**Lab 9 : Docker with volume container**

In the second part of lab 5, a whiskeyshop application was modified to use a mySql database and build to a dockerized image. A problem with that setup is that the data is persisted in the same container and reset to the initial context on every restart. The solution for this problem is to persist the data outside of the container that contains the database software. This is what you will be doing in this lab.

**Prerequisites**

- Have Docker installed

**Let’s get started**

The application we are going to use in this tutorial is the whiskeyshop application (created in lab 7 from GitHub <https://github.com/AMIS-Services/sig-springboot-1>).

A finalized image for that lab 7 is (temporarily) available at:

https://www.dropbox.com/s/rn99pq3wch8ec8b/whiskeyshop.tar?dl=0

# Step 1. Create volume container

In this step you are going to create a volume container which will be used attached to/used by the container with the mySql database.

* Open a command shell, eg. powershell with admin privileges (DO NOT USE Powershell ISE!!)
* Verify the volumes already present on your machine:

**docker volume ls**

* Check which containers are on your machine (the –a option gives you the containers that have stopped running as well)

**docker ps –a**

* Create a new volume container

*docker create -v /var/lib/mysql --name vc\_mySql busybox*

Although the above statement creates a volume container perfectly, it leaves the container in stopped status after creation. To be able to get access into the container use the following statement:

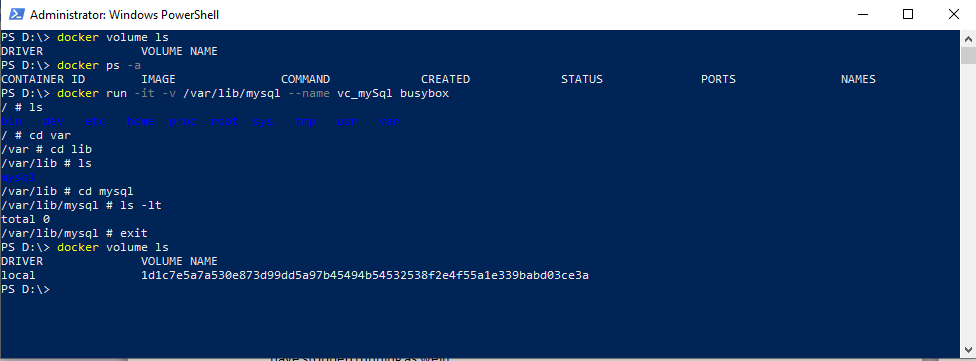
**docker run -it -v /var/lib/mysql --name vc\_mySql busybox**

* (Optionally) check the contents in the volume container

**ls -lt /var/lib/mysql**

If the volume container creation went well, you are now located at a commend line. You can list (command: ls) the contents of the folder /var/lib/mysql , which should be empty.

* Do not exit the container yet (although this is possible)!



# Step 2. Start a container with the mySql database

Now you have a volume container available, it can be attached to other containers for persistency of data. For our example, you are going to start a container with a mySql database:

* **Open a second powershell window with admin priviliges**
* **docker run --volumes-from vc\_mySql -e MYSQL\_ROOT\_PASSWORD=pass -e MYSQL\_DATABASE=test -p 3306:3306 -d mysql**

Via the option “—volumes-from <container>” you mount the path(s) (and/or file(s)) of a specified (volume) container to another container. What happens is that the path(s) (and/or file(s)) in the volume container will be used, instead of the same path(s) (and/or file(s) in the container started with the docker run command.

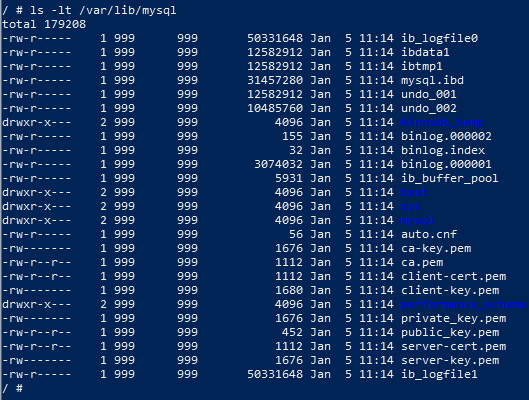
# Step 3. Check the contents in the volume container

Switch back to the shell, you were in at the end of step 1.

Re-check the contents of the folder var/lib/mysql:

**ls -lt /var/lib/mysql**

See that it now contains the files that will be used by the mySql database



Now, exit the shell

**exit**

# Step 4.

Check the state of the **running** containers

**docker ps**

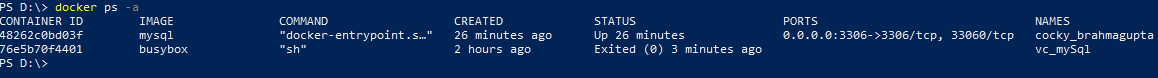


As you can see, the vc-mySql is not listed here.

This is because the container stopped running.

To also list the non running containers type:

**docker ps -a**



# Note !

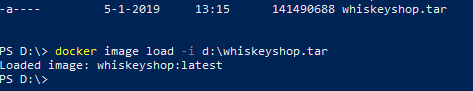
Although the volume container is in the stopped running state, it’s volumes are still available to the container(s) that mount to it.



# Step 5.

In this step you will be running a container from the whiskeyshop application image that was created in lab 7. If you do not have that image available anymore (you can check this via the docker command : **docker image ls** andsee if whiskeyshop is listed), you can load it from the whiskeyshop.tar file that comes with this lab.

**docker image load –i** <location to the whiskeyshop.tar>



Now that you have the image for the whiskeyshop at your disposal, start a container from it.

**docker run -dp 9000:8080 -e spring.profiles.active="mysql" whiskeyshop**

As you might remember from that lab, the profile setting instructs the application to use the mySql database (in stead of the in memory database).



Check that the container started by typing **docker ps,** and note the port mapping



# Step 6.

Open a browser and type : <http://localhost:9000/whiskeys>

As you can see, no whiskeys are displayed. Only an collection is shown.

The empty collection was created when you started the whiskeyshop container from the image. As you might recall from lab 5, the whiskeyshop application has a configuration file, named application-mysql.properties, which contains the setting ‘spring.jpa.hibernate.ddl-auto=create’. Through this setting the schema and tables are (re)created in the database. To prevent this from happening, the setting can be changed to the value none. For the changed value to take effect, a new image should be build from the source code, something that is not preferable for a changed configuration. The value for this setting should be an injectable to the application, something that can be done through environment variables (which is a separate subject and nit in scope in this lab).

Let’s add some data to the database!

To do this, start a client in the container created from the mysql image.

**docker ps**

lookup the container name (as shown below)

**docker exec -it <container name> mysql -uroot –p**

**this command starts a mySql client**

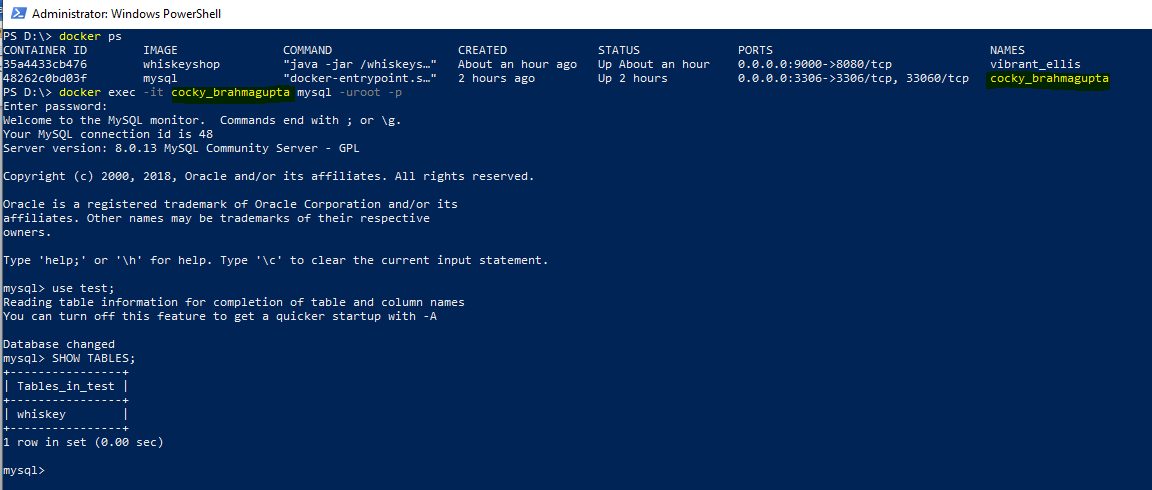
For the password, type **pass**

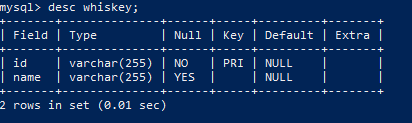
**use test;**

Switch to the test database

**SHOW TABLES;**

The whiskey table is shown;





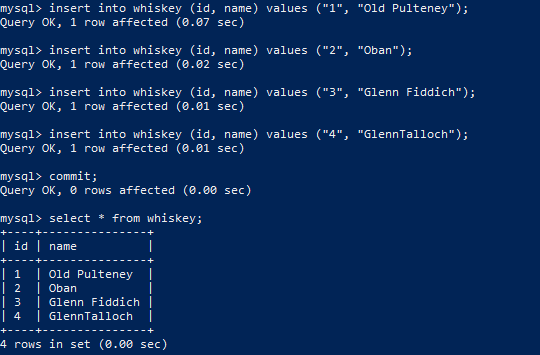
**insert into whiskey (id, name) values ("1", "Old Pulteney");**

**insert into whiskey (id, name) values ("2", "Oban");**

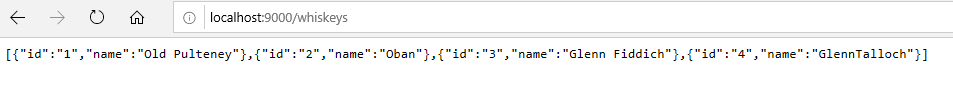
**insert into whiskey (id, name) values ("3", "Glenn Fiddich");**

**insert into whiskey (id, name) values ("4", "GlennTalloch");**

**commit;**



Refresh the browser for : <http://localhost:9000/whiskeys>



# Step 7.

To prove that the data is persisted, stop the container from the mySql image.

After the container has stopped, start a new one.

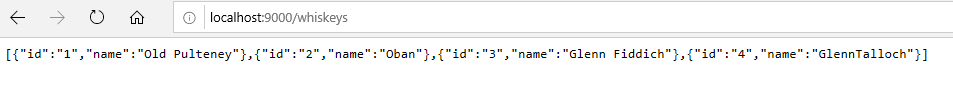
**docker ps**

lookup the container id for mySql

**docker stop <container id>**

**docker run --volumes-from vc\_mySql -e MYSQL\_ROOT\_PASSWORD=pass -e MYSQL\_DATABASE=test -p 3306:3306 -d mysql**

Refresh the browser for : <http://localhost:9000/whiskeys>



See that whiskeys are still there.

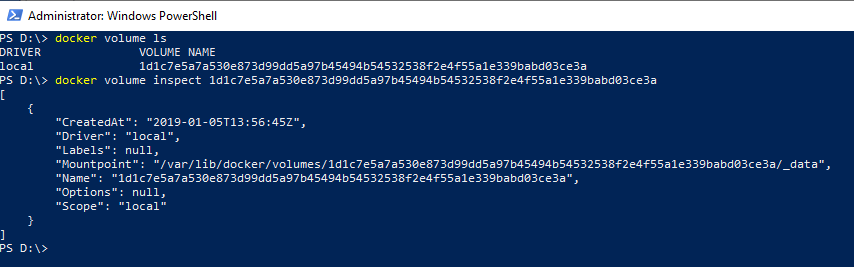
# Step 8. Inspect

List the volume(s)

**docker volume ls**

Inspect the volume

**docker volume inspect <volume name>**



# Step 9. Clean up

A volume container can be mounted to one or more containers. Docker keeps track of this.

A volume can only be removed (**docker volume rm <volume name>**) if there exist no references to it, otherwise an error message is shown.



In our example, the volume is referenced by the mySql containers (1 running and 1 stopped). It also is referenced by the volume container

Stop the running mySql container:

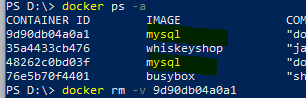
**docker ps** (to find the container id)

**docker stop <container id>**

Remove the mySql containers:

**docker ps –a** (to also find the stopped containers)

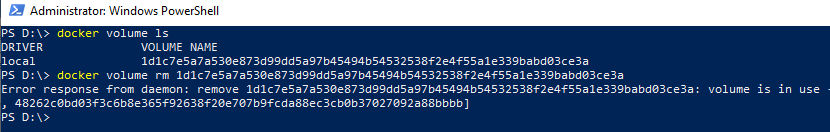
**docker rm –v <container id>** (remove one of the 2 stoppped mySql containers)



**docker volume ls** (list the volumes)

try to remove the volume again and see that this still gives an error

**docker volume rm <volume name>**



Remove the 2nd stopped mySql container:

**docker rm –v <container id>**

List the volumes again

**docker volumes ls**

Remove the volume container (this is the one created from image busybox):

**docker rm –v <container id>**

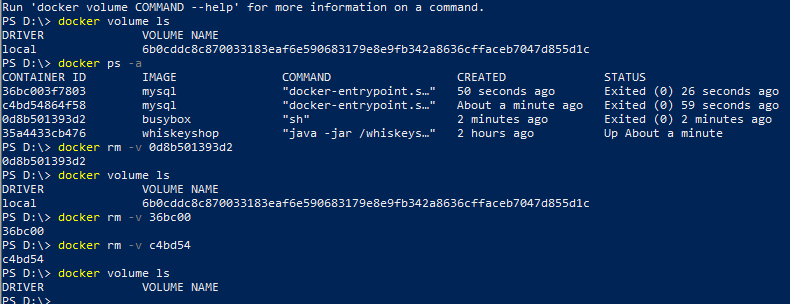
List the volumes again

**docker volumes ls**

If everything went well, there shouldn’t be any volumes listed anymore.

**Can you see why it is important to specify the –v option when removing containers that have reference(s) to volume containers. It decreases the reference counter that docker keeps track of, and therefor volumes that are no longer being referenced can be removed automatically. It can be a burden for you, to find out which references still exist to containers that have been removed without the –v (volume) option.**

The below screen shot proves that the reference to a volume is not only held by the volume container, but by the mysql container as well. On removing the volume container, the volume is not released yet. This is done after removing the mySql container(s) as well.





De schrijfwijze is afhankelijk van de herkomst van de drank. Whiskey uit [Ierland](https://nl.wikipedia.org/wiki/Ierland_(land)) en de [Verenigde Staten](https://nl.wikipedia.org/wiki/Verenigde_Staten) wordt met een *e* geschreven, whisky uit [Schotland](https://nl.wikipedia.org/wiki/Schotland) en [Canada](https://nl.wikipedia.org/wiki/Canada) zonder *e*. Het meervoud is, volgens de Engelse spellingregels, *whiskeys*, respectievelijk *whiskies*. De officiële Nederlandse schrijfwijze is *whiskey*, en in meervoud *whiskeys*[[1]](https://nl.wikipedia.org/wiki/Whisky#cite_note-1), resp. *whisky*, en in meervoud *whisky's*.[[2](https://nl.wikipedia.org/wiki/Whisky#cite_note-2)