Dockerized PostgreSQL and pgAdmin Setup with Preloaded Schema and Data

1. Project Folder Structure

```
project
project/Dockerfile
project/init
project/init/01-schema.sql
project/init/02-inputs.sql
```

> Files are executed in alphabetical order inside the container. Naming them with `01-`, `02-`, etc. ensures proper order.

2. Dockerfile (custom image with SQL scripts)

```
FROM postgres:latest

# Copies all .sql files into PostgreSQL's initialization directory
COPY init/*.sql /docker-entrypoint-initdb.d/
```

Explanation:

Parameter	Description
`FROM postgres:latest`	Uses the official PostgreSQL image as base.
`COPY init/*.sql`	Copies your schema and data scripts into the container. These will run on first launch only.

3. Create a Docker Network

Creates a custom bridge network so containers can talk by name.

```
docker network create pg_network
```

4. Build the PostgreSQL Image

Navigate to the `project` folder and run:

```
docker build -t my_postgres_image .
```

Command	Description
`-t my_postgres_image`	Tags the image with a custom name.
`.`	Tells Docker to use the Dockerfile in the current directory.

5. Run the PostgreSQL Container

```
docker run -d --name postgres_container --network pg_network -e POSTGRES_USER=admin -e POSTGRES_PASSWORD=admin -e POSTGRES_DB=my_database -p 5432:5432 my_postgres_image
```

Parameter	Explanation
`-d`	Runs the container in detached mode (in the background).
`--name postgres_container`	Names the container.
`--network pg_network`	Joins the container to the shared network.
`-e POSTGRES_USER=admin`	Creates a database user.
`-e POSTGRES_PASSWORD=admin`	Password for the user.
`-e POSTGRES_DB=my_database`	Creates this database on container init.
`-p 5432:5432`	Exposes PostgreSQL to the host.
`my_postgres_image`	Runs the container from the custom image you built.

6. Run the pgAdmin Container

```
docker run -d --name pgadmin_container --network pg_network -e PGADMIN_DEFAULT_EMAIL=admin@example.com -e PGADMIN_DEFAULT_PASSWORD=admin -p 5050:80 dpage/pgadmin4
```

Breakdown of Parameters:

Parameter	Explanation
`-d`	Runs pgAdmin in detached mode.
`--name pgadmin_container`	Names the pgAdmin container.
`--network pg_network`	Connects it to the same network as PostgreSQL
`-e PGADMIN_DEFAULT_EMAIL=admin@example.com`	Login email for pgAdmin.
`-e PGADMIN_DEFAULT_PASSWORD=admin`	Login password for pgAdmin.
`-p 5050:80`	Maps port 80 from the container to port 5050 on your machine.
`dpage/pgadmin4`	Uses the official pgAdmin image.

7. Access pgAdmin from Browser

Open pgAdmin in your browser: `http//localhost:5050`

Login Credentials:

Field	Value
Email	`admin@example.com`
Password	`admin`

8. Register a New Server in pgAdmin

When inside the pgAdmin interface: Right-click on **Servers→ Register → Server...**

Connection Details:

General Tab:

Field	Value
Name	`PostgreSQL Local` (or any name you prefer)

Connection Tab:

Field	Value
Host name/address	`postgres_container`
Port	`5432`
Maintenance DB	`postgres` (or `my_database`)
Username	`admin`

Field	Value
Password	`admin`

> You must use the container name as hostname because both containers are on the same Docker network.