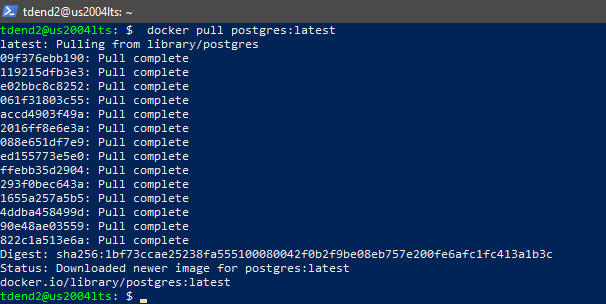
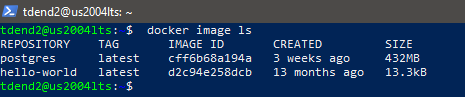
**Running PostgreSQL DBMS using Docker**

a. Pulling a PostgreSQL image



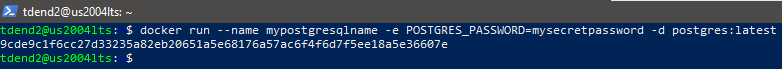
check whether the download was successful by running the command: ***docker image ls***



*Found the Postgres (latest) image in the list.*

***—---------------------------------------------------------------------------------------------------------------------------***

b. Running your PostgreSQL container:



Executed a Postgres container by specifying all the necessary information in one command which tells Docker to run a new container under a particular container name, defines the Postgres password, and downloads the latest Postgres release.

Docker run –name <name of container > -e <environment tag: here it’spassword>=<set password> -d postgres:latest

*Here are the* ***operations*** *that are broken down:*

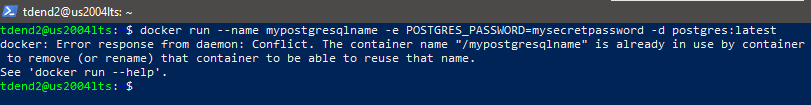
• --name: We’re naming the container ‘mypostgresqlname’.

• -e: Set the password to “mysecretpassword” using the environment tag -e.

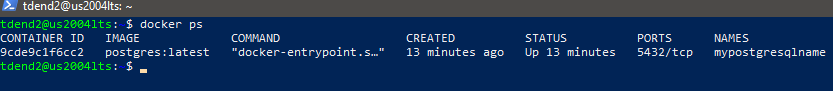
• -d: We're running detached (-d) mode (so in the background).

• Used Postgres:latest to launch the container.

If we didn’t pull the image before, docker would check whether we have the image in your local directory. If we don’t have it in your local directory, Docker will pull the image automatically.



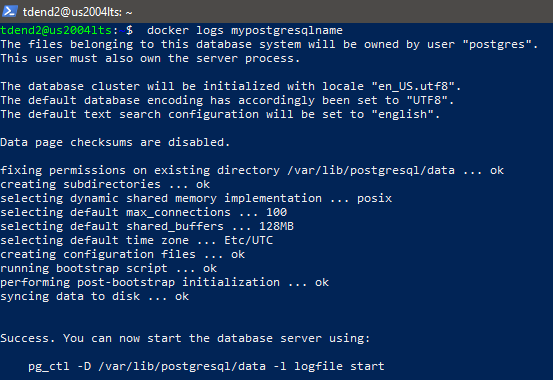
Prompting Docker to list all running containers: ***docker ps***



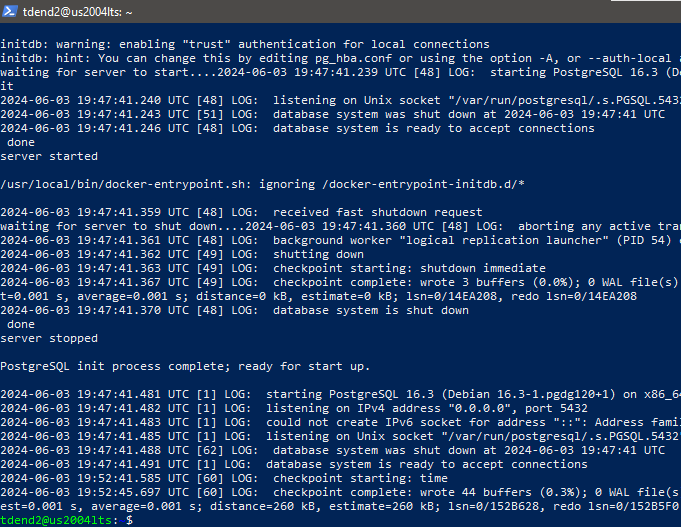
***Container ID*** of ‘mypostgresqlname’ container is **9cde9c1f6cc2**

**Log output :**

It can be checked using ***docker logs <name of container>***

****

***Database ready to accept connections and the second half of result is below:***

****

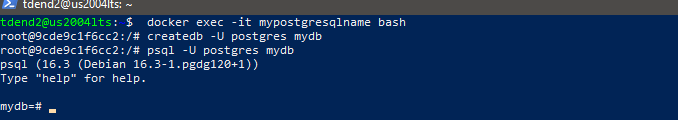
**—----------------------------------------------------------------------------------------------------------------------------**

c. Creating a database and table for the Titanic dataset in your PostgreSQL DBMS



created a database to load a dataset using ‘docker exec’ to launch an interactive shell running inside our mypostgresqlname container with the PostgreSQL CLI tools installed. This saves us from needing tools to connect to and manage PostgreSQL databases installed locally.

Also the container ID as the hostname of container.



o created a new database with the name ‘mydb’ and launched the psql utility, which is a CLI tool for PostgreSQL, connected to our mydb database.

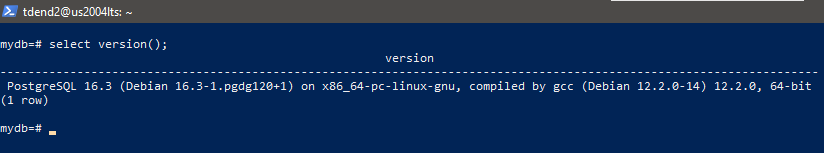
***‘postgres’ is a default superuser or administrator like PostgreSQL DB’s root.***

**PSQL basic commands:**

‘\l’ lists the databases:

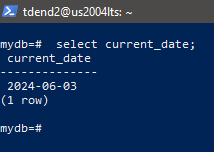
Mydb and postgres are also found in the list of all databases.

Version: can be using ***select version();***



The version of postgresql found is 16.3

Current date found is june 3, 2024 using ***select current\_date;:***



To quit from the psql: \q

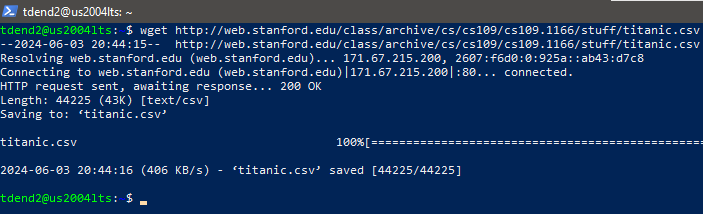


Now **exiting** returns to VM where the container is running.

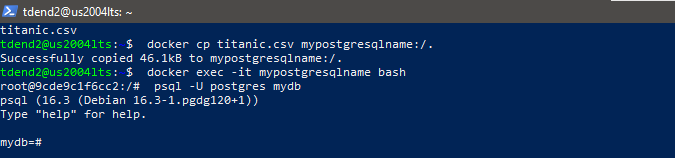
**Creating a database for titanic dataset:**

1)Downloading the stanford titanic dataset from the internet using wget as follows:

wget <http://web.stanford.edu/class/archive/cs/cs109/cs109.1166/stuff/titanic.csv>

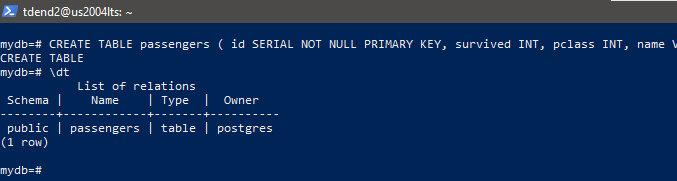


2) copy the file into the root directory in your container,connect your PostgreSQL container, run the ‘ls’ command to check the file and then connect your DB using ‘psql’ to create a table to store the data.



3) **create table as follows :**

**CREATE TABLE passengers ( id SERIAL NOT NULL PRIMARY KEY, survived INT, pclass INT, name VARCHAR(255), sex TEXT, age FLOAT8, siblings\_spouses INT, parents\_children INT, fare FLOAT8 );**

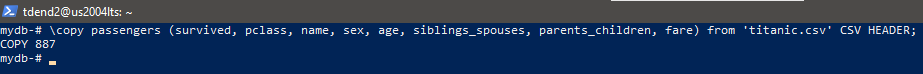
****

Verified the existence of **passengers table which stores the Titanic dataset using \dt.**

**4) Import titanic.csv** to the passengers table to populate it.

The path of titanic csv file is /home/tdend2/titanic.csv

\copy passengers (survived, pclass, name, sex, age, siblings\_spouses, parents\_children, fare) from 'titanic.csv' CSV HEADER;



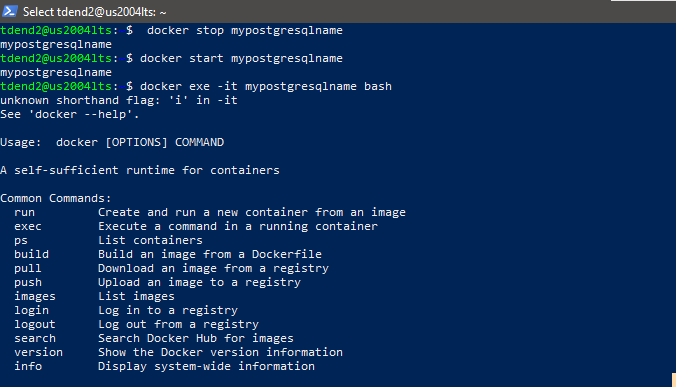
This inserted 887 passengers into your database that we can work with.

**—----------------------------------------------------------------------------------------------------------------------------**

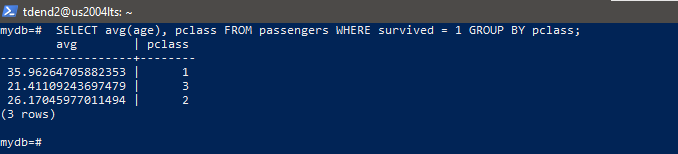
d. Running an analytic query “average age of passengers who survived the

Titanic disaster by ticket class” and results.

Before running the query checking if stopping and starting docker results in any loss of data.



Then connected to container and then to mydb using psql and executed the following:



*Ticket class is pclass which has 3 types of classes namely 1, 2 and 3 and the average age of passengers found is above 35, 21 and 26 respectively.*

================================THE END=================================