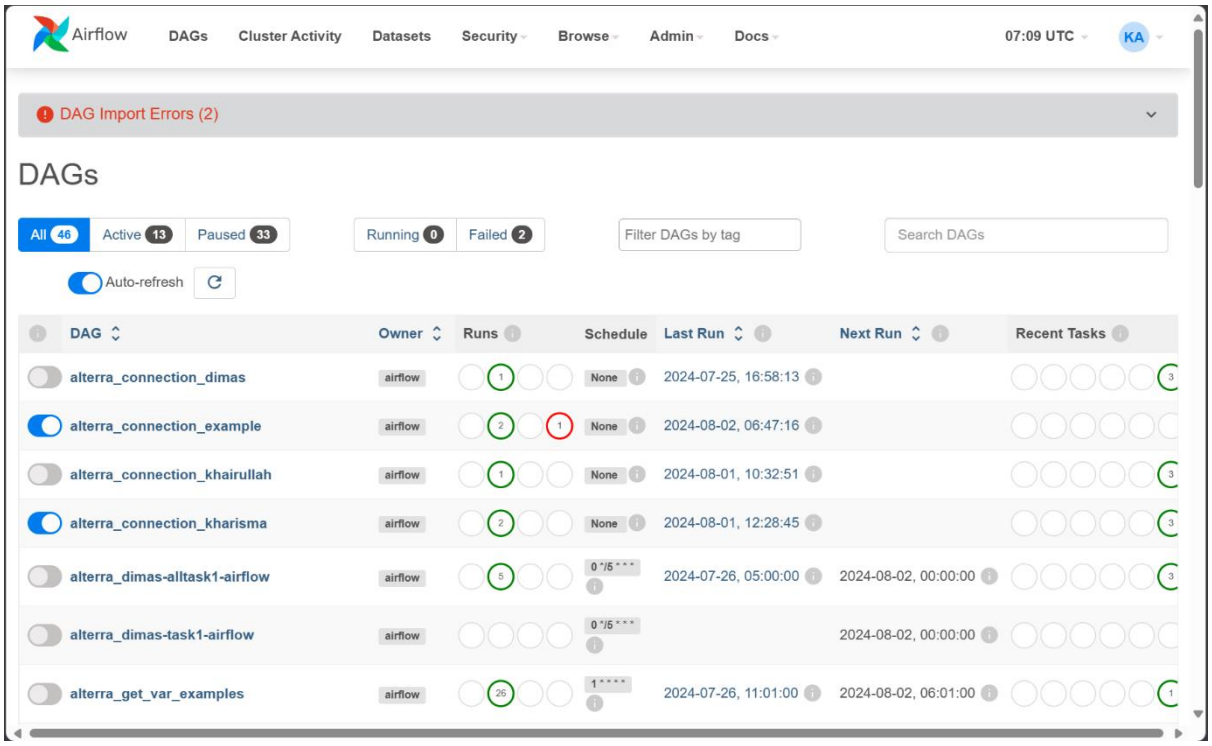
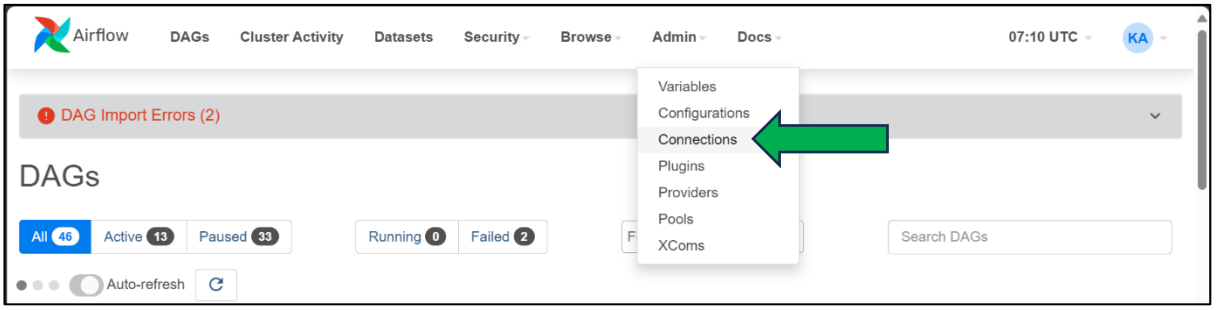
Part 2 – Airflow

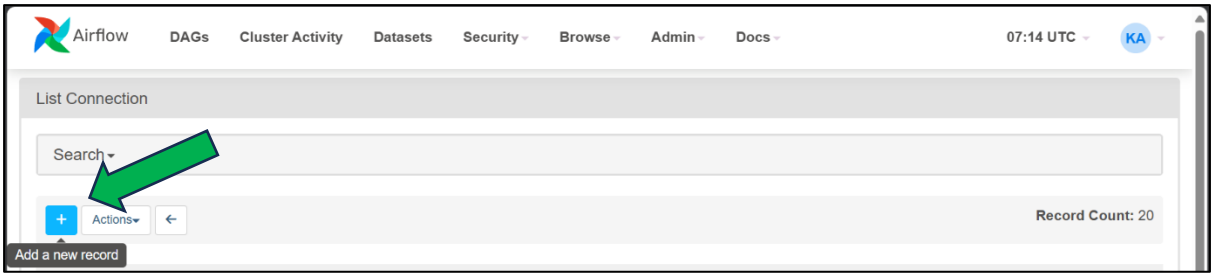
1. Masuk ke website airflow dengan memasukkan URL http://34.101.224.54:8080/.



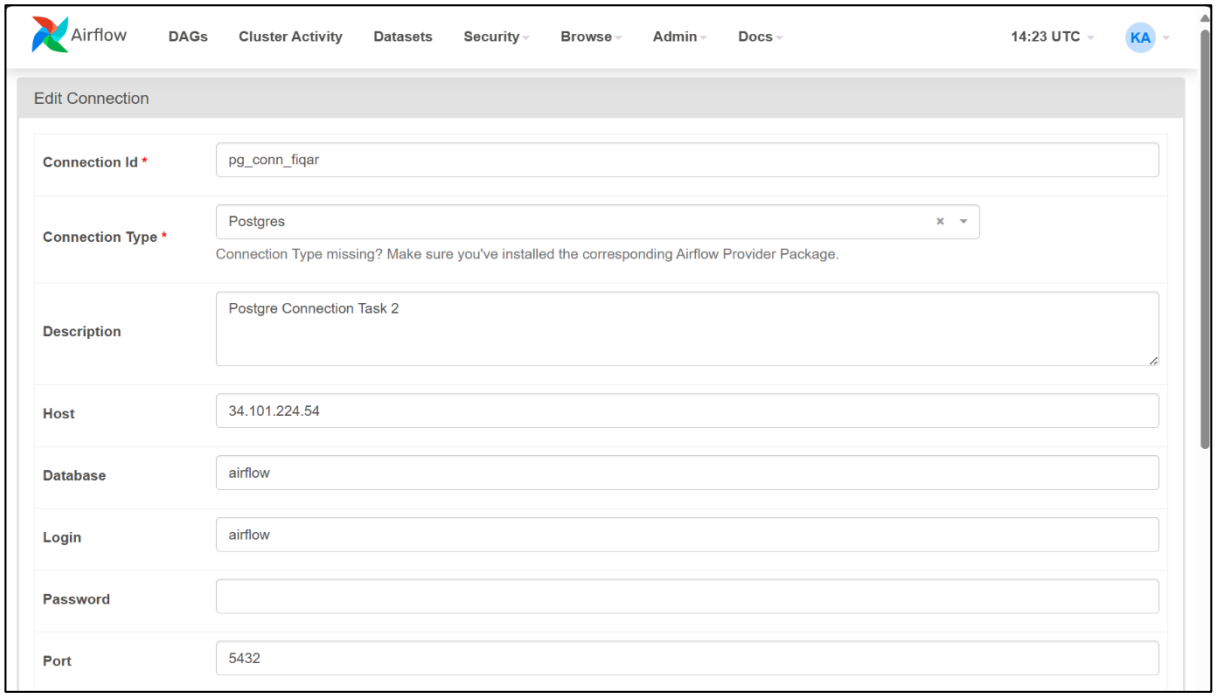
2. Kemudian pilih admin > connection.



3. Pada list connection, klik tombol plus warna biru untuk membuat connection baru.



4. Lalu isi Connection Id, Connection Type, Description, Host, Database, Login, Password, dan Port seperti gambar screenshot di bawah ini:

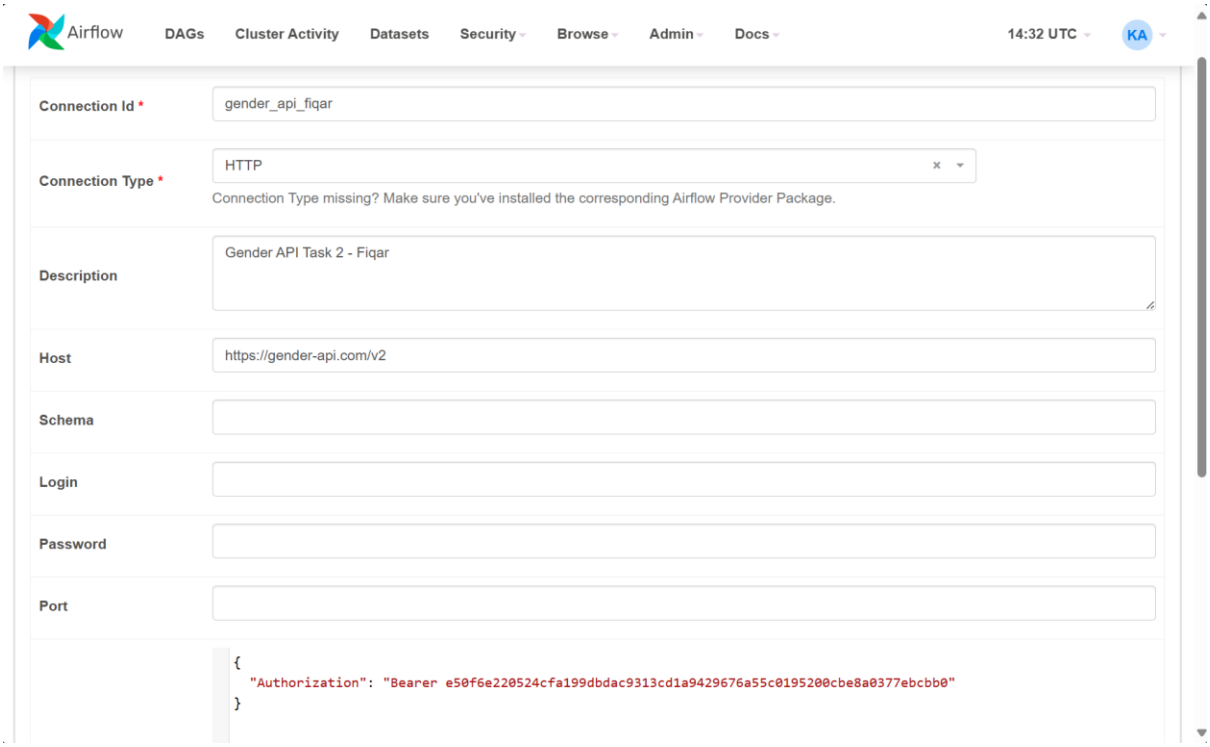


5. Setelah itu klik save untuk menyimpan hasil setting connection yang sudah dibuat.

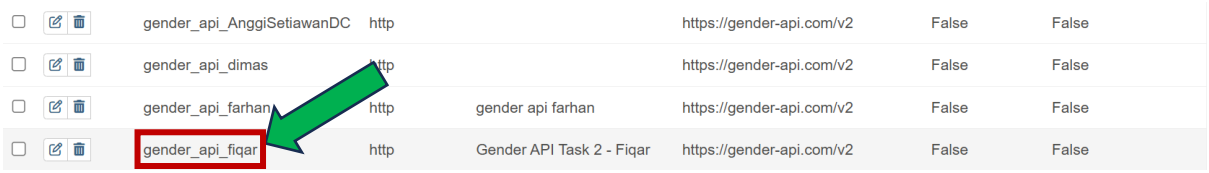
6. Connection sudah berhasil dibuat.

gender_api_kharisma	http		https://gender-api.com/v2		False	False
gender_api_rais	http		https://gender-api.com/v2		False	False
gender_api_wartadi	http	gender api by wartadi	https://gender-api.com/v2		False	False
gender_api_yovina	http		https://gender-api.com/v2		False	False
gender_id	http		https://gender-api.com/v2		False	False
pg_conn_dimas	postgres		34.101.224.54	5432	False	False
pg_conn_farhan	postgres	postgre	34.101.224.54	5432	False	False
pg_conn_fiqar	postgres	Postgre Connection Task 2	34.101.224.54	5432	False	False

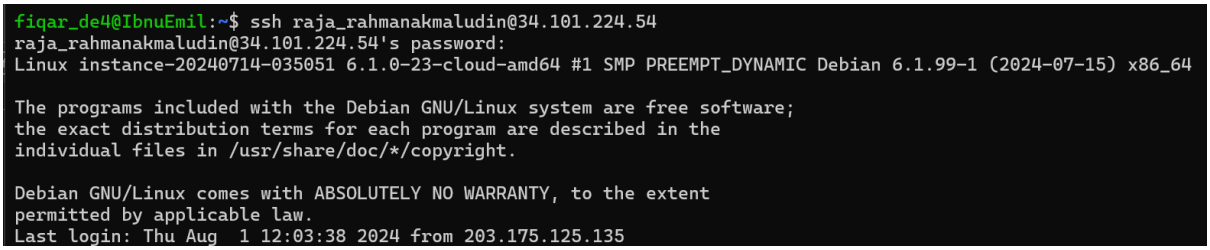
7. Lalu buatlah connection baru dengan nama connection id gender_api_fiqar dengan langkah langkah yang sama seperti sebelumnya tetapi yang diisi seperti berikut ini :



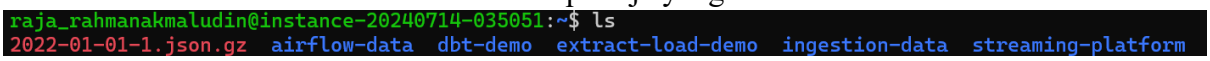
8. Connection sudah berhasil dibuat.



9. Kemudian buka aplikasi Ubuntu, lalu masukkan secure shell “ssh raja_rahmanakmaludin@34.101.224.54“ dan password “mentoralterra2024”.



10. Lalu ketik ls untuk melihat file atau folder apa saja yang ada.



11. Lalu masuk ke folder airflow-data dengan mengetik cd airflow-data dan masuk lagi ke folder docker di dalam folder tersebut dengan mengetik cd docker.

```
raja_rahmanakmaludin@instance-20240714-035051:~$ cd airflow-data
raja_rahmanakmaludin@instance-20240714-035051:~/airflow-data$ ls
TASK-1  airflow.cfg        day-2-airflow.md  day-4-airflow.md  img
TASK-2  day-1-airflow.md   day-3-airflow.md  docker            requirements.txt
raja_rahmanakmaludin@instance-20240714-035051:~/airflow-data$ cd docker
raja_rahmanakmaludin@instance-20240714-035051:~/airflow-data/docker$
```

12. Kemudian di dalam folder docker ketik ls untuk mengetahui isi dari folder docker tersebut dan masuk ke folder dags dengan mengetik cd dags.

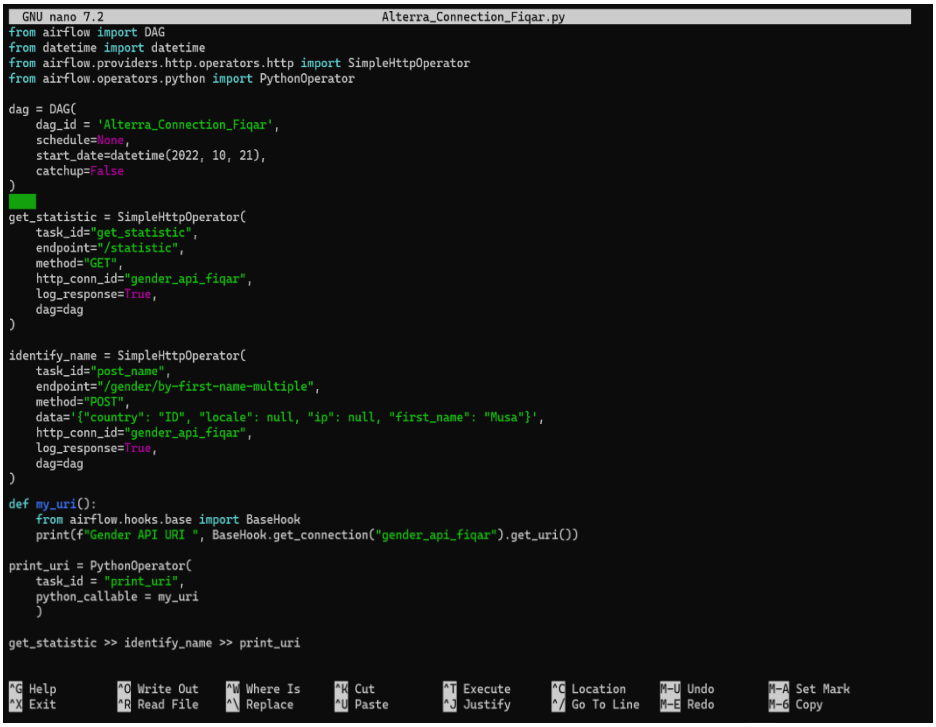
```
raja_rahmanakmaludin@instance-20240714-035051:~/airflow-data/docker$ ls
Dockerfile  config  dags  docker-compose.yaml  farhan-test1.py  farhan_test.py  logs  plugins  transformation
raja_rahmanakmaludin@instance-20240714-035051:~/airflow-data/docker$
```

13. Setelah itu masuk di folder dags.

```
raja_rahmanakmaludin@instance-20240714-035051:~/airflow-data/docker$ cd dags
raja_rahmanakmaludin@instance-20240714-035051:~/airflow-data/docker/dags$ ls
__pycache__                          etl_github_data.py            kharisma-airflow-task2.py
alterra_connection_kharisma.py       farhan_task2.py               kharisma_airflow_task1.py
alterra_connection_yovina.py         farhan_task_1.py              kharisma_airflow_task2.py
alterra_hasda_task-1.py              farhan_test.py                loop_print_var_example.py
alterra_hasda_task-2.py              fitri_airflowtask1.py         rais-alltask1-airflow.py
alterra_hasda_task2.py               fitri_airflowtask2.py         rais-alltask2-airflow.py
alterra_hook_kharisma.py             get_var_example.py            sample.csv
alterra_hook_yovina.py               hello_world.py                wartadi_task1_airflow.py
alterra_nurhasanahdarus_task1.py     hello_world_operator.py       wartadi_task1_airflow_new.py
alterra_tes_connection_khairullah.py hook_example.py               wartadi_task2_airflow.py
connection_dimas_task2.py            hook_example_dimas.py         xcom_example_decorator.py
connection_example.py                hook_example_fitri.py         xcom_example_native.py
connection_example_dimas.py          ingestion.py                  yovina_airflow_task1.py
connection_example_fitri.py          integrate_all.py              yovina_airflow_task2.py
dag_github_data.py                   integrate_all_part2.py        zola_task1_airflow.py
dimas-alltask1-airflow.py            integrate_all_part3.py        zola_task2_airflow.py
dimas-alltask2-airflow.py            khairullah_airflow_task1.py
dimas-task1-airflow.py               kharisma-airflow-task2-new.py
```

Berikut adalah isi dari folder dags. Kemudian buat file ekstensi python dengan mengetik sudo nano Alterra_Connection_Fiqar.py.

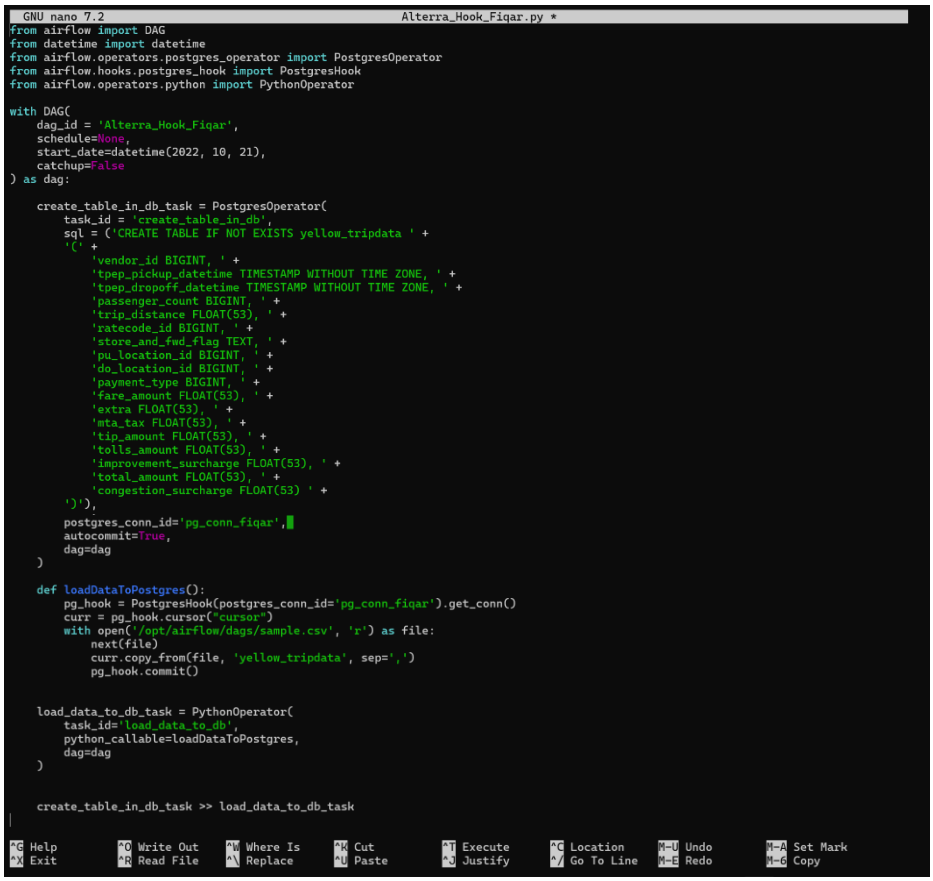
14. Lalu membuat code dalam file tersebut.



```
 GNU nano 7.2                                  Alterra_Connection_Fiqar.py
from airflow import DAG
from datetime import datetime
from airflow.providers.http.operators.http import SimpleHttpOperator
from airflow.operators.python import PythonOperator

dag = DAG(
    dag_id = 'Alterra_Connection_Fiqar',
    schedule=None,
    start_date=datetime(2022, 10, 21),
    catchup=False
)

get_statistic = SimpleHttpOperator(
    task_id="get_statistic",
    endpoint="/statistic",
    method="GET",
    http_conn_id="gender_api_fiqar",
    log_response=True,
    dag=dag
)

identify_name = SimpleHttpOperator(
    task_id="post_name",
    endpoint="/gender/by-first-name-multiple",
    method="POST",
    data='{"country": "ID", "locale": null, "ip": null, "first_name": "Musa"}',
    http_conn_id="gender_api_fiqar",
    log_response=True,
    dag=dag
)

def my_uri():
    from airflow.hooks.base import BaseHook
    print(f"Gender API URI ", BaseHook.get_connection("gender_api_fiqar").get_uri())

print_uri = PythonOperator(
    task_id = "print_uri",
    python_callable = my_uri
    )

get_statistic >> identify_name >> print_uri

^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo       M-A Set Mark
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line  M-E Redo       M-6 Copy
```

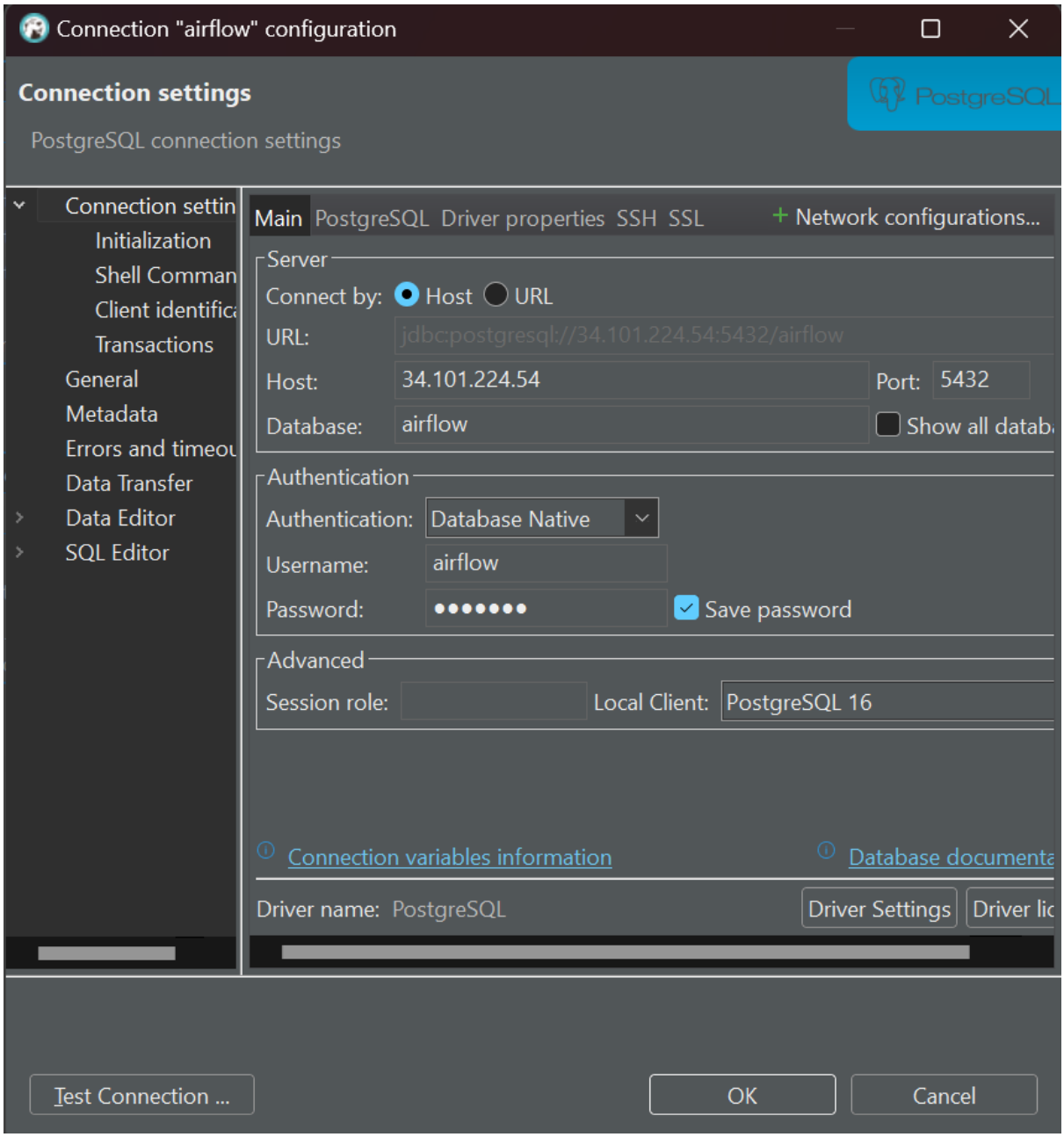
Setelah itu tekan tombol kombinasi Ctrl + X, kemudian Y, kemudian Enter untuk keluar dari mode nano.

15. Kemudian membuat code lagi dengan mengetik sudo nano Alterra_Hook_Fiqar.py.

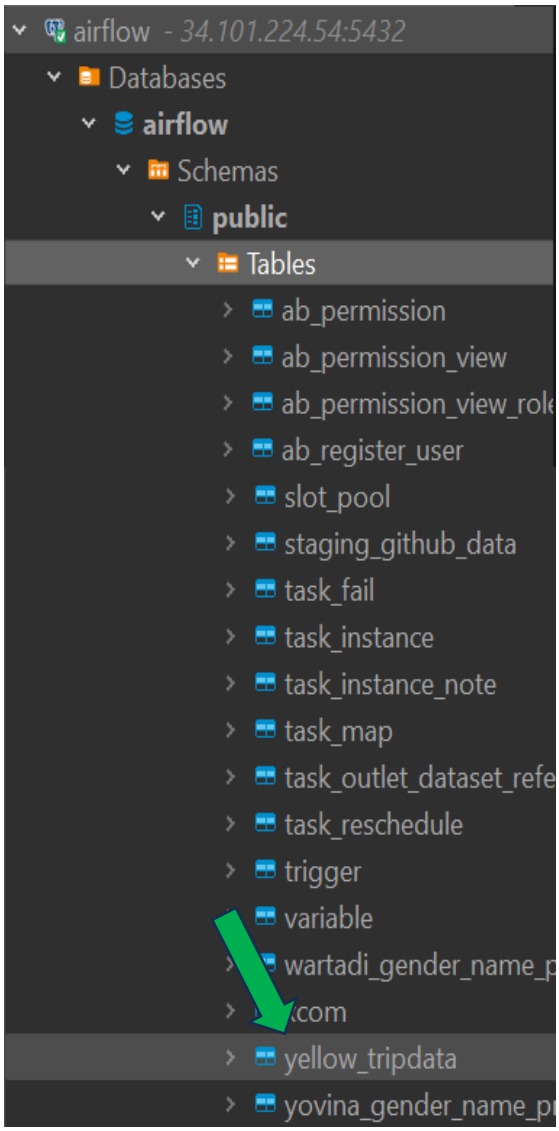


```
 GNU nano 7.2                                  Alterra_Hook_Fiqar.py *
from airflow import DAG
from datetime import datetime
from airflow.operators.postgres_operator import PostgresOperator
from airflow.hooks.postgres_hook import PostgresHook
from airflow.operators.python import PythonOperator

with DAG(
    dag_id = 'Alterra_Hook_Fiqar',
    schedule=None,
    start_date=datetime(2022, 10, 21),
    catchup=False
) as dag:

    create_table_in_db_task = PostgresOperator(
        task_id = 'create_table_in_db',
        sql = ('CREATE TABLE IF NOT EXISTS yellow_tripdata ' +
        '(' +
            'vendor_id BIGINT, ' +
            'tpep_pickup_datetime TIMESTAMP WITHOUT TIME ZONE, ' +
            'tpep_dropoff_datetime TIMESTAMP WITHOUT TIME ZONE, ' +
            'passenger_count BIGINT, ' +
            'trip_distance FLOAT(53), ' +
            'ratecode_id BIGINT, ' +
            'store_and_fwd_flag TEXT, ' +
            'pu_location_id BIGINT, ' +
            'do_location_id BIGINT, ' +
            'payment_type BIGINT, ' +
            'fare_amount FLOAT(53), ' +
            'extra FLOAT(53), ' +
            'mta_tax FLOAT(53), ' +
            'tip_amount FLOAT(53), ' +
            'tolls_amount FLOAT(53), ' +
            'improvement_surcharge FLOAT(53), ' +
            'total_amount FLOAT(53), ' +
            'congestion_surcharge FLOAT(53) ' +
        ')'),
        postgres_conn_id='pg_conn_fiqar',
        autocommit=True,
        dag=dag
    )

    def loadDataToPostgres():
        pg_hook = PostgresHook(postgres_conn_id='pg_conn_fiqar').get_conn()
        curr = pg_hook.cursor("cursor")
        with open('/opt/airflow/dags/sample.csv', 'r') as file:
            next(file)
            curr.copy_from(file, 'yellow_tripdata', sep=',')
            pg_hook.commit()


    load_data_to_db_task = PythonOperator(
        task_id='load_data_to_db',
        python_callable=loadDataToPostgres,
        dag=dag
    )


    create_table_in_db_task >> load_data_to_db_task

^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo       M-A Set Mark
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line  M-E Redo       M-6 Copy
```

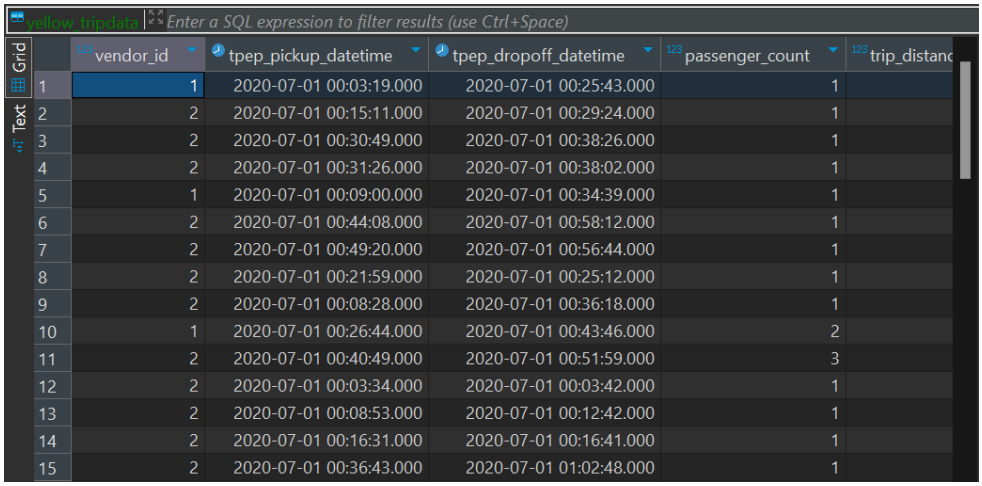
16. Kemudian masuk ke aplikasi DBeaver dan buat koneksi baru dengan memilih PostgreSQL sebagai connection, lalu isi kolom Host, Database, Username, Password, dan Port seperti gambar screenshot di bawah ini :



17. Lalu pilih connection airflow > databases > airflow > schemas > public > tables > yellow_tripdata.



18. Berikut adalah Gambaran data dari table yellow_tripdata.



yellow_tripdata — Enter a SQL expression to filter results (use Ctrl+Space)

Grid | Text

	vendor_id	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_distanc
1	1	2020-07-01 00:03:19.000	2020-07-01 00:25:43.000	1	
2	2	2020-07-01 00:15:11.000	2020-07-01 00:29:24.000	1	
3	2	2020-07-01 00:30:49.000	2020-07-01 00:38:26.000	1	
4	2	2020-07-01 00:31:26.000	2020-07-01 00:38:02.000	1	
5	1	2020-07-01 00:09:00.000	2020-07-01 00:34:39.000	1	
6	2	2020-07-01 00:44:08.000	2020-07-01 00:58:12.000	1	
7	2	2020-07-01 00:49:20.000	2020-07-01 00:56:44.000	1	
8	2	2020-07-01 00:21:59.000	2020-07-01 00:25:12.000	1	
9	2	2020-07-01 00:08:28.000	2020-07-01 00:36:18.000	1	
10	1	2020-07-01 00:26:44.000	2020-07-01 00:43:46.000	2	
11	2	2020-07-01 00:40:49.000	2020-07-01 00:51:59.000	3	
12	2	2020-07-01 00:03:34.000	2020-07-01 00:03:42.000	1	
13	2	2020-07-01 00:08:53.000	2020-07-01 00:12:42.000	1	
14	2	2020-07-01 00:16:31.000	2020-07-01 00:16:41.000	1	
15	2	2020-07-01 00:36:43.000	2020-07-01 01:02:48.000	1	

19. Lalu buat code lagi di dalam server dengan mengetik sudo nano Fiqar_Airflow_Task2.py.

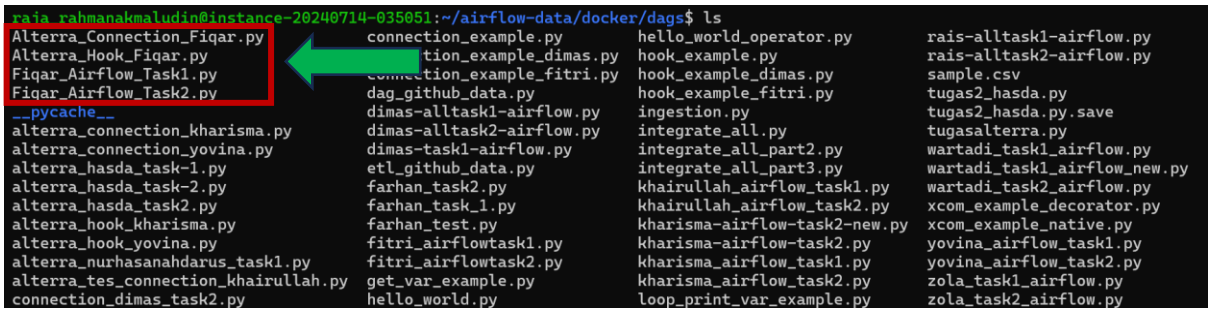
GNU nano 7.2 — Fiqar_Airflow_Task2.py

```
from datetime import datetime, timedelta
from airflow import DAG
from airflow.operators.python import PythonOperator
from airflow.providers.http.operators.http import SimpleHttpOperator
from airflow.providers.postgres.hooks.postgres import PostgresHook
from airflow.providers.postgres.operators.postgres import PostgresOperator
from airflow.utils.dates import days_ago
import json


dag = DAG(
        'Fiqar_Airflow_Task2',
        description='Alterra Airflow Task 2',
        schedule_interval='0 */5 * * *',
        start_date=datetime(2023, 10, 18),
        catchup=False
)
predict_names_task = SimpleHttpOperator(
    task_id='profile_from_gender',
    method='POST',
    http_conn_id='gender_api_fiqar',
    endpoint='/gender/by-first-name-multiple',
    headers={"Content-Type": "application/json"},
    data=json.dumps([
  {
    "first_name": "Fiqar",
    "country": "ID"
  },
  {
    "first_name": "Yahya",
    "country": "ID"
  }
]),
    response_filter=lambda response: json.loads(response.text),
    log_response=True,
    dag=dag,
)
create_table_task = PostgresOperator(
    task_id='create_table_to_postgres',
    postgres_conn_id='pg_conn_fiqar',
    sql="""
    CREATE TABLE IF NOT EXISTS fiqar_gender_name_prediction (
        input JSONB,
        details JSONB,
        result_found BOOLEAN,
        first_name VARCHAR(50),
        probability FLOAT,
        gender VARCHAR(10),
        timestamp TIMESTAMPTZ DEFAULT CURRENT_TIMESTAMP
    );
    """,
    retries=3,
    retry_delay=timedelta(minutes=5),
    dag=dag,
    autocommit=True,
)
def load_predictions_to_postgres(**kwargs):
    ti = kwargs['ti']
    predictions = ti.xcom_pull(task_ids='profile_from_gender')
    pg_hook = PostgresHook(postgres_conn_id='pg_conn_fiqar')
    for prediction in predictions:
        input_data = json.dumps(prediction['input'])
        details_data = json.dumps(prediction['details'])
        result_found = prediction['result_found']
        first_name = prediction['first_name']
        probability = prediction['probability']
        gender = prediction['gender']
        pg_hook.run("""
            INSERT INTO fiqar_gender_name_prediction (input, details, result_found, first_name, probability, gender)
            VALUES (%s, %s, %s, %s, %s, %s);
        """, parameters=(input_data, details_data, result_found, first_name, probability, gender))
load_predictions_task = PythonOperator(
    task_id='profile_gender_to_postgres',
    python_callable=load_predictions_to_postgres,
    provide_context=True,
    dag=dag,
)
predict_names_task >> create_table_task >> load_predictions_task
```

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location M-U Undo M-A Set Mark
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line M-E Redo M-6 Copy

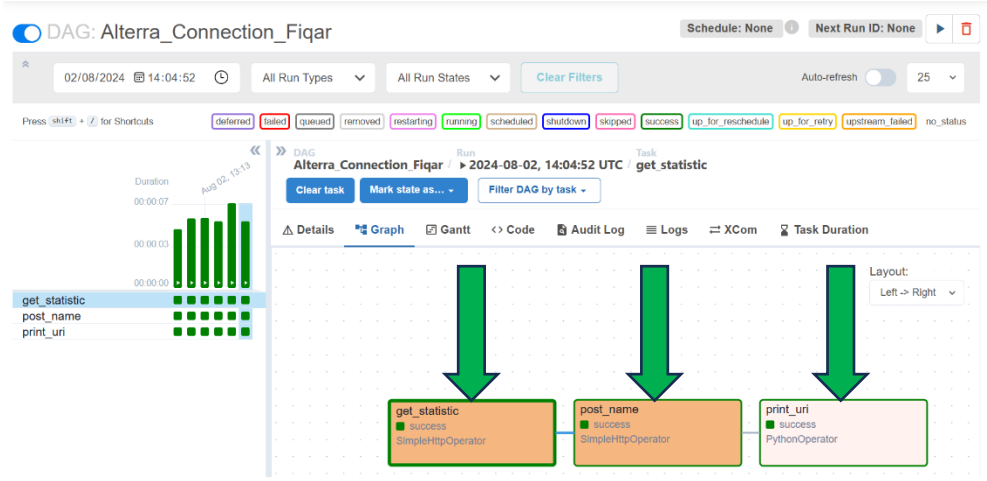
Tekan Ctrl + X, lalu Y, lalu Enter untuk keluar dari mode nano.

20. Kemudian cek semua file ekstensi python yang sudah kita buat apakah sudah ada semua atau belum. Dengan cara mengetik command ls pada folder tersebut.

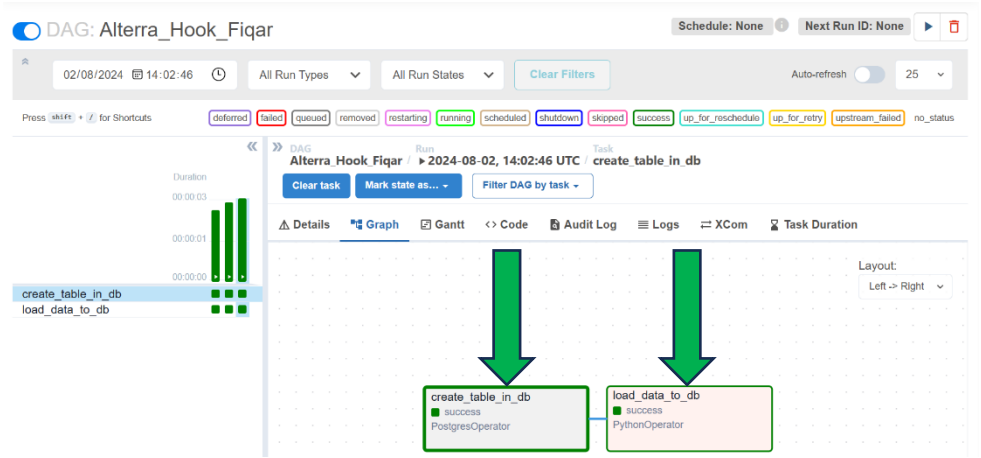


Semua file yang sudah dibuat sudah ada.

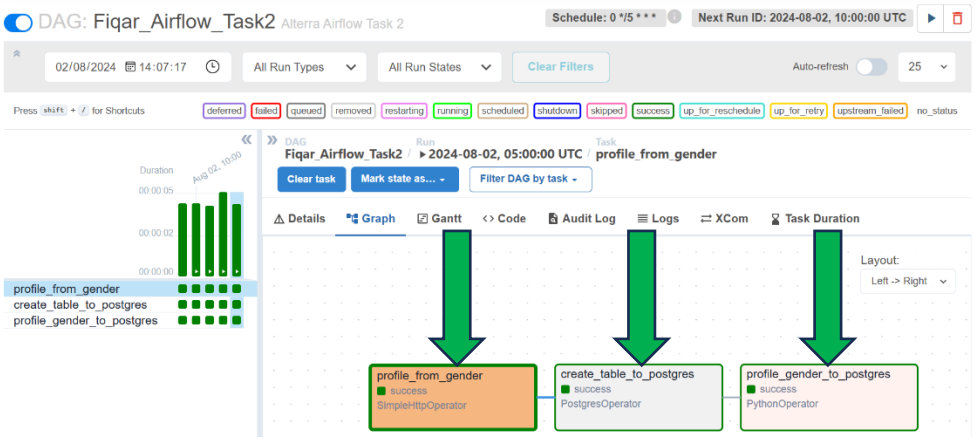
21. Sekarang mari kita cek apakah semuanya berjalan dengan normal.



DAG dengan nama dag_id “Alterra_Connection_Fiqar” berhasil dengan ditandai status success pada panah hijau pada screenshot tersebut.

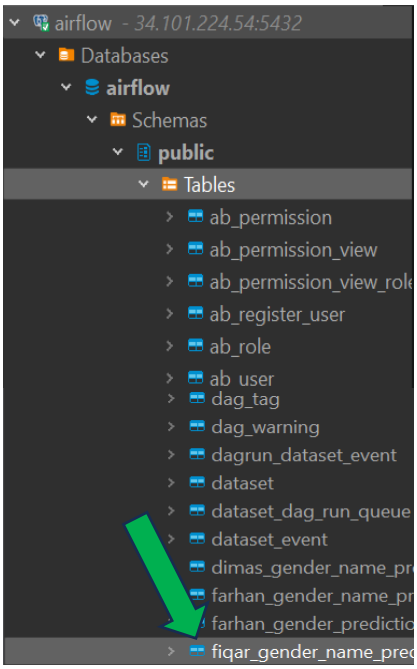


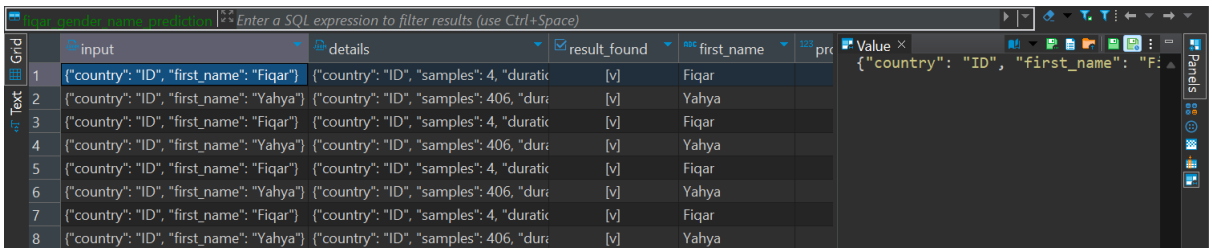
DAG dengan nama dag_id “Alterra_Hook_Fiqar” berhasil dengan ditandai status success pada panah hijau pada screenshot tersebut.



DAG dengan nama dag_id “Fiqar_Airflow_Task2” berhasil dengan ditandai status success pada panah hijau pada screenshot tersebut.

22. Langkah terakhir, mari kita cek di aplikasi DBeaver, apakah tabel dengan nama “fiqar_gender_name_prediction” sudah ada?
Mari kita buktikan dengan membuka aplikasi DBeaver, kemudian pilih connection airflow > databases > airflow > schemas > public > tables > fiqar_gender_name_prediction.





Tabel dengan nama fiqar_gender_name_prediction sudah ada di dalam database airflow.