

## 1. Installing docker:

<https://docs.docker.com/get-docker/>

## 2. Make a directory

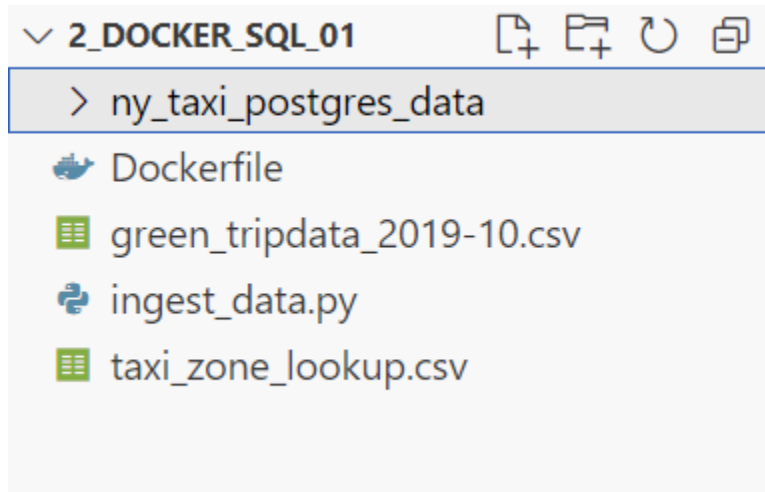
```
zhaoy@ZYX MINGW64 /e/DataEngineer
$ mkdir 2_docker_sql_01

zhaoy@ZYX MINGW64 /e/DataEngineer
$ cd 2_docker_sql_01

zhaoy@ZYX MINGW64 /e/DataEngineer/2_docker_sql_01
$ code .
```

## 3. Create a docker file and ingest\_data.py (check my Github)

## 4. Create a folder named ny\_taxi\_postgres\_data



## 5. Paste those code in the terminal and run this code

```
docker run -it \
-e POSTGRES_USER="postgres" \
-e POSTGRES_PASSWORD="postgres" \
-e POSTGRES_DB="ny_taxi" \
-v e:/DataEngineer/2_docker_sql_01/ny_taxi_postgres_data/ny_taxi_postgres_data:/var/lib/postgresql/data \
-p 5432:5432 \
--network=pg-network \
--name pg-database-01 \
Postgres:13
```

This command starts a **PostgreSQL database** server in a Docker container. It uses the **PostgreSQL 17 Alpine image** and sets up database parameters, port forwarding, and volume mapping for persistent storage.

**docker run --rm**

Starts a new Docker container, remove the container when it stops, cleaning up resources

**-e (Environment Variables)**

- These options pass **environment variables** to the container to configure the PostgreSQL database:
  - `POSTGRES_DB=ny_taxi`: Sets the default database name to `ny_taxi`.
  - `POSTGRES_USER=postgres`: Sets the username for the database to `postgres`.
  - `POSTGRES_PASSWORD=postgres`: Sets the password for the `postgres` user to `postgres`.

### **-p 5432:5432**

- Maps port **5432** on your host machine to port **5432** in the container.
  - PostgreSQL's default port is **5432**.
  - This allows you to connect to the database using tools like `psql`, DBeaver, or a Python script from your host machine.

### **-v**

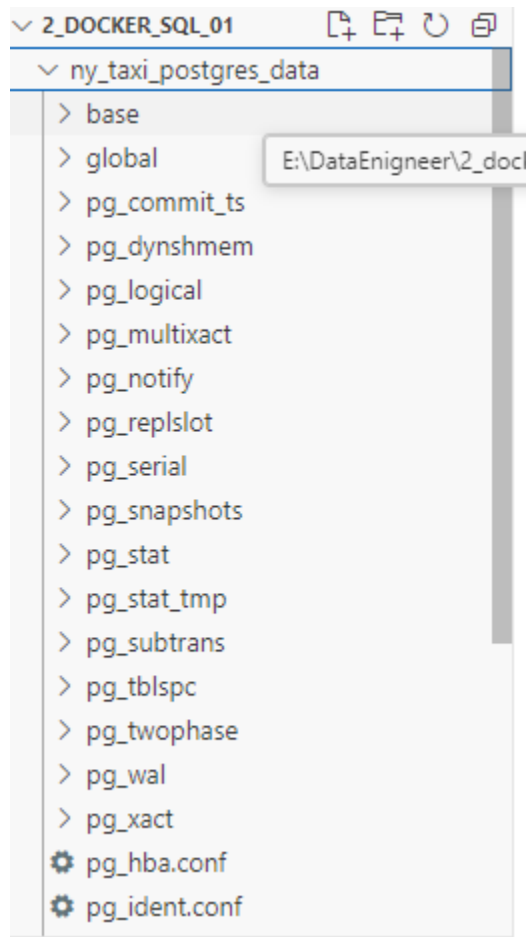
**E:/DataEnigneer/2\_docker\_sql\_01/ny\_taxi\_postgres\_data:/var/lib/postgresql/data**

- **Volume Mapping**: Maps a directory on your local file system (`E:/DataEnigneer/2_docker_sql_01/ny_taxi_postgres_data`) to the container's **data directory** (`/var/lib/postgresql/data`).
  - **Purpose**: Ensures that your database data is stored persistently on your local machine, even if the container is removed.

### **postgres:17-alpine**

- Specifies the **Docker image** to use:
  - `postgres`: PostgreSQL database server image.
  - `17-alpine`: Version **17** of PostgreSQL built on the lightweight **Alpine Linux**.

6. After this code, you will see there are many files under the `ny_taxi_postgres_data` folder.



### What Happens When You Run It?

1. A Docker container starts using the PostgreSQL 17 Alpine image.
2. It initializes a database named `ny_taxi`, with the username and password both set to `postgres`.
3. The database listens for connections on port 5432 (accessible from your host machine).
4. Data is stored in the local directory (`E:/DataEngineer/...`) to persist across container restarts.

### Why Use This Setup?

1. **Port Mapping:** Allows database access from your host system or other applications.
2. **Environment Variables:** Configures the database automatically on startup.
3. **Persistent Storage:** Ensures database data is not lost when the container is removed.
4. **Alpine Image:** Optimized for size, making it lightweight and faster to start.

## 7. Download your dataset or import your dataset.

Open a new terminal, open the jupyter notebook

```
zhaoy@ZYX MINGW64 /e/DataEngineer/2_docker_sql_01
$ jupyter notebook
[I 2025-01-26 12:00:43.995 ServerApp] Package notebook took 0.0000s to import
[I 2025-01-26 12:00:44.038 ServerApp] Package jupyter_lsp took 0.0431s to import
[W 2025-01-26 12:00:44.038 ServerApp] A `jupyter_server_extension_points` function was not found in jupyter_lsp. Instead, a `jupyter_server_extension_paths` function was found and will be used for now. This function name will be deprecated in future releases of Jupyter Server.
[I 2025-01-26 12:00:44.059 ServerApp] Package jupyter_server_terminals took 0.0197s to import
[I 2025-01-26 12:00:44.060 ServerApp] Package jupyterlab took 0.0000s to import
[I 2025-01-26 12:00:44.179 ServerApp] Package notebook_shim took 0.0000s to import
[W 2025-01-26 12:00:44.179 ServerApp] A `jupyter_server_extension_points` function was not found in notebook_shim. Instead, a `jupyter_server_extension_paths` function was found and will be used for now. This function name will be deprecated in future releases of Jupyter Server.
[I 2025-01-26 12:00:45.073 ServerApp] Package panel.io.jupyter_server_extension took 0.8928s to import
[I 2025-01-26 12:00:45.074 ServerApp] jupyter_lsp | extension was successfully linked.
[I 2025-01-26 12:00:45.078 ServerApp] jupyter_server_terminals | extension was successfully linked.
[I 2025-01-26 12:00:45.083 ServerApp] jupyterlab | extension was successfully linked.
[I 2025-01-26 12:00:45.087 ServerApp] notebook | extension was successfully linked.
[I 2025-01-26 12:00:45.424 ServerApp] notebook_shim | extension was successfully linked.
[I 2025-01-26 12:00:45.424 ServerApp] panel.io.jupyter_server_extension | extension was successfully linked.
[I 2025-01-26 12:00:45.451 ServerApp] notebook_shim | extension was successfully loaded.
[I 2025-01-26 12:00:45.453 ServerApp] jupyter_lsp | extension was successfully loaded.
[I 2025-01-26 12:00:45.454 ServerApp] jupyter_server_terminals | extension was successfully loaded.
[I 2025-01-26 12:00:45.455 ServerApp] jupyterlab | extension loaded from C:\Users\zhaoy\anaconda3\lib\site-packages
```

## 8. Open a new terminal and run those code.

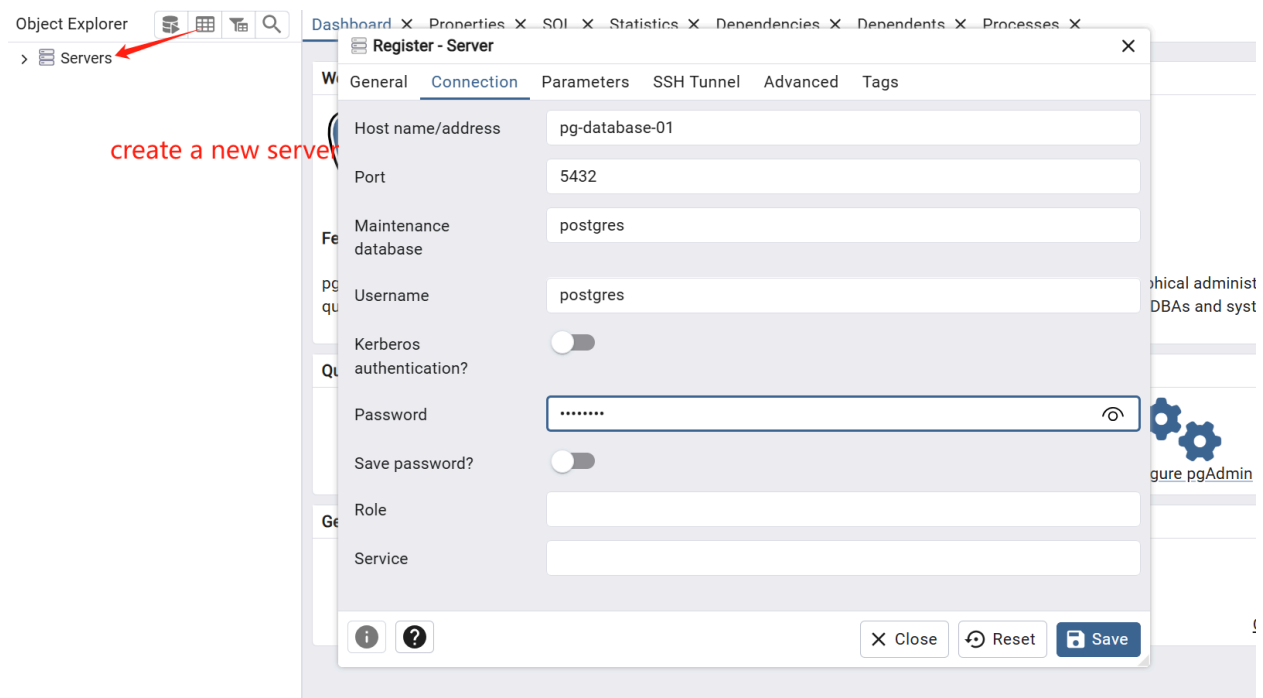
```
docker run -it \
-e PGADMIN_DEFAULT_EMAIL="admin@admin.com" \
-e PGADMIN_DEFAULT_PASSWORD="postgres" \
-p 8080:80 \
--network=pg-network \
--name pgadmin-2 \
dpage/pgadmin4
```

## 9. Open <http://localhost:8080/> in your browser

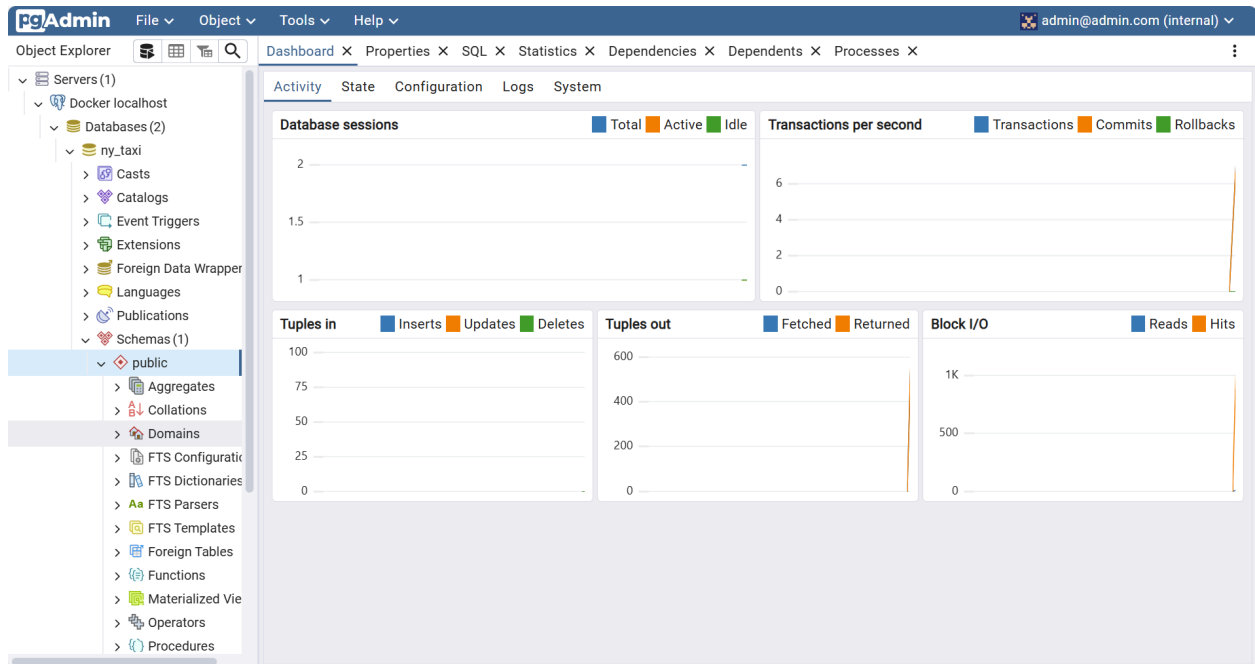
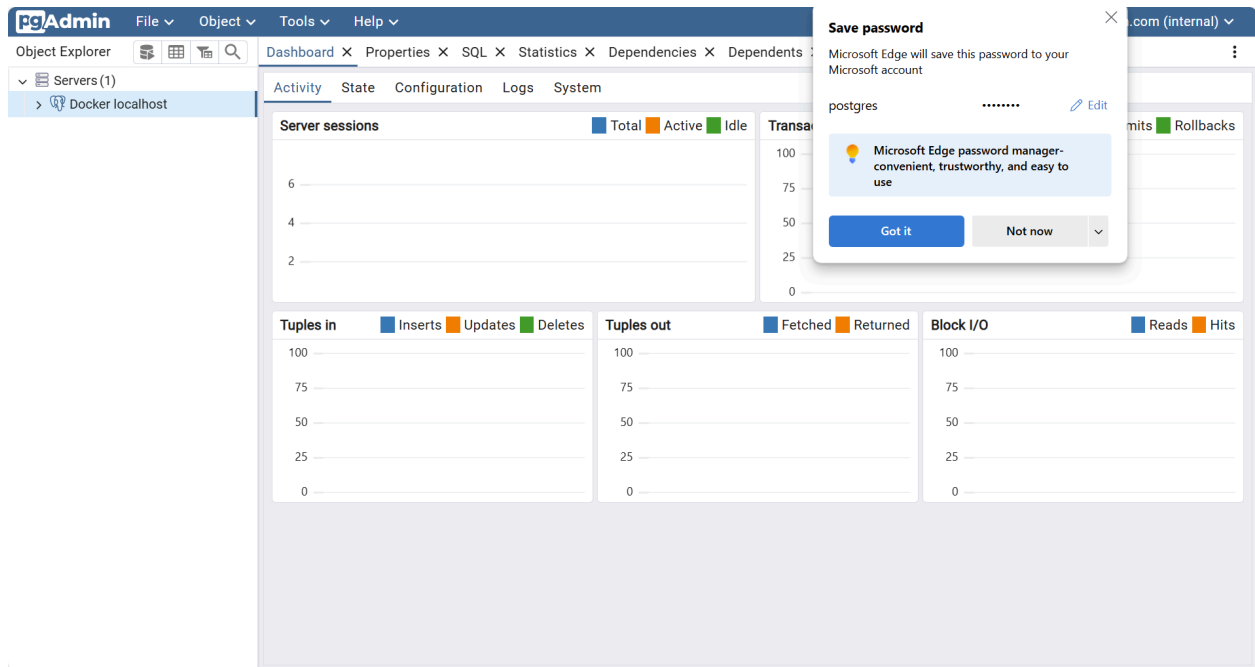
Login use your name and password

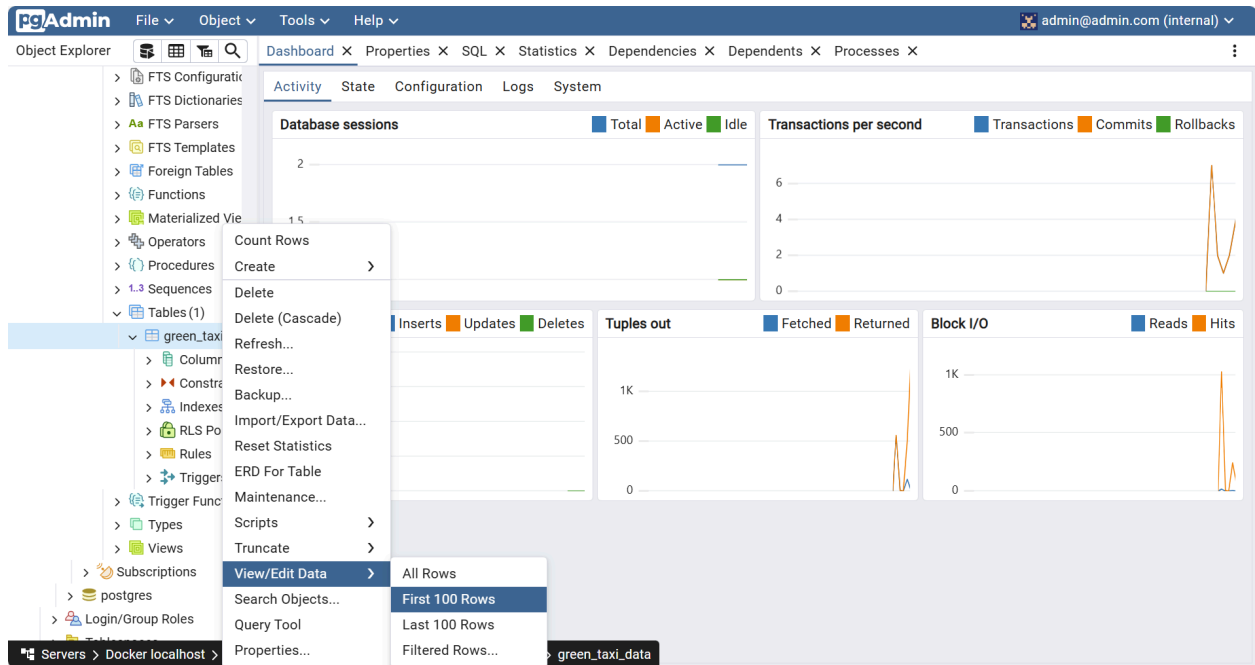


Create a new server



# 10. Explore our data





## 11. Write your code here

