MSSQL 2019 Server on Docker Container:

Prerequisites 'OS requirements':

To install Docker Engine, you need the 64-bit version of one of these Ubuntu versions:

Ubuntu Groovy 20.10 Ubuntu Focal 20.04 (LTS) Ubuntu Bionic 18.04 (LTS) Ubuntu Xenial 16.04 (LTS) Docker Engine is supported on x86_64 (or amd64), armhf, and arm64 architectures.

NOTE: in this example we are using Ubuntu Bionic 18.04 (LTS)

Install using the repository:

Before you install Docker Engine for the first time on a new host machine, you need to set up the Docker repository. Afterward, you can install and update Docker from the repository.

 # Update the apt package index and install packages to allow apt to use a repository over HTTPS:

$ sudo apt-get update && sudo apt-get install

$ sudo apt-transport-https ca-certificates curl gnupg lsb-release

 # Add Docker's official GPG key:

$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg

 # Use the following command to set up the stable repository.

To add the nightly or test repository, add the word nightly or test (or both) after the word stable in the commands below.

$ echo \

"deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu \

$(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

 # Now install Docker:

$ sudo apt-get update

$ sudo apt-get install docker-ce docker-ce-cli containerd.io

 # Verify that Docker Engine is installed correctly by running the hello-world image.

$ sudo docker run hello-world

```
root@                    :/home/ubuntu# sudo docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/
```

 # This command downloads a test image and runs it in a container. When the container runs, it prints an informational message and exits

You can start Docker and check status:

$ sudo systemctl start docker

$ sudo systemctl status docker

```
root@t                   :/home/ubuntu# sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2021-04-01 22:15:57 UTC; 2 days ago
     Docs: https://docs.docker.com
 Main PID: 31919 (dockerd)
    Tasks: 24
   CGroup: /system.slice/docker.service
           ├─  599 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 1433 -container-ip 172.17.0.2 -container-port 1433
           └─31919 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Apr 01 22:15:57                  dockerd[31919]: time="2021-04-01T22:15:57.861076631Z" level=info msg="API listen on /var/run/docker.sock"
Apr 01 22:16:21                  dockerd[31919]: time="2021-04-01T22:16:21.102940835Z" level=info msg="ignoring event" container=27598829fc528fab3b788b4f40bd9
Apr 01 22:32:59                  dockerd[31919]: time="2021-04-01T22:32:59.208771784Z" level=info msg="ignoring event" container=b60c5e18f0b600bae9898015cd127
Apr 01 22:34:05                  dockerd[31919]: time="2021-04-01T22:34:05.068606513Z" level=error msg="Not continuing with pull after error: errors:\ndenied:
Apr 01 22:34:05                  dockerd[31919]: time="2021-04-01T22:34:05.068691601Z" level=info msg="Ignoring extra error returned from registry: unauthoriz
Apr 03 20:45:54                  dockerd[31919]: time="2021-04-03T20:45:54.719656830Z" level=info msg="ignoring event" container=249cb8c9dd8254b5cf385bbba67c1
Apr 03 20:51:39                  dockerd[31919]: time="2021-04-03T20:51:39.950946103Z" level=info msg="ignoring event" container=bdf0686fb45cd55305e18b2d78508
Apr 04 09:54:42                  dockerd[31919]: time="2021-04-04T09:54:42.975694673Z" level=error msg="Not continuing with pull after error: errors:\ndenied:
Apr 04 09:54:42                  dockerd[31919]: time="2021-04-04T09:54:42.975805649Z" level=info msg="Ignoring extra error returned from registry: unauthoriz
Apr 04 21:04:42                  dockerd[31919]: time="2021-04-04T21:04:42.893854674Z" level=info msg="ignoring event" container=7e7afc46c6c4518f6790ae31d973b
```

As you see above, I already installed MSSQL image and it' running on port 1433 on localhost.

 # Pull and run the 2019 container image

Before starting the following steps, make sure that you have selected your preferred shell (bash, PowerShell, or cmd) at the top of this article.

 # Pull the SQL Server 2019 Linux container image from Microsoft Container Registry.

$ sudo docker pull mcr.microsoft.com/mssql/server:2019-latest

```
root@                    :/home/ubuntu# sudo docker pull mcr.microsoft.com/mssql/server:2019-latest
2019-latest: Pulling from mssql/server
Digest: sha256:ec5492b0b3f9c0707fddd37f0bd3d47d3ebea94a3054afb8b50e9e746d1e5f37
Status: Image is up to date for mcr.microsoft.com/mssql/server:2019-latest
mcr.microsoft.com/mssql/server:2019-latest
```

As you see image is already downloaded and up to date.

To run the container image with Docker, you can use the following command from a bash shell (Linux/macOS) or elevated PowerShell command prompt.

$ sudo docker run -e "ACCEPT_EULA=Y" -e "SA_PASSWORD=<pasteYourPasswordHere>" -p 1433:1433 --name mssql -h mssql -d mcr.microsoft.com/mssql/server:2019-latest

To view your Docker containers, use the docker ps command.

$ sudo docker ps -a

```
root@              /home/ubuntu# sudo docker ps -a
CONTAINER ID   IMAGE                                   COMMAND                  CREATED          STATUS                      PORTS                    NA
MES
7e7afc46c6c4   hello-world                             "/hello"                 11 minutes ago   Exited (0) 11 minutes ago                            wo
nderful_wing
945ee6257b0e   mcr.microsoft.com/mssql/server:2019-latest   "/opt/mssql/bin/perm…"   24 hours ago     Up 24 hours                 0.0.0.0:1433->1433/tcp   ms
sql
```

As you see, container is up for 24 hours and this is a quick way to check the status of your container.

To access it via sqlcmd:

$ docker exec -it mssql /opt/mssql-tools/bin/sqlcmd -S localhost -U sa -P 'pasteYourPasswordHere'

```
root@`  `  `         :/home/ubuntu# docker exec -it mssql /opt/mssql-tools/bin/sqlcmd -S localhost -U sa -P '[redacted]'


1> select name from sys.databases
2> go
name
--------------------------------------------------------------------------------------------------------------------------------
master
tempdb
model
msdb
Demo

(5 rows affected)
1>
```

And now, you're inside the database!!

 # You can also access it through SSMS or even Azure Data Studio: