed successfully.
1:27:32 PM	Started executing query at Line 9 Commands completed successfully. Total execution time: 00:00:00.302

At the bottom of the interface, the status bar shows: Ln 20, Col 1 | Spaces: 4 | UTF-8 | LF | 0 rows | MSSQL | 00:00:00 | localhost : DatabaseAPI | [Help](#).

Database Testing – after seeding

The screenshot shows the SSMS interface with the following details:

- Connections:** A tree view on the left shows connections to localhost, including the default (sa) user, system databases, and the DotNetCourseDatabase.
- Current Connection:** The connection to localhost:DatabaseAPI is active.
- Query Bar:** The top bar includes buttons for Run, Cancel, Disconnect, Change, Parse, and Database dropdowns set to DotNetCourseDatabase.
- Results Tab:** The main pane displays the results of a query against the Users table. The table has columns: UserId, FirstName, LastName, Email, Gender, and Active. The data consists of 20 rows of test users.
- Messages Tab:** This tab is currently inactive.
- Toolbar:** On the right side, there is a toolbar with various icons for managing objects like tables, keys, constraints, triggers, indexes, statistics, and views.
- Status Bar:** At the bottom, it shows the number of selected rows (146), spaces used (4), character encoding (UTF-8), line feed (LF), row count (1,000), and the connection information (localhost : DotNetCourseDatabase).

UserId	FirstName	LastName	Email	Gender	Active
1	Albertina	O'Finan	aofinan0@blogspot.com	Female	0
2	Franky	Laidler	flaidler1@over-blog...	Bigender	1
3	Farica	Wynn	fwynn2@redcross.org	Female	0
4	Bordie	Rodda	brodda3@columbia.edu	Female	1
5	Charley	Pack	cpack4@answers.com	Female	0
6	Morganne	Clother	mclother5@nifty.com	Male	0
7	Arty	Gillott	agillott6@barnesandn...	Male	0
8	Thorvald	Vines	tvines7@cloudflare.c...	Male	1
9	Hilarius	Nicholl	hnicholl8@amazonaws...	Female	1
10	Enriqueta	Tilston	etilston9@cbc.ca	Female	0
11	Symon	Oakland	soaklanda@flickr.com	Female	1
12	Violet	Perkis	vperkisb@typepad.com	Male	1
13	Stephine	Paschke	spaschke@chronoengi...	Male	0
14	Salvatore	Sprouls	ssproulsd@businessin...	Female	1
15	Liuka	Di Meo	ldimeoe@economist.com	Female	1
16	Rubina	Griss	rgrissff@fema.gov	Female	1
17	Aloysia	Ledington	aledingtong@bandcamp...	Male	0
18	Daria	Suddock	dsuddockh@wired.com	Male	1
19	Gaylord	Moreman	gmoremani@godaddy.com	Female	0
20	Brina	Cronkshaw	bcronkshawj@webeden...	Male	1

Acknowledgements

- docker – Docker Desktop
 - <https://www.docker.com/products/docker-desktop/>
- docker docs – Docker Compose
 - <https://docs.docker.com/compose/>
- dockerhub
 - <https://hub.docker.com/>
- SQL Server
 - <https://www.microsoft.com/en-us/sql-server/sql-server-downloads>
- Azure Data Studio
 - <https://azure.microsoft.com/en-us/products/data-studio>