Week 1 Homework: Docker & SQL

Georgios Grigoriou
georgiosvgrigoriou@gmail.com
Deadline: 26 January (Thursday), 22:00 CET

In this homework, I prepared the environment and practice with Docker and SQL as part of the Data Engineering Zoomcamp course. Solutions' headings are in italic and highlighted in yellow under each Question's instruction.

Question 1. Knowing docker tags

Run the command to get information on Docker

```
docker --help
```

Now run the command to get help on the "docker build" command

Which tag has the following text? - *Write the image ID to the file*

- `--imageid string`
- `--iidfile string`
- `--idimage string`
- `--idfile string`

Question 1 Solution

The correct answer is `--iidfile string`. *By running in the terminal the following command*

docker - - help build

We can see see that the answer corresponding to Write the image ID to the file is

`--iidfile string`

```
Last login: Mon Jan 23 18:45:47 on console
(base) georgiosgrigoriou@Georgioss-MacBook-Pro ~ % docker --help build

Usage:  docker build [OPTIONS] PATH | URL | -

Build an image from a Dockerfile

Options:
      --add-host list           Add a custom host-to-IP mapping (host:ip)
      --build-arg list          Set build-time variables
      --cache-from strings      Images to consider as cache sources
      --cgroup-parent string    Optional parent cgroup for the container
      --compress                Compress the build context using gzip
      --cpu-period int          Limit the CPU CFS (Completely Fair
                                Scheduler) period
      --cpu-quota int           Limit the CPU CFS (Completely Fair
                                Scheduler) quota
  -c, --cpu-shares int          CPU shares (relative weight)
      --cpuset-cpus string      CPUs in which to allow execution (0-3, 0,1)
      --cpuset-mems string      MEMs in which to allow execution (0-3, 0,1)
      --disable-content-trust   Skip image verification (default true)
  -f, --file string             Name of the Dockerfile (Default is
                                'PATH/Dockerfile')
      --force-rm                Always remove intermediate containers
      --iidfile string          Write the image ID to the file
      --isolation string        Container isolation technology
      --label list              Set metadata for an image
  -m, --memory bytes            Memory limit
      --memory-swap bytes       Swap limit equal to memory plus swap:
                                '-1' to enable unlimited swap
      --network string          Set the networking mode for the RUN
                                instructions during build (default "default")
      --no-cache                Do not use cache when building the image
      --pull                    Always attempt to pull a newer version of
                                the image
  -q, --quiet                   Suppress the build output and print image
                                ID on success
      --rm                      Remove intermediate containers after a
                                successful build (default true)
      --security-opt strings    Security options
      --shm-size bytes          Size of /dev/shm
      --squash                  Squash newly built layers into a single
                                new layer
  -t, --tag list                Name and optionally a tag in the
                                'name:tag' format
      --target string           Set the target build stage to build.
      --ulimit ulimit           Ulimit options (default [])
```

Figure 1: running *docker - - help build* in terminal

Question 2. Understanding docker first run

Run docker with the python:3.9 image in an interactive mode and the entrypoint of bash. Now check the python modules that are installed (use pip list). How many python packages/modules are installed?

- 1
- 6
- 3
- 7

Question 2 Solution

The correct answer is 3. *By running in the terminal the following commands*

docker run -it —entrypoint=bash python:3.9
pip list

*We can see that we have **3 packages/modules***

```
(base) georgiosgrigoriou@Georgioss-MacBook-Pro data_engineering_zoomcamp % docker run -it --entrypoint=bash python:3.9
root@efa559ca4134:/# pip list
Package    Version
---------- -------
pip        22.0.4
setuptools 58.1.0
wheel      0.38.4
WARNING: You are using pip version 22.0.4; however, version 22.3.1 is available.
You should consider upgrading via the '/usr/local/bin/python -m pip install --upgrade pip' command.
root@efa559ca4134:/#
```

Prepare Postgres

Run Postgres and load data as shown in the videos We'll use the green taxi trips from January 2019:

```
wget https://github.com/DataTalksClub/nyc-tlc-data/releases/download/green/green_tripdata_2019-01.csv.gz
```

You will also need the dataset with zones:

```
wget https://s3.amazonaws.com/nyc-tlc/misc/taxi+_zone_lookup.csv
```

Download this data and put it into Postgres (with jupyter notebooks or with a pipeline)

Prepare Postgres

Using the pipeline shown in the sessions of week 1 (DE Zoomcamp 1.2.1-1.2.6) and the instructions here, the datasets about the green taxi trips and taxi zone lookups were stored in the Postgresql database

Folder, files and code can be found in this link

Prepare Postgres and pgAdmin

docker-compose.yml file

```yaml
services:

 pgdatabase:

   image: postgres:13

   environment:

      - POSTGRES_USER=root

      - POSTGRES_PASSWORD=root

      - POSTGRES_DB=ny_taxi
```

```
  volumes:
    - "./ny_taxi_postgres_data:/var/lib/postgresql/data:rw"
  ports:
    - "5432:5432"
pgadmin:
  image: dpage/pgadmin4
  environment:
    - PGADMIN_DEFAULT_EMAIL=admin@admin.com
    - PGADMIN_DEFAULT_PASSWORD=root
  ports:
    - "8080:80"
```

Running docker-compose.yml file

```
Last login: Tue Jan 24 08:38:13 on ttys000
(base) georgiosgrigoriou@Georgioss-MacBook-Pro ~ % cd Desktop
(base) georgiosgrigoriou@Georgioss-MacBook-Pro Desktop % cd homework_week1_docker_sql
(base) georgiosgrigoriou@Georgioss-MacBook-Pro homework_week1_docker_sql % docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
(base) georgiosgrigoriou@Georgioss-MacBook-Pro homework_week1_docker_sql % docker-compose up
[+] Running 3/3
 ⠿ Network homework_week1_docker_sql_default         Created              0.1s
 ⠿ Container homework_week1_docker_sql-pgadmin-1     Created              0.2s
 ⠿ Container homework_week1_docker_sql-pgdatabase-1  Created              0.2s
Attaching to homework_week1_docker_sql-pgadmin-1, homework_week1_docker_sql-pgdatabase-1
homework_week1_docker_sql-pgdatabase-1  | The files belonging to this database system will be owned by user "postgres".
homework_week1_docker_sql-pgdatabase-1  | This user must also own the server process.
homework_week1_docker_sql-pgdatabase-1  |
homework_week1_docker_sql-pgdatabase-1  | The database cluster will be initialized with locale "en_US.utf8".
homework_week1_docker_sql-pgdatabase-1  | The default database encoding has accordingly been set to "UTF8".
homework_week1_docker_sql-pgdatabase-1  | The default text search configuration will be set to "english".
homework_week1_docker_sql-pgdatabase-1  |
homework_week1_docker_sql-pgdatabase-1  | Data page checksums are disabled.
homework_week1_docker_sql-pgdatabase-1  |
homework_week1_docker_sql-pgdatabase-1  | fixing permissions on existing directory /var/lib/postgresql/data ... ok
homework_week1_docker_sql-pgdatabase-1  | creating subdirectories ... ok
homework_week1_docker_sql-pgdatabase-1  | selecting dynamic shared memory implementation ... posix
homework_week1_docker_sql-pgdatabase-1  | selecting default max_connections ... 100
homework_week1_docker_sql-pgdatabase-1  | selecting default shared_buffers ... 128MB
homework_week1_docker_sql-pgdatabase-1  | selecting default time zone ... Etc/UTC
homework_week1_docker_sql-pgdatabase-1  | creating configuration files ... ok
homework_week1_docker_sql-pgdatabase-1  | running bootstrap script ... ok
homework_week1_docker_sql-pgdatabase-1  | performing post-bootstrap initialization ... ok
homework_week1_docker_sql-pgadmin-1     | NOTE: Configuring authentication for SERVER mode.
homework_week1_docker_sql-pgadmin-1     |
homework_week1_docker_sql-pgadmin-1     | pgAdmin 4 - Application Initialisation
homework_week1_docker_sql-pgadmin-1     | ======================================
homework_week1_docker_sql-pgadmin-1     |
homework_week1_docker_sql-pgdatabase-1  | syncing data to disk ... ok
homework_week1_docker_sql-pgdatabase-1  |
homework_week1_docker_sql-pgdatabase-1  | initdb: warning: enabling "trust" authentication for local connections
homework_week1_docker_sql-pgdatabase-1  | You can change this by editing pg_hba.conf or using the option -A, or
homework_week1_docker_sql-pgdatabase-1  | --auth-local and --auth-host, the next time you run initdb.
homework_week1_docker_sql-pgdatabase-1  |
homework_week1_docker_sql-pgdatabase-1  | Success. You can now start the database server using:
homework_week1_docker_sql-pgdatabase-1  |
homework_week1_docker_sql-pgdatabase-1  |     pg_ctl -D /var/lib/postgresql/data -l logfile start
homework_week1_docker_sql-pgdatabase-1  |
homework_week1_docker_sql-pgdatabase-1  | waiting for server to start....2023-01-24 08:40:16.047 UTC [49] LOG:  starting PostgreSQL 13.9 (Debian 13.9-1.pgdg110+1) on x8
Debian 10.2.1-6) 10.2.1 20210110, 64-bit
homework_week1_docker_sql-pgdatabase-1  | 2023-01-24 08:40:16.050 UTC [49] LOG:  listening on Unix socket "/var/run/postgresql/.s.PGSQL.5432"
homework_week1_docker_sql-pgdatabase-1  | 2023-01-24 08:40:16.075 UTC [50] LOG:  database system was shut down at 2023-01-24 08:40:12 UTC
homework_week1_docker_sql-pgdatabase-1  | 2023-01-24 08:40:16.104 UTC [49] LOG:  database system is ready to accept connections
homework_week1_docker_sql-pgdatabase-1  |  done
homework_week1_docker_sql-pgdatabase-1  | server started
homework_week1_docker_sql-pgdatabase-1  | CREATE DATABASE
homework_week1_docker_sql-pgdatabase-1  |
homework_week1_docker_sql-pgdatabase-1  |
homework_week1_docker_sql-pgdatabase-1  | /usr/local/bin/docker-entrypoint.sh: ignoring /docker-entrypoint-initdb.d/*
homework_week1_docker_sql-pgdatabase-1  |
homework_week1_docker_sql-pgdatabase-1  | 2023-01-24 08:40:19.312 UTC [49] LOG:  received fast shutdown request
homework_week1_docker_sql-pgdatabase-1  | waiting for server to shut down...2023-01-24 08:40:19.315 UTC [49] LOG:  aborting any active transactions
homework_week1_docker_sql-pgdatabase-1  | 2023-01-24 08:40:19.318 UTC [49] LOG:  background worker "logical replication launcher" (PID 56) exited with exit code 1
homework_week1_docker_sql-pgdatabase-1  | .2023-01-24 08:40:19.322 UTC [51] LOG:  shutting down
homework_week1_docker_sql-pgdatabase-1  | 2023-01-24 08:40:19.406 UTC [49] LOG:  database system is shut down
homework_week1_docker_sql-pgdatabase-1  |  done
homework_week1_docker_sql-pgdatabase-1  | server stopped
```

Figure 2: running *docker composer* in terminal

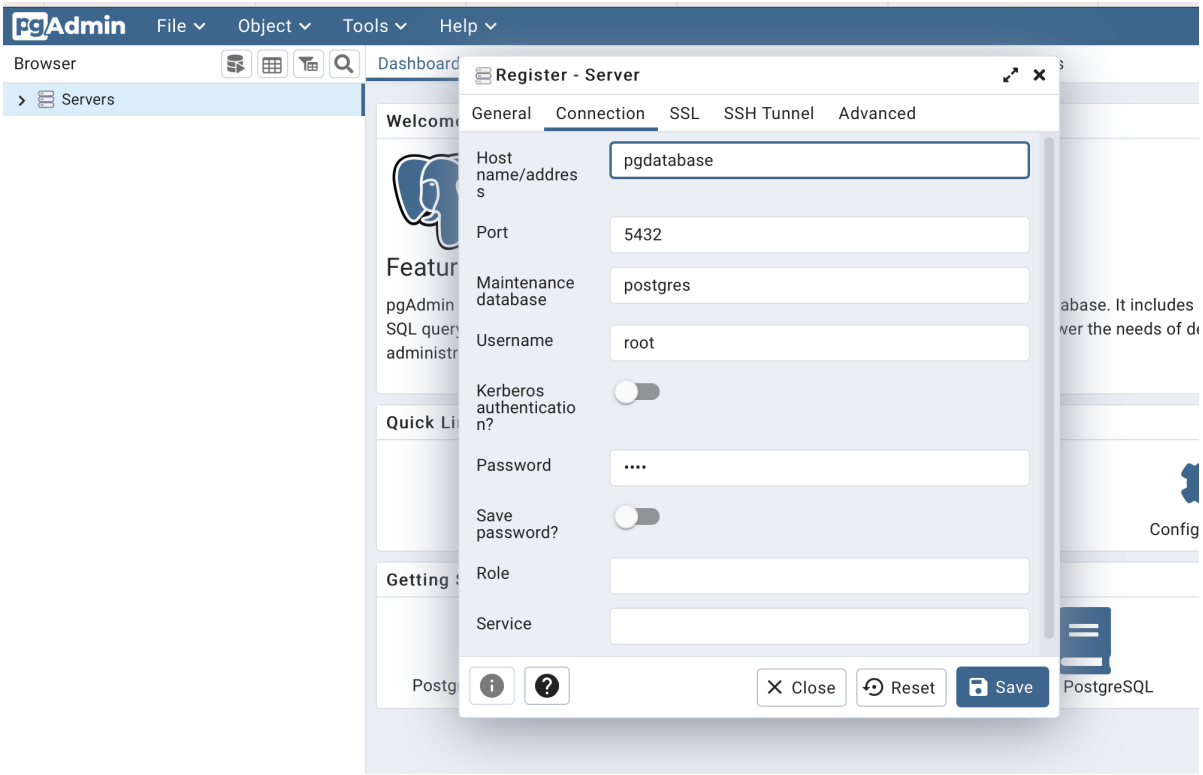
Download taxi_zone_looker.csv

```
(base) georgiosgrigoriou@Georgioss-MacBook-Pro homework_week1_docker_sql % wget https://s3.amazonaws.com/nyc-tlc/misc/taxi+_zone_lookup.csv
--2023-01-24 08:44:44--  https://s3.amazonaws.com/nyc-tlc/misc/taxi+_zone_lookup.csv
Resolving s3.amazonaws.com (s3.amazonaws.com)... 52.216.10.173, 54.231.201.8, 52.217.41.150, ...
Connecting to s3.amazonaws.com (s3.amazonaws.com)|52.216.10.173|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 12322 (12K) [application/octet-stream]
Saving to: 'taxi+_zone_lookup.csv'

taxi+_zone_lookup.c 100%[===================>]  12.03K  --.-KB/s    in 0s

2023-01-24 08:44:44 (52.7 MB/s) - 'taxi+_zone_lookup.csv' saved [12322/12322]

(base) georgiosgrigoriou@Georgioss-MacBook-Pro homework_week1_docker_sql %
```



Building and running ingest_data.py script via docker image taxi_green_ingest

```
Last login: Tue Jan 24 09:51:08 on ttys002
(base) georgiosgrigoriou@Georgioss-MacBook-Pro ~ % cd Desktop/homework_week1_docker_sql
(base) georgiosgrigoriou@Georgioss-MacBook-Pro homework_week1_docker_sql % docker network ls
NETWORK ID     NAME                                DRIVER    SCOPE
a167628bbb02   bridge                              bridge    local
7e7827ab3b33   data_engineering_zoomcamp_default   bridge    local
df950443c706   homework_week1_docker_sql_default   bridge    local
f73aba6dac50   host                                host      local
371281e23d6b   none                                null      local
3762f02dd876   pg-network                          bridge    local
(base) georgiosgrigoriou@Georgioss-MacBook-Pro homework_week1_docker_sql % docker build -t taxi_green_ingest:v01 .
[+] Building 21.8s (11/11) FINISHED
 => [internal] load build definition from Dockerfile
  => => transferring dockerfile: 264B
  => [internal] load .dockerignore
  => => transferring context: 2B
  => [internal] load metadata for docker.io/library/python:3.9.1
  => [auth] library/python:pull token for registry-1.docker.io
  => [1/5] FROM docker.io/library/python:3.9.1@sha256:ca8bd3c91af8b12c2d042ade99f7c8f578a9f80a0dbbd12ed261eeba96dd632f
  => [internal] load build context
  => => transferring context: 120B
  => CACHED [2/5] RUN apt-get install wget
  => [3/5] RUN pip install pandas sqlalchemy psycopg2-binary
  => [4/5] WORKDIR /app2
  => [5/5] COPY ingest_data.py ingest_data.py
  => exporting to image
  => => exporting layers
  => => writing image sha256:1ccb1eee6e6db69096e5d078660249c662e65238cc1e0407b8d4e1e0442ab611
  => => naming to docker.io/library/taxi_green_ingest:v01
```

Running data_ingest.py and store in Postgresql Database

```
(base) georgiosgrigoriou@Georgioss-MacBook-Pro homework_week1_docker_sql % docker run -it \
  --network=homework_week1_docker_sql_default \
  taxi_green_ingest:v01 \
    --user=root \
    --password=root \
    --host=pgdatabase \
    --port=5432 \
    --db=ny_taxi \
    --table_name=green_taxi_trips \
    --url='https://github.com/DataTalksClub/nyc-tlc-data/releases/download/green/green_tripdata_2019-01.csv.gz'
--2023-01-24 09:58:09--  https://github.com/DataTalksClub/nyc-tlc-data/releases/download/green/green_tripdata_2019-01.csv.gz
Resolving github.com (github.com)... 140.82.121.4
Connecting to github.com (github.com)|140.82.121.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/513814948/d3904232-1a2b-431b-803d-0ee802cd14fc?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIWNJYAX4CSVEH53A%2F20230124%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20230124T095810Z&X-Amz-Expires=300&X-Amz-Signature=48cae1216e09804061a1a6e76e488e3382ae3e68e9e71f0d2bdfee882da1ae83&X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=513814948&response-content-disposition=attachment%3B%20filename%3Dgreen_tripdata_2019-01.csv.gz&response-content-type=application%2Foctet-stream [following]
--2023-01-24 09:58:10--  https://objects.githubusercontent.com/github-production-release-asset-2e65be/513814948/d3904232-1a2b-431b-803d-0ee802cd14fc?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIWNJYAX4CSVEH53A%2F20230124%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20230124T095810Z&X-Amz-Expires=300&X-Amz-Signature=48cae1216e09804061a1a6e76e488e3382ae3e68e9e71f0d2bdfee882da1ae83&X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=513814948&response-content-disposition=attachment%3B%20filename%3Dgreen_tripdata_2019-01.csv.gz&response-content-type=application%2Foctet-stream
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.111.133, 185.199.109.133, 185.199.108.133, ...
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 11099245 (11M) [application/octet-stream]
Saving to: 'output.csv.gz'

output.csv.gz                       100%[===========================================================================================================>]  10.58M  16.6MB/s    in 0.6s

2023-01-24 09:58:11 (16.6 MB/s) - 'output.csv.gz' saved [11099245/11099245]

inserted another chunk, took 18.159 second
inserted another chunk, took 19.084 second
inserted another chunk, took 19.885 second
inserted another chunk, took 18.738 second
inserted another chunk, took 20.016 second
inserted another chunk, took 7.866 second
Finished ingesting data into the postgres database
```

Running taxi_zone_lookup.py and store data from taxi+_zone_lookup.csv to Postgresql Database

In [1]:
```
import pandas as pd
```

In [6]:
```
url="https://d37ci6vzurychx.cloudfront.net/misc/taxi+_zone_lookup.csv"
```

In [7]:
```
df=pd.read_csv(url)
```

In [8]:
```
df.head()
```

Out[8]:

	LocationID	Borough	Zone	service_zone
0	1	EWR	Newark Airport	EWR
1	2	Queens	Jamaica Bay	Boro Zone
2	3	Bronx	Allerton/Pelham Gardens	Boro Zone
3	4	Manhattan	Alphabet City	Yellow Zone
4	5	Staten Island	Arden Heights	Boro Zone

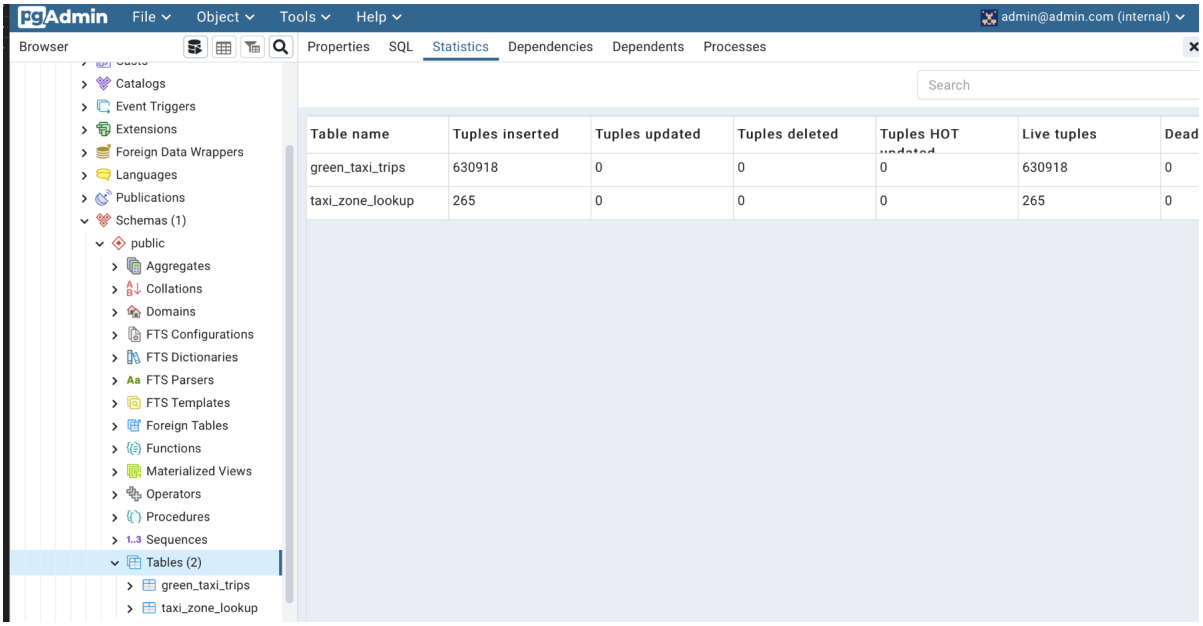
In [10]:
```
from sqlalchemy import create_engine
engine=create_engine("postgresql://root:root@localhost:5432/ny_taxi")
engine.connect()
```

Out[10]:
```
<sqlalchemy.engine.base.Connection at 0x7fe2c4cef580>
```

In [11]:
```
df.to_sql(name='taxi_zone_lookup',con=engine,if_exists='replace')
```

Out[11]:
```
265
```

pgAdmin and pgcli UI



```
Last login: Tue Jan 24 09:53:20 on ttys002
(base) georgiosgrigoriou@Georgioss-MacBook-Pro ~ % pgcli -h localhost -p 5432 -u root -d ny_taxi
Password for root:
Server: PostgreSQL 13.9 (Debian 13.9-1.pgdg110+1)
Version: 3.5.0
Home: http://pgcli.com
root@localhost:ny_taxi> \dt
+--------+------------------+-------+-------+
| Schema | Name             | Type  | Owner |
|--------+------------------+-------+-------|
| public | green_taxi_trips | table | root  |
| public | taxi_zone_lookup | table | root  |
+--------+------------------+-------+-------+
SELECT 2
Time: 0.038s
root@localhost:ny_taxi>

root@localhost:ny_taxi>
```

Question 3. Count records

How many taxi trips were totally made on January 15?

Tip: started and finished on 2019-01-15.

Remember that `lpep_pickup_datetime` and `lpep_dropoff_datetime` columns are in the format timestamp (date and hour+min+sec) and not in date.

- 20689
- 20530
- 17630
- 21090

Question 3 Count records Solution

The correct answer is 20530. *Running the following snippet in Postgresql*

```
SELECT COUNT(*) from green_taxi_trips
where date_trunc('day', lpep_pickup_datetime)='2019-01-15'
and date_trunc('day', lpep_dropoff_datetime)='2019-01-15';
```

We find out that the correct answer is 20530

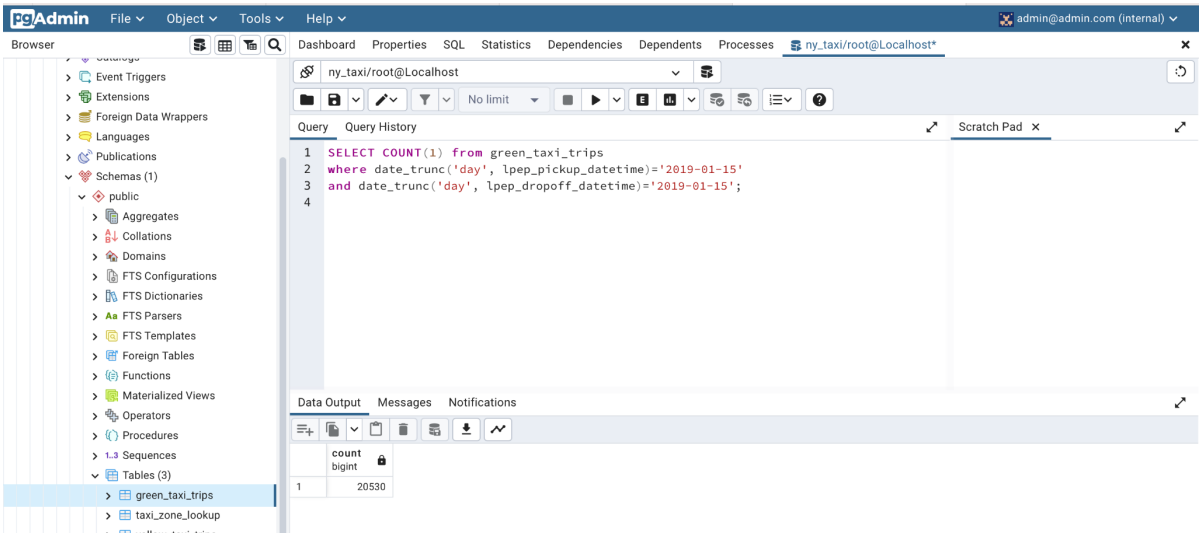


Figure 3: running the SQL command in pgAdmin

Question 4. Largest trip for each day

Which was the day with the largest trip distance

Use the pick up time for your calculations.

- 2019-01-18
- 2019-01-28
- 2019-01-15
- 2019-01-10

Question 4 Largest trip for each day

The correct answer is 2019-01-15. *Running the following snippet in Postgresql*

```
SELECT
	CAST(lpep_pickup_datetime as date) AS "day",
	trip_distance
FROM
	green_taxi_trips
GROUP BY
	CAST(lpep_pickup_datetime as date),
	trip_distance
ORDER BY
	trip_distance DESC
```

We find that the correct answer is ***2019-01-15***

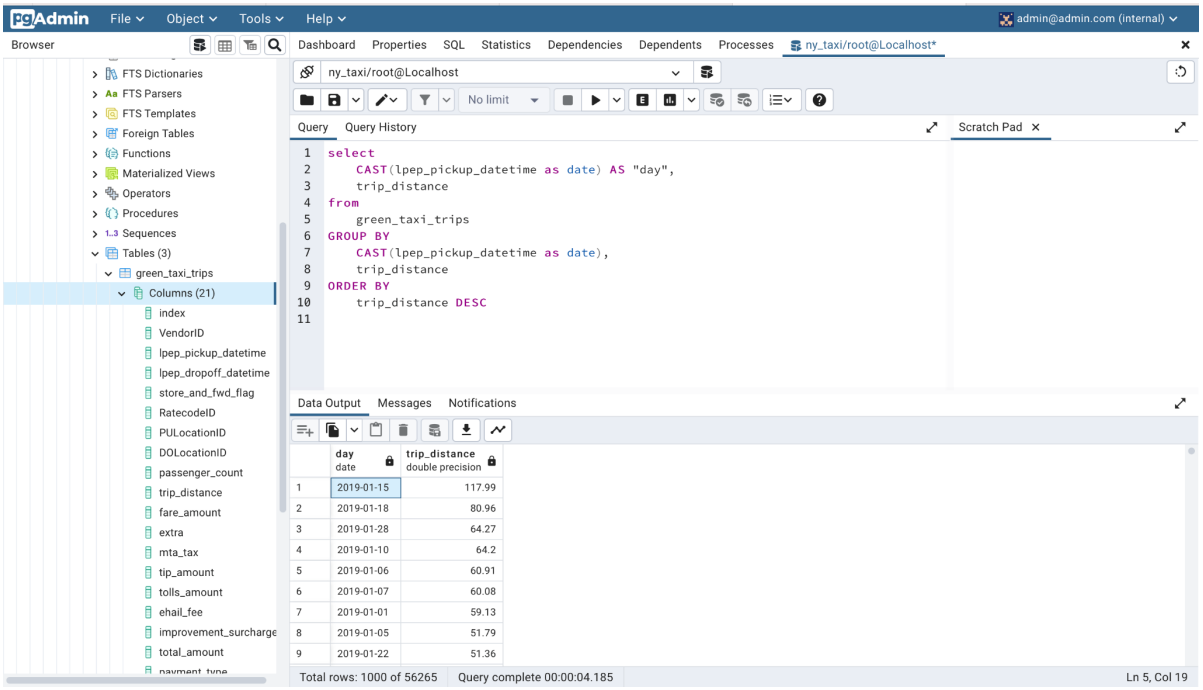


Figure 4: running the SQL command in pgAdmin

Question 5. The number of passengers

In 2019-01-01 how many trips had 2 and 3 passengers?

- 2: 1282 ; 3: 266
- 2: 1532 ; 3: 126
- 2: 1282 ; 3: 254
- 2: 1282 ; 3: 274

Question 5 number of passengers Solution

The correct answer is 2: 1282 ; 3: 254 .

Running the following snippet in Postgresql

For passenger:2

```
SELECT DATE_TRUNC('DAY',lpep_pickup_datetime),
	count(passenger_count)
FROM
		green_taxi_trips
WHERE DATE_TRUNC('DAY',lpep_pickup_datetime)='2019-01-01' and
		passenger_count=2
GROUP BY
		DATE_TRUNC('DAY',lpep_pickup_datetime)
```

For passenger:3

```
SELECT DATE_TRUNC('DAY',lpep_pickup_datetime),
       count(passenger_count)
FROM
                green_taxi_trips
WHERE DATE_TRUNC('DAY',lpep_pickup_datetime)='2019-01-01' and
          passenger_count=3
GROUP BY
          DATE_TRUNC('DAY',lpep_pickup_datetime)
```

We find out that the correct answer is **1282 and 254 respectively**

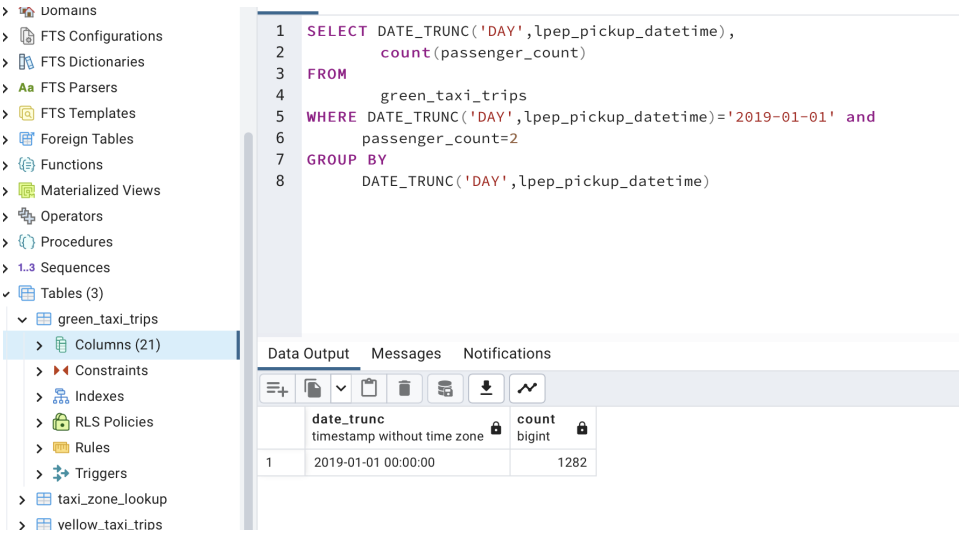


Figure 5: running the SQL command in pgAdmin

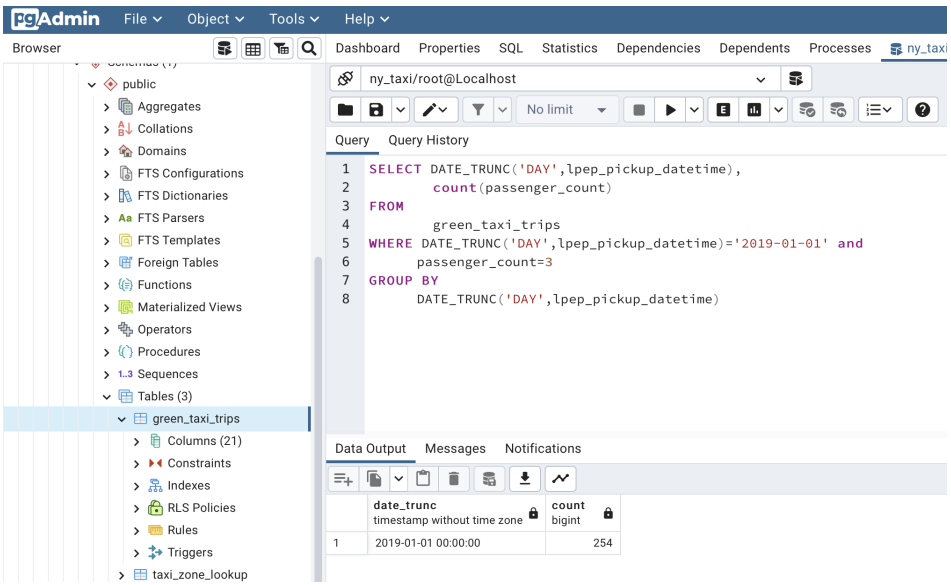


Figure 6: running the SQL command in pgAdmin

Question 6. Largest tip

For the passengers picked up in the Astoria Zone which was the drop off zone that had the largest tip? We want the name of the zone, not the id.

Note: it's not a typo, it's `tip` , not `trip`

- Central Park
- Jamaica
- South Ozone Park
- Long Island City/Queens Plaza

Question 6 Largest tip Solution

The correct answer is Long Island City/Queens Plaza. *Running the following snippet in Postgresql*

```
select
	lpep_pickup_datetime,
	lpep_dropoff_datetime,
	zpu."Zone" as pickupZone,
	zdo."Zone" as dropoffZone,
	t.tip_amount
from
 green_taxi_trips  t join taxi_zone_lookup zpu
	on t."PULocationID"=zpu."LocationID"
join taxi_zone_lookup zdo
	on	t."DOLocationID"=zdo."LocationID"
where
	zpu."Zone"='Astoria'
order by
	t.tip_amount desc
```

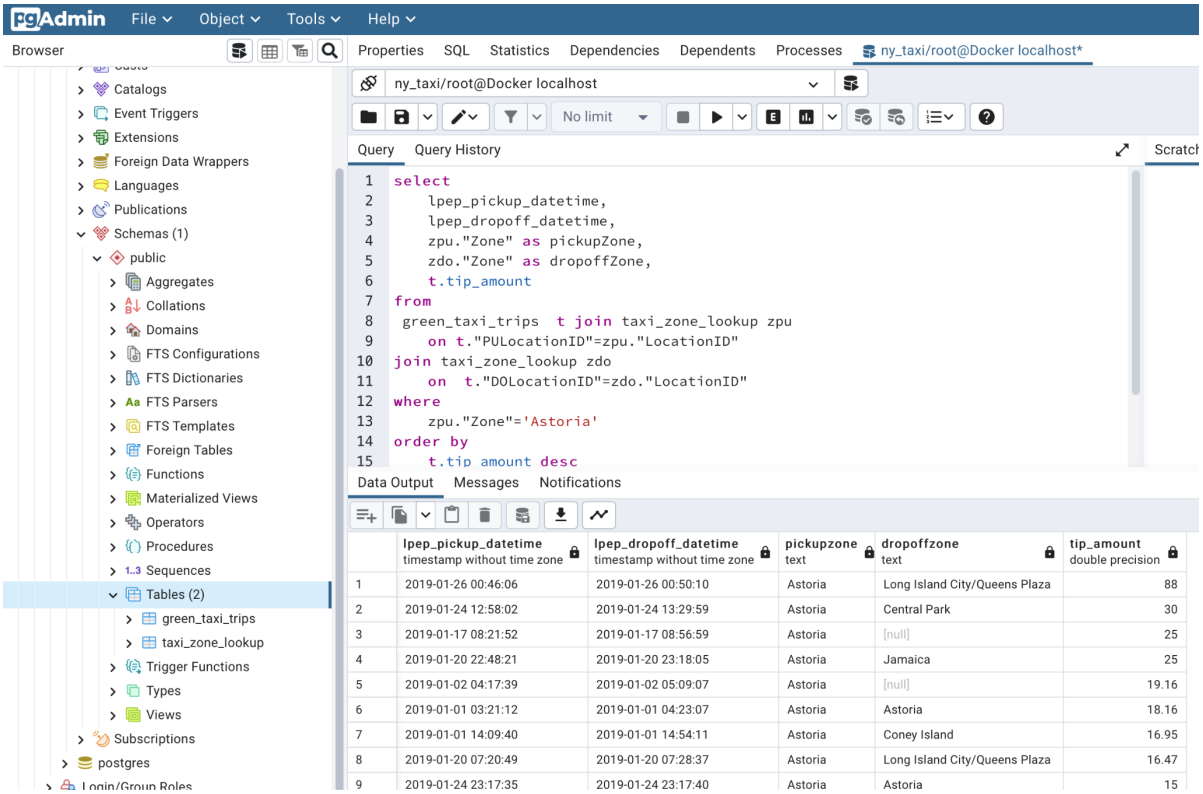


Figure 7: running the SQL command in pgAdmin