Tutorial-Install SQL Server on Mac

Here you'll find how to get SQL Server up and running on your Mac and you'll have SQL Server running locally without needing any virtualization software.

Prior to SQL Server 2017, if you wanted to run SQL Server on your Mac, you first had to create a virtual machine (using VirtualBox, Parallels Desktop, VMware Fusion, or Bootcamp), then install Windows onto that VM, then finally SQL Server. This is still a valid option depending on your requirements (here's how to install SQL Server on a Mac with VirtualBox if you'd like to try that method) (Also you can find this option in pre-class documents). But we do not recommend this option due to various reasons.

Starting with SQL Server 2017, you can now install SQL Server directly on to a Linux machine. And because macOS is Unix based (and Linux is Unix based), you can run SQL Server for Linux on your Mac. The way to do this is to run SQL Server on Docker.

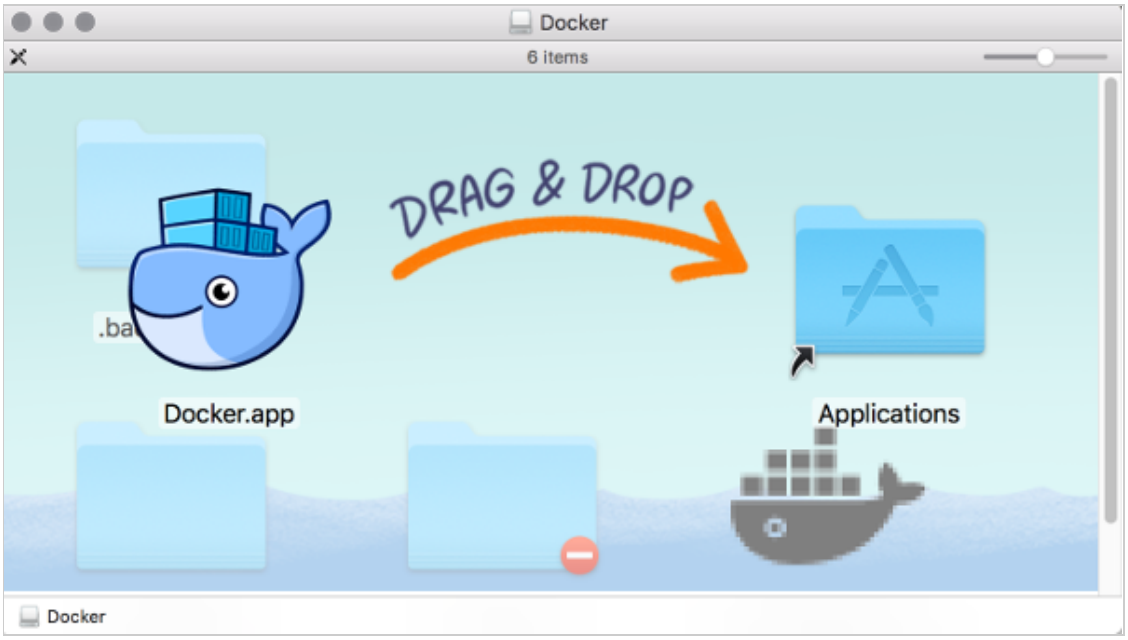
So let's go ahead and install Docker. Then we'll download and install SQL Server.

1. Install Docker

Download the (free) Docker Community Edition for Mac (unless you've already got it installed on your system). This will enable you to run SQL Server from within a Docker container.

To download, visit the Docker CE for Mac download page and click `Get Docker`.

To install, double-click on the `.dmg` file and then drag the `Docker.app` icon to your `Applications` folder.



Docker installation on a Mac.

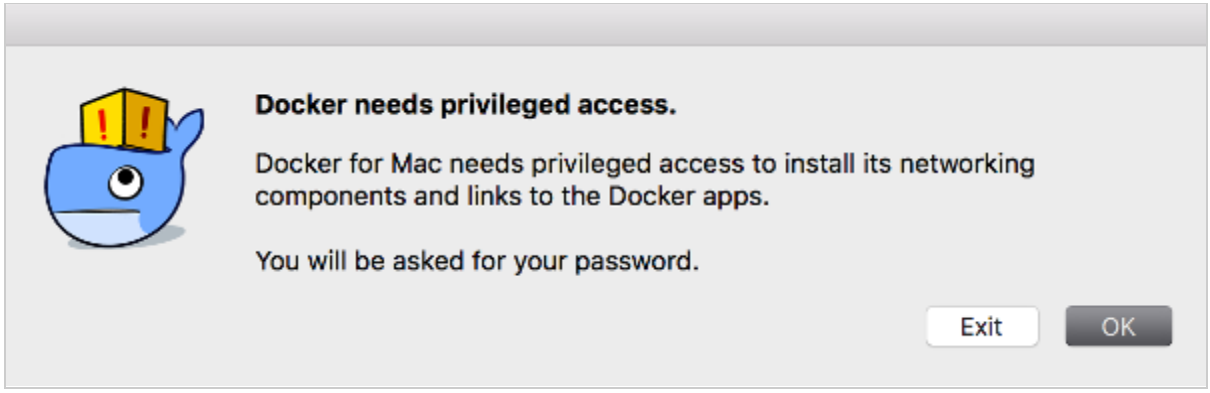
"***What is Docker?***

Docker is a platform that enables software to run in its own isolated environment. SQL Server (from 2017) can be run on Docker in its own isolated container. Once Docker is installed, you simply download — or "pull" — the SQL Server on Linux Docker Image to your Mac, then run it as a Docker container. This container is an isolated environment that contains everything SQL Server needs to run."

2. Launch Docker

Launch Docker the same way you'd launch any other application (eg, via the Applications folder, the Launchpad, etc).

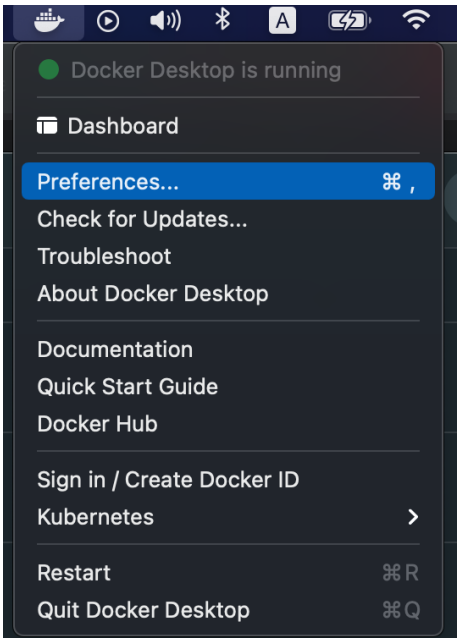
When you open Docker, you might be prompted for your password so that Docker can install its networking components and links to the Docker apps. Go ahead and provide your password, as Docker needs this to run.



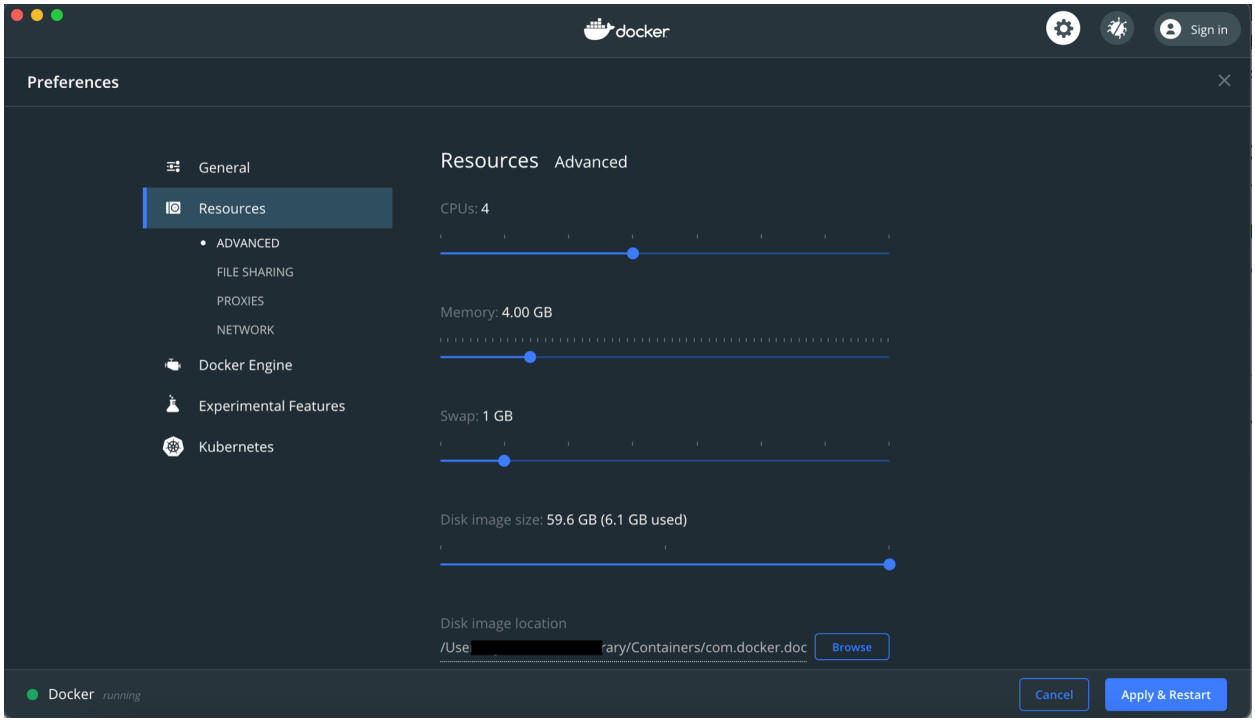
The password request dialog

3. Increase the Memory (optional)

By default, Docker will have 2GB of memory allocated to it. SQL Server needs at least 2GB. However, it won't hurt to increase it if you can.
In my case, I increased it to 4GB.
To do this, select `Preferences` from the little Docker icon in the top menu:



Then on the `Resources` > `Advanced` screen, slide the memory slider up to at least 4GB:



Then finish off by clicking `Apply & Restart`

4. Download SQL Server

Now that Docker is installed, we can download and install SQL Server for Linux.

Open a Terminal window and run the following command.

```
sudo docker pull mcr.microsoft.com/mssql/server:2017-latest
```

This downloads the latest SQL Server 2017 for Linux Docker image to your computer. You can also check for the latest container version on the Docker website if you wish.

Terminal may ask you to enter your password here. Then it will start downloading which will take some time.

5. Launch the Docker Image

Run the following command to launch an instance of the Docker image you just downloaded (But of course, use your own name and password):

```
docker run -d --name sql_server_demo -e 'ACCEPT_EULA=Y' -e 'SA_PASSWORD=reallyStrongPwd123' -p 1433:1433 mcr.microsoft.com/mssql/server:2017-latest
```

*****An important note, if the command above fails on Docker (quits immediately after start), try with using double quotes (“”)*****

```
docker run -d --name sql_server_demo -e “ACCEPT_EULA=Y” -e “SA_PASSWORD=reallyStrongPwd123” -p 1433:1433 mcr.microsoft.com/mssql/server:2017-latest
```

6. Also, if you downloaded a different Docker image, replace `mcr.microsoft.com/mssql/server:2017-latest` with the one you downloaded.
   Here’s an explanation of the parameters:

`-d`	This optional parameter launches the Docker container in daemon mode. This means that it runs in the background and doesn’t need its own Terminal window open. You can omit this parameter to have the container run in its own Terminal window.
`--name sql_server_demo`	Another optional parameter. This parameter allows you to name the container. This can be handy when stopping and starting your container from the Terminal.
`-e 'ACCEPT_EULA=Y'`	The `Y` shows that you agree with the EULA (End User Licence Agreement). This is required in order to have SQL Server for Linux run on your Mac.
`-e 'SA_PASSWORD=reallyStrongPwd123'`	Required parameter that sets the `sa` database password.

`-p 1433:1433`	This maps the local port 1433 to port 1433 on the container. This is the default TCP port that SQL Server uses to listen for connections.
`mcr.microsoft.com/mssql/server:2017-latest`	This tells Docker which image to use. If you downloaded a different one, use it instead.

Password Strength

If you get the following error at this step, try again, but with a stronger password.

```
Microsoft(R) SQL Server(R) setup failed with error code 1. Please check the setup log in /var/opt/mssql/log for more information.
```

Check the Docker container (optional)

You can type the following command to check that the Docker container is running.

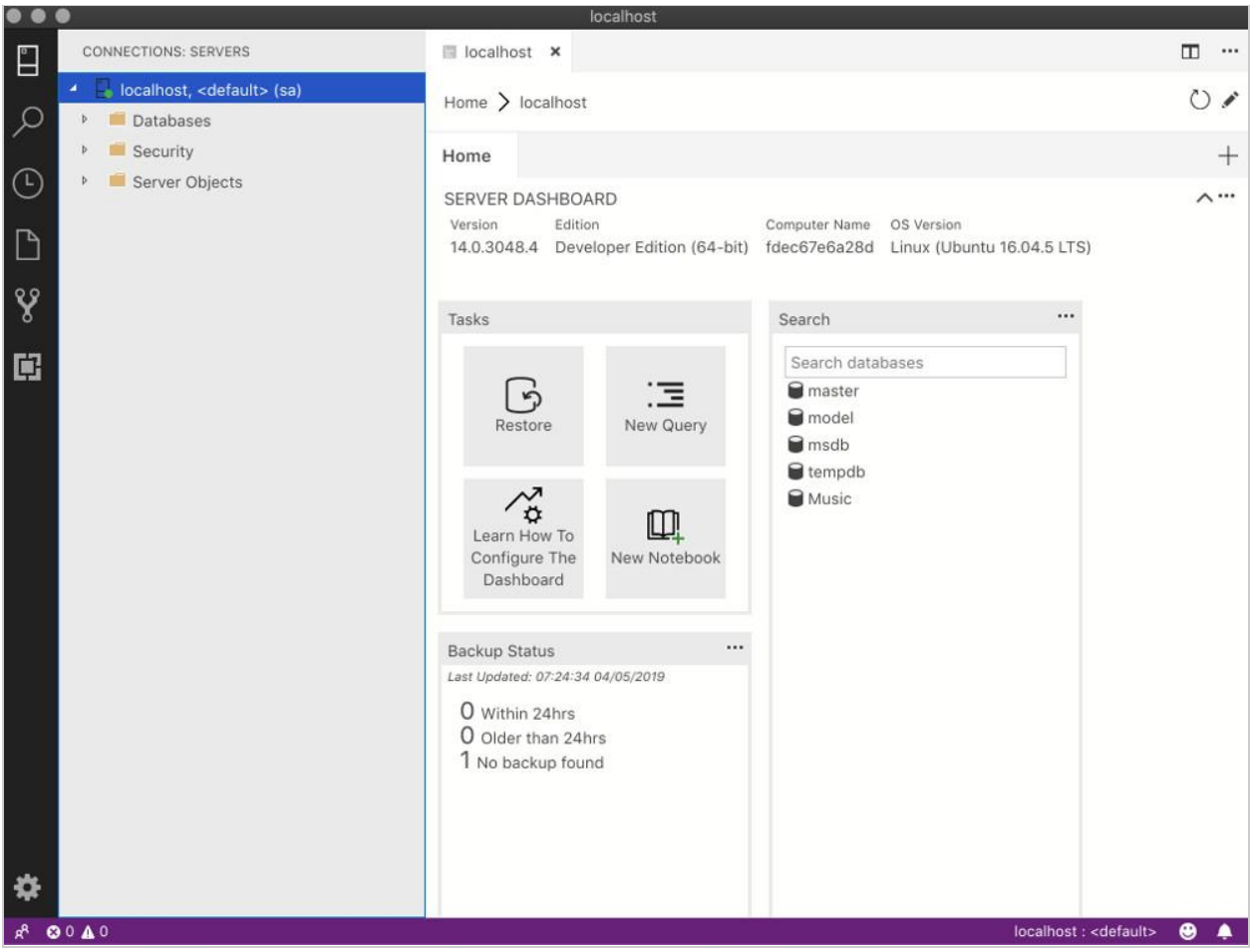
```
docker ps
```

If it's up and running, it should return something like this:

```
CONTAINER ID        IMAGE
COMMAND                  CREATED              STATUS
PORTS                    NAMES
4e4aa21eb391
mcr.microsoft.com/mssql/server:2017-latest
"/opt/mssql/bin/sqls…" 23 seconds ago      Up 21 seconds
0.0.0.0:1433->1433/tcp   sql_server_demo
```

7. Connecting to SQL Server (Method 1-GUI)

An SQL Server GUI for your Mac – Azure Data Studio



The Azure Data Studio dashboard.

Azure Data Studio (previously known as *SQL Operations Studio*) is a free tool that you can use to manage SQL Server. It uses a graphical user interface (GUI) that helps you view the various databases and objects within a SQL Server instance. It can run on Windows, macOS, and Linux, and it's also designed to be used with Azure SQL Database, and Azure SQL Data Warehouse.

Here I explain how to install Azure Data Studio onto a Mac, then how to use it to connect to SQL Server.

Install Azure Data Studio

To install Azure Data Studio onto your Mac:

1. Visit the Azure Data Studio download page, and click the .zip file for macOS
2. Once the .zip file has finished downloading, double click it to expand its contents

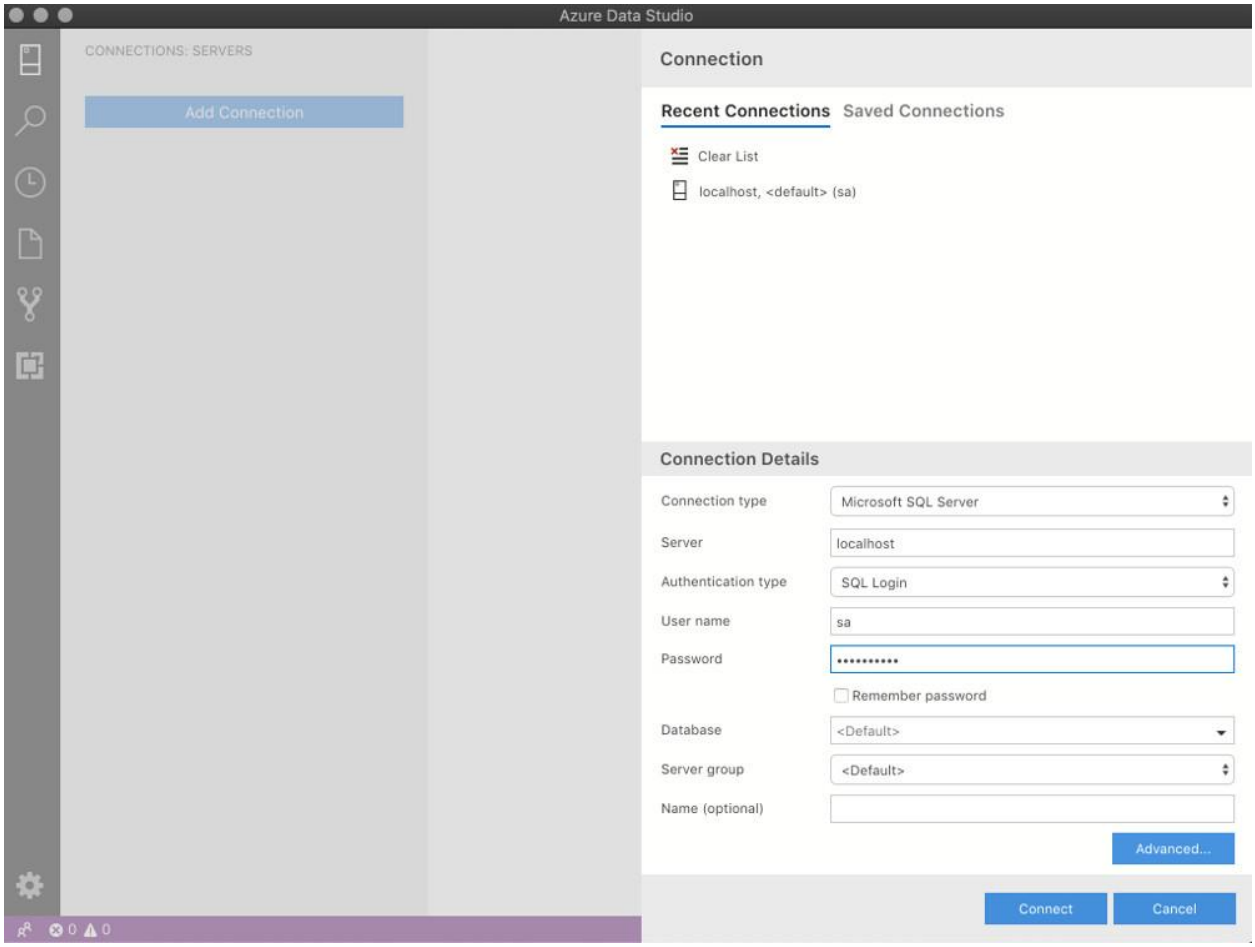
3. Drag the .app file to the Applications folder (the file will probably be called *Azure Data Studio.app*)

If you use Windows or Linux, the above linked page also includes download files for those platforms, as well as instructions for installing.

Connect to SQL Server

Now that Azure Data Studio is installed, you can use it to connect to SQL Server.

1. Launch Azure Data Studio just as you would launch any other application (e.g. from the Launchpad or Applications folder)
2. Then click to “Create a connection” section and enter the login credentials and other information for the SQL Server instance that you’d like to connect to:



Mine looked like this :

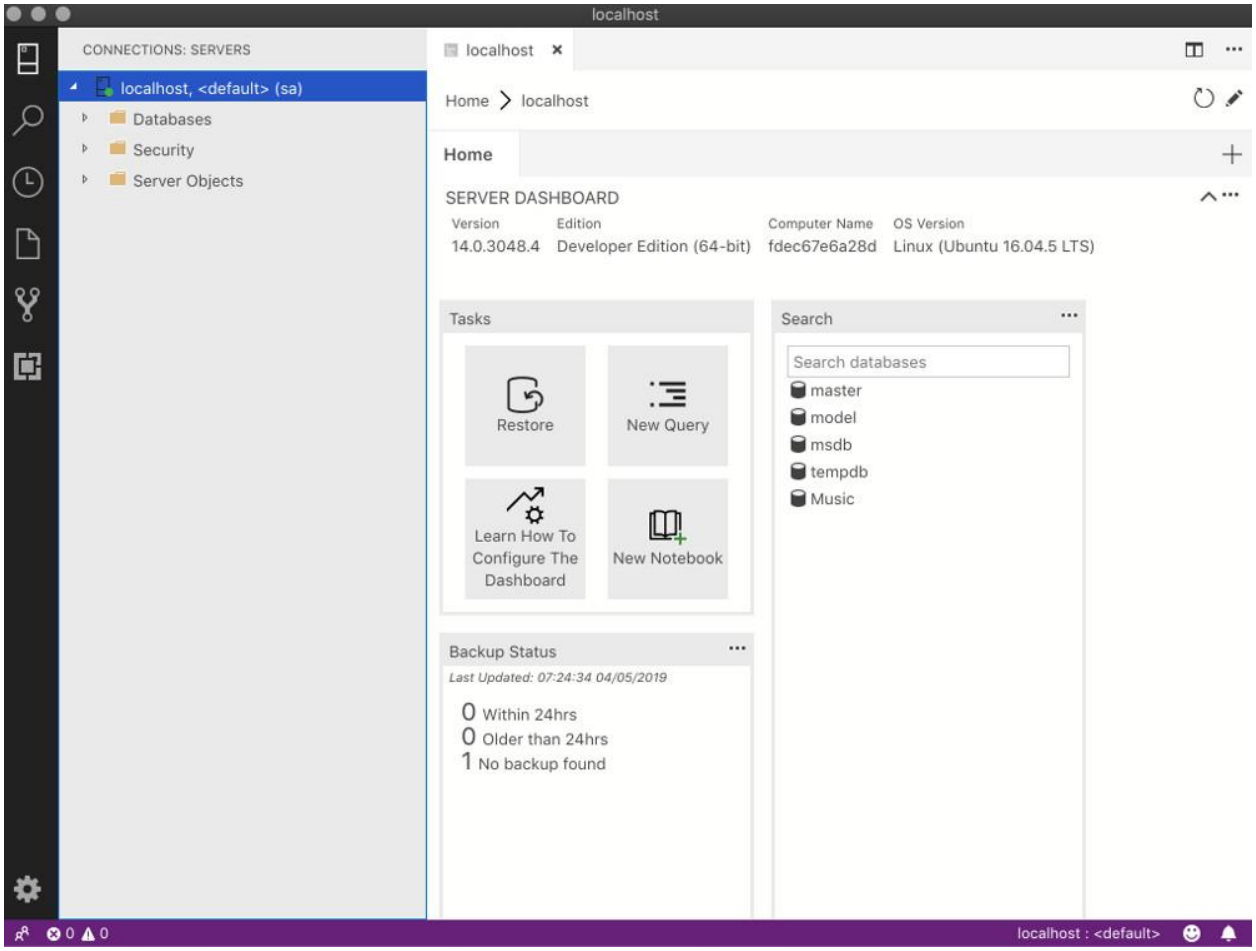
- Server Name: localhost
- Authentication Type: SQL Login
- User name: sa

- Password: reallyStrongPwd123
- Database Name: <default>
- Server Group: <default>

If you use a port other than the default 1433, click `Advanced` and enter it in the `Port` field.

Alternatively, you can append it to your server name, with a comma between. For example, if you use port 1400, use `localhost,1400`.

Once Azure Data Studio has connected to the SQL Server instance, you'll be presented with the server dashboard, which looks something like this:



You can now go ahead and create databases, run scripts, and perform other SQL Server management tasks.

Error when Connecting?

If you receive an error when trying to connect, check that SQL Server is in fact running.

If you've previously installed SQL Server on your Mac, but you still get a connection error, make sure your Mac has Docker running and you've started the SQL Server Docker container.

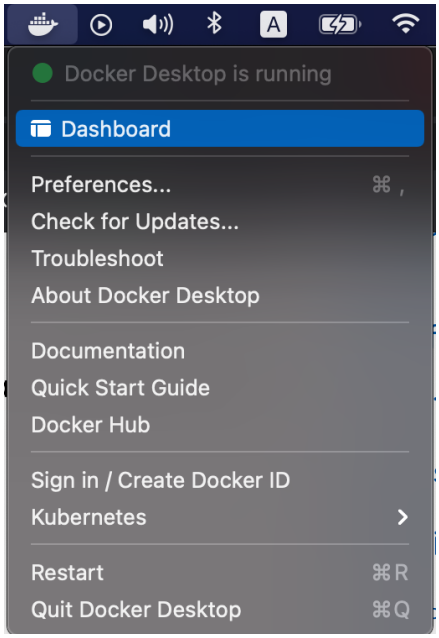
Below are instructions for launching Docker and starting the Docker container.

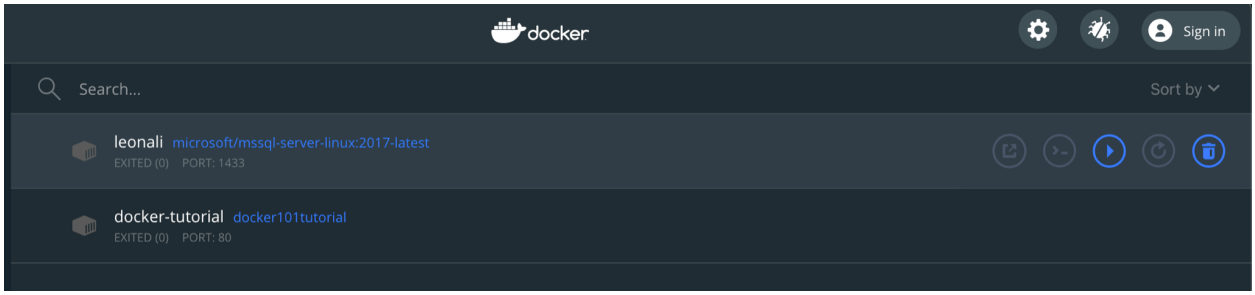
Open Docker and Start the Docker Container

1. Open Docker just as you would open any other application (e.g. via the Launchpad, Applications folder, etc).
2. Once Docker is running, open your Terminal and start the Docker container. It should look something like this:
   `docker start sql_server_demo`

   This starts a previously stopped container called `sql_server_demo`*. You'll need to modify this command to suit your own container's name.*

3. You may try via app's "START" button designated below:





This starts a previously stopped container called `leonali`.

8. Connecting to SQL Server (Method 2-Terminal)

a. **Install sql-cli (unless already installed)**
Run the following command to install the sql-cli command line tool. This tool allows you to run queries and other commands against your SQL Server instance.
`npm install -g sql-cli`

This assumes you have NodeJs installed. If you don't, download it from Nodejs.org first. Installing NodeJs will automatically install npm which is what we use in this command to install sql-cli.

Permissions Error?

If you get an error, and part of it reads something like Please try running this command again as root/Administrator, try again, but this time prepend `sudo` to your

command:

```
sudo npm install -g sql-cli
```

b. **Connect to SQL Server**
Now that sql-cli is installed, we can start working with SQL Server via the Terminal window on our Mac.
Connect to SQL Server using the `mssql` command, followed by the username and password parameters.

```
npm install -g sql-cli reallyStrongPwd123
```

You should see something like this:

```
Connecting to localhost...done

      sql-cli version 0.6.0
      Enter ".help" for usage hints.
      mssql>
```

This means you've successfully connected to your instance of SQL Server.

c. **Run a Quick Test**
Run a quick test to check that SQL Server is up and running and you can query it.
For example, you can run the following command to see which version of SQL Server your running:

```
select @@version
```

If it's running, you should see something like this (but of course, this will depend on which version you're running):

```
+--------------------+
| (No column name)   |
|--------------------|
| Microsoft SQL Server 2017 (RTM-CU3) (KB4538853) -
15.0.4023.6 (X64)
        Mar  4 2020 00:59:26
```

```
        Copyright (C) 2017 Microsoft Corporation
        Developer Edition (64-bit) on Linux (Ubuntu 18.04.4
LTS)                          |
+---------------------+
(1 row affected)
```

If you see a message like this, congratulations — SQL Server is now up and running on your Mac!

Related Links for M1 chip:
https://medium.com/geekculture/docker-express-running-a-local-sql-server-on-your-m1-mac-8bbc22c49dc9