Docker build –help | grep ‘Write the image ID to the file’

```
groot@meduza:~$ docker build --help

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      --pull                    Always attempt to pull a newer version of the image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 [])
groot@meduza:~$ docker build --help | grep 'Write the image ID to the file'
      --iidfile string          Write the image ID to the file
groot@meduza:~$
```

```
groot@meduza: ~
groot@meduza:~$ sudo groupadd docker
[sudo] password for groot:
groupadd: group 'docker' already exists
groot@meduza:~$ sudo gpasswd -a groot docker
Adding user groot to group docker
groot@meduza:~$ sudo service docker restart
groot@meduza:~$ docker ps
Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock
tp://%2Fvar%2Frun%2Fdocker.sock/v1.24/containers/json": dial unix /var/run/docker.sock: connect: permiss
d
groot@meduza:~$ ^C
groot@meduza:~$ sudo chmod 666 /var/run/docker.sock
groot@meduza:~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
groot@meduza:~$ docker run -it --entrypoint bash python:3.9
root@7a39c662e0d7:/# 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@7a39c662e0d7:/#
```

```
groot@meduza: /media/groot/_data/gcpzoomcamp
groot@meduza:/media/groot/_data/gcpzoomcamp$ sudo ls -al /var/lib/postgresql/
total 28
drwxr-xr-x  5 postgres postgres 4096 Jan 20 10:19 .
drwxr-xr-x 76 root     root     4096 Dec 30 08:30 ..
drwxr-xr-x  2 postgres root     4096 Jan 20 10:19 13
drwxr-xr-x  3 postgres postgres 4096 Oct 30 21:25 14
drwxr-xr-x  3 postgres postgres 4096 Jan  1 20:49 15
-rw-------  1 postgres postgres   81 Nov 10 08:24 .bash_history
-rw-------  1 postgres postgres   48 Nov 16 11:34 .psql_history
groot@meduza:/media/groot/_data/gcpzoomcamp$ sudo chown postgres:root /var/lib/postgresql/13
groot@meduza:/media/groot/_data/gcpzoomcamp$ sudo ls -al /var/lib/postgresql/
total 28
drwxr-xr-x  5 postgres postgres 4096 Jan 20 10:19 .
drwxr-xr-x 76 root     root     4096 Dec 30 08:30 ..
drwxr-xr-x  2 postgres root     4096 Jan 20 10:19 13
drwxr-xr-x  3 postgres postgres 4096 Oct 30 21:25 14
drwxr-xr-x  3 postgres postgres 4096 Jan  1 20:49 15
-rw-------  1 postgres postgres   81 Nov 10 08:24 .bash_history
-rw-------  1 postgres postgres   48 Nov 16 11:34 .psql_history
groot@meduza:/media/groot/_data/gcpzoomcamp$ docker run -it   -e POSTGRES_USER="[redacted]"   -e POSTGRES_PASSWORD="[redacted]"   -e POSTGRES_DB="ny_taxi"   -
v /media/groot/_data/gcpzoomcamp:/var/lib/postgresql/13/data   -p 5434:5432   postgres:13
The files belonging to this database system will be owned by user "postgres".
This user must also own the server process.

The database cluster will be initialized with locale "en_US.utf8".
The default database encoding has accordingly been set to "UTF8".
The default text search configuration will be set to "english".

Data page checksums are disabled.

fixing permissions on existing directory /var/lib/postgresql/data ... ok
creating subdirectories ... ok
selecting dynamic shared memory implementation ... posix
selecting default max_connections ... 100
selecting default shared_buffers ... 128MB
selecting default time zone ... Etc/UTC
creating configuration files ... ok
running bootstrap script ... ok
performing post-bootstrap initialization ... ok
syncing data to disk ... ok

initdb: warning: enabling "trust" authentication for local connections
You can change this by editing pg_hba.conf or using the option -A, or
--auth-local and --auth-host, the next time you run initdb.
```

media > groot > _data > ingest_data.py > main

```python
    table_name = params.table_name
    filename = params.filename
    #url = params.url

    # the backup files are gzipped, and it's important to keep the correct extension
    # for pandas to be able to open the file
    #if url.endswith('.csv.gz'):
    #    csv_name = 'output.csv.gz'
    #else:
    #    csv_name = 'output.csv'

    # os.system(f"wget {url} -O {csv_name}")

    engine = create_engine(f'postgresql://{user}:{password}@{host}:{port}/{db}')

    #df_iter = pd.read_csv(csv_name, iterator=True, chunksize=100000)
    df_iter = pd.read_csv(filename, iterator=True, chunksize=100000)

    df = next(df_iter)
    clmns = df.columns

    if "lpep_pickup_datetime" in clmns and "lpep_dropoff_datetime" in clmns:
        df.lpep_pickup_datetime = pd.to_datetime(df.lpep_pickup_datetime)
        df.lpep_dropoff_datetime = pd.to_datetime(df.lpep_dropoff_datetime)

    df.head(n=0).to_sql(name=table_name, con=engine, if_exists='replace')

    df.to_sql(name=table_name, con=engine, if_exists='append')

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL CODEWHISPERER REFERENCE LOG

```
groot@meduza: ~
groot@meduza:~$ python /media/groot/_data/ingest_data.py   --user=[redacted]   --password=[redacted]   --host=localhost   --por
t=5434   --db=ny_taxi   --table_name=yellow_taxi_trips   --filename=[redacted]/Downloads/green_tripdata_2019-01.csv'
inserted another chunk, took 4.479 second
inserted another chunk, took 4.473 second
inserted another chunk, took 4.469 second
inserted another chunk, took 4.511 second
inserted another chunk, took 4.539 second
inserted another chunk, took 1.428 second
Finished ingesting data into the postgres database
groot@meduza:~$ python /media/groot/_data/ingest_data.py   --user=[redacted]   --password=[redacted]   --host=localhost   --por
t=5434   --db=ny_taxi   --table_name=taxi_zones   --filename='~/Downloads/taxi+_zone_lookup.csv'
Finished ingesting data into the postgres database
```

```sql
-- #3
-- count all trips starting and ending on the '2019.01.15'
with dates as (
    select index, date_trunc('day',lpep_pickup_datetime) as start_date, date_trunc('day', lpep_dropoff_datetime) as end_date
    from yellow_taxi_trips
),
filtered as (
    select dates.index, dates.start_date, dates.end_date
    from dates
    where dates.start_date = '2019.01.15' and dates.end_date = '2019.01.15'
)
select count(*)
from yellow_taxi_trips ytt
where ytt.index in (select f.index from filtered f)
-- count 20,530

-- #4
-- longest trips starting on the '2019.01.18','2019.01.28','2019.01.15','2019.01.10'
with dates as (
    select index, date_trunc('day',lpep_pickup_datetime) as start_date, trip_distance
    from yellow_taxi_trips
),
filtered as (
    select dates.start_date, max(trip_distance) as max_trip_distance
    from dates
    group by dates.start_date
    having dates.start_date in ('2019.01.18','2019.01.28','2019.01.15','2019.01.10')
)
select *
from yellow_taxi_trips ytt, filtered f
where ytt.trip_distance = f.max_trip_distance
-- 2019.01.18 80.96
-- 2019.01.28 64.27
-- 2019.01.15 117.99
-- 2019.01.10 64.2
```

yellow_taxi_trips 1

with dates as (select index, date_trunc(Enter a SQL expression to filter results (use Ctrl+Space)

Grid / Text

	index		lpep_pickup_datetime	lpep_dropoff_datetime			P			trip_distance	fare_amount
1	191,847	1	2019-01-10 18:58:25.000	2019-01-10 20:37:02.000	N	1	61	265	6	64.2	168
2	564,614	2	2019-01-28 21:01:59.000	2019-01-28 22:32:31.000	N	1	73	265	1	64.27	170.5
3	347,967	2	2019-01-18 07:06:27.000	2019-01-18 16:21:06.000	N	5	191	130	4	80.96	10
4	297,377	2	2019-01-15 19:27:58.000	2019-01-15 22:59:01.000	N	1	221	265	1	117.99	323

```sql
-- #5
-- in 2019-01-01 how many trips had 2 and 3 passengers?
with dates as (
    select index, passenger_count, date_trunc('day',lpep_pickup_datetime) as start_date, date_trunc('day', lpep_dropoff_datetime) as
    from yellow_taxi_trips
),
filtered as (
    select dates.index, dates.passenger_count, dates.start_date, dates.end_date
    from dates
    where dates.start_date = '2019.01.01' --and dates.end_date = '2019.01.01'
    and passenger_count in (2,3)
),
grouped as (
    select ytt.index, passenger_count
    from yellow_taxi_trips ytt
    where ytt.index in (select f.index from filtered f)
)
select passenger_count, count(*)
from grouped
group by passenger_count
-- 2 - 1282
-- 3 - 254

-- #6
-- 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.
-- PULocationID
-- DOLocationID
-- from taxi_zones LocationID, Borough, Zone (this one is the name), service_zone
```

yellow_taxi_trips 1

with dates as (select index, passenger_ Enter a SQL expression to filter results (use Ctrl+Space)

	passenger_count	count
1	2	1,282
2	3	254

```sql
-- #6
-- 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.
-- PULocationID
-- DOLocationID
-- from taxi_zones LocationID, Borough, Zone (this one is the name), service_zone
with pickups as (
    select ytt."PULocationID", max(ytt.tip_amount) as max_tip
    from yellow_taxi_trips ytt
    group by ytt."PULocationID"
    having ytt."PULocationID" in
    (select tz."LocationID"
    from taxi_zones tz
    where tz."Zone"  = 'Astoria')
),
dropoffs as (
    select ytt."index", ytt.lpep_pickup_datetime, ytt.lpep_dropoff_datetime, ytt."PULocationID", ytt."DOLocationID",
    ytt.trip_distance, ytt.fare_amount, ytt.tip_amount
    from yellow_taxi_trips ytt, pickups
    where ytt.tip_amount = pickups.max_tip and ytt."PULocationID" in (select pickups."PULocationID" from pickups)
)
select df.index, df.lpep_pickup_datetime, df.lpep_dropoff_datetime, df."PULocationID", df."DOLocationID",
tz."Zone", df.trip_distance, df.fare_amount, df.tip_amount
from dropoffs df, taxi_zones tz
where tz."LocationID" = df."DOLocationID"
-- Long Island City/Queens Plaza

--select ytt."PULocationID", max(ytt.tip_amount) as max_tip
--from yellow_taxi_trips ytt
--group by ytt."PULocationID"
```

yellow_taxi_trips(+) 1

with pickups as (select ytt."PULocation | Enter a SQL expression to filter results (use Ctrl+Space)

	123 index	lpep_pickup_datetime	lpep_dropoff_datetime	123 PULocationID	123 DOLocationID	Zone	123 trip_distanc
1	506,162	2019-01-26 00:46:06.000	2019-01-26 00:50:10.000	7	146	Long Island City/Queens Plaza	

```
mats_tumblebuns@cloudshell:~/tf-tutorial/gcpzoomcamp (gcpzoomcamp)$ terraform apply -var="project=gcpzoomcamp"
google_bigquery_dataset.dataset: Refreshing state... [id=projects/gcpzoomcamp/datasets/trips_data_all]
google_storage_bucket.data-lake-bucket: Refreshing state... [id=gcpzoomcamp]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # google_bigquery_dataset.dataset will be created
  + resource "google_bigquery_dataset" "dataset" {
      + creation_time              = (known after apply)
      + dataset_id                 = "trips_data_all"
      + delete_contents_on_destroy = false
      + etag                       = (known after apply)
      + id                         = (known after apply)
      + labels                     = (known after apply)
      + last_modified_time         = (known after apply)
      + location                   = "asia-east1"
      + project                    = "gcpzoomcamp"
      + self_link                  = (known after apply)

      + access {
          + domain         = (known after apply)
          + group_by_email = (known after apply)
          + role           = (known after apply)
          + special_group  = (known after apply)
          + user_by_email  = (known after apply)

          + dataset {
              + target_types = (known after apply)

              + dataset {
                  + dataset_id = (known after apply)
                  + project_id = (known after apply)
                }
            }

          + routine {
              + dataset_id = (known after apply)
              + project_id = (known after apply)
```

```
      + project                    = (known after apply)
      + public_access_prevention   = (known after apply)
      + self_link                  = (known after apply)
      + storage_class              = "STANDARD"
      + uniform_bucket_level_access = true
      + url                        = (known after apply)

      + lifecycle_rule {
          + action {
              + type = "Delete"
            }

          + condition {
              + age                   = 30
              + matches_prefix        = []
              + matches_storage_class = []
              + matches_suffix        = []
              + with_state            = (known after apply)
            }
        }

      + versioning {
          + enabled = true
        }

      + website {
          + main_page_suffix = (known after apply)
          + not_found_page   = (known after apply)
        }
    }

Plan: 2 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

google_bigquery_dataset.dataset: Creating...
google_storage_bucket.data-lake-bucket: Creating...
google_storage_bucket.data-lake-bucket: Creation complete after 3s [id=gcpzoomcamp]
google_bigquery_dataset.dataset: Creation complete after 3s [id=projects/gcpzoomcamp/datasets/trips_data_all]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
mats_tumblebuns@cloudshell:~/tf-tutorial/gcpzoomcamp (gcpzoomcamp)$
```