

```
Ahmad Fathoni A@Laptoptoni MINGW64 /c/Alterra/dbt-demo-exc
$ touch docker-compose.yml
```

Membuat file docker-compose.yml dan mengisi file tersebut dengan query docker yang diperlukan

```
Ahmad Fathoni A@Laptoptoni MINGW64 /c/Alterra/dbt-demo-exc
$ docker compose up -d
[+] Running 0/0
- Network dbt-demo-exc_default      Cre[+] Running 0/2
- Network dbt-demo-exc_default      Created0.2s
[+] Running 0/3dbt-demo-exc_postgres-network Cre
- Network dbt-demo-exc_default      Created0.3s ner dbt-demo-exc_master      Cre
- Network dbt-demo-exc_postgres-network Cre[+] Running 0/4
- Network dbt-demo-exc_default      Created0.4s
- Network dbt-demo-exc_postgres-network Created0.3s
[+] Running 0/5t-demo-exc_master      Cre
- Network dbt-demo-exc_default      Created0.5s ner dbt-demo-exc_manager      Cre
```

Mengaktifkan file docker-compose.yml dengan perintah docker compose up

```
Ahmad Fathoni A@Laptoptoni MINGW64 /c/Alterra/dbt-demo-exc
$ python -m venv .env
```

Membuat/menginstall envirotnment ke dalam folder kita

```
Ahmad Fathoni A@Laptoptoni MINGW64 /c/Alterra/dbt-demo-exc
$ source .env/Scripts/activate
(.env)
```

Menjalankan evinrontment yang telah diinstall sebelumnya, selanjutnya ketika ingin menggunakan DBT kita harus menjalan envirotnment tersebut

```
Ahmad Fathoni A@Laptoptoni MINGW64 /c/Alterra/dbt-demo-exc
$ pip install dbt-postgres
Collecting dbt-postgres
  Using cached dbt_postgres-1.7.13-py3-none-any.whl (28 kB)
Collecting agate
  Using cached agate-1.10.1-py2.py3-none-any.whl (95 kB)
Collecting psycpg2-binary~=2.8
  Using cached psycpg2_binary-2.9.9-cp310-cp310-win_amd64.whl (1.2 MB)
Collecting dbt-core==1.7.13
```

Menginstall paket DBT ke dalam folder kita

```
Ahmad Fathoni A@Laptoptoni MINGW64 /c/Alterra/dbt-demo-exc
$ pip freeze | grep dbt
dbt-core==1.7.13
dbt-extractor==0.5.1
dbt-postgres==1.7.13
dbt-semantic-interfaces==0.4.4
(.env)
Ahmad Fathoni A@Laptoptoni MINGW64 /c/Alterra/dbt-demo-exc
$ pip freeze | grep dbt >> requirements.txt
```

Melakukan list terhadap paket-paket DBT yang telah diinstall dan memasukkan paket-paket tersebut ke file requirement.txt

```
Ahmad Fathoni A@Laptoptoni MINGW64 /c/Alterra/dbt-demo-exc
$ dbt init my_project
13:08:24 Running with dbt=1.7.13
13:08:24
Your new dbt project "my_project" was created!
```

Membuat folder DBT Project, nantinya seluruh hal terkait transformasi dan modeling akan dilakukan di dalam folder my\_project ini

```
(.venv)
Ahmad Fathoni A@Laptoptoni MINGW64 /c/Alterra/dbt-demo-exc
$ touch dbt-profiles/profiles.yml
```

Membuat folder DBT Profiles, dan di dalam folder tersebut memuat file profiles.yml. File profiles.yml ini akan menjadi connector antara data warehouse kita dengan DBT

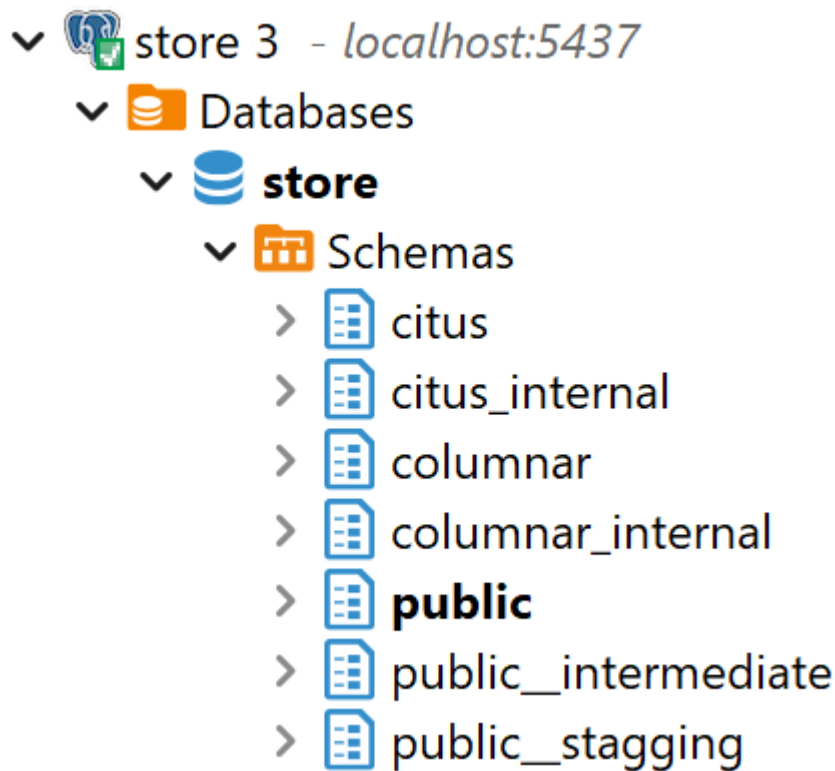
```
Ahmad Fathoni A@Laptoptoni MINGW64 /c/Alterra/dbt-demo-exc
$ export DBT_PROFILES_DIR=$(pwd)/dbt-profiles(.venv)
```

Mengexport DBT\_PROFILES\_DIR ke DBT Profiles

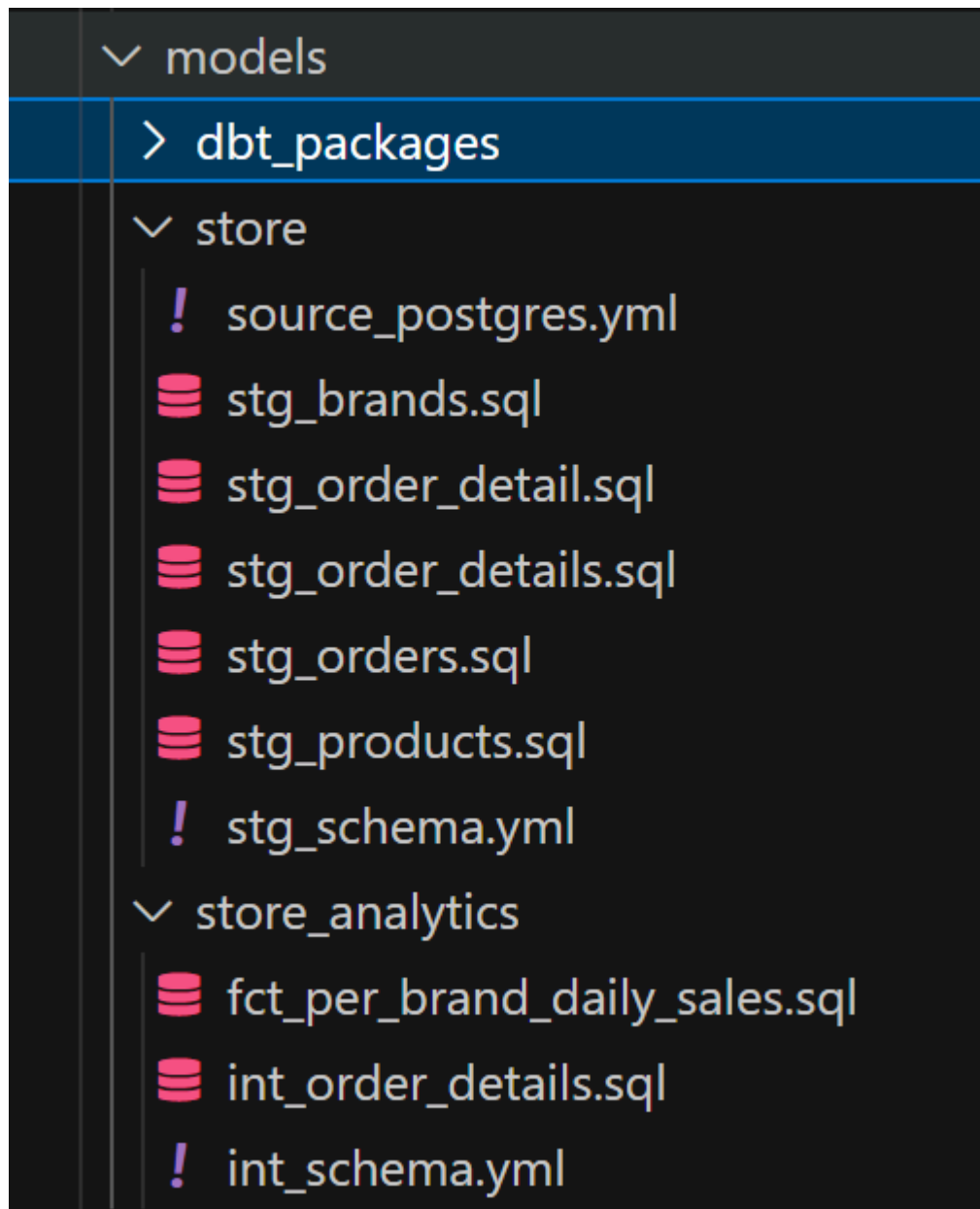
```
dbt-profiles > ! profiles.yml
1  my_project:
2    outputs:
3
4    dev:
5      type: postgres
6      threads: 1
7      host: localhost
8      port: 5437
9      user: postgres
10     pass: pass
11     dbname: store
12     schema: public
13
14   target: dev
```

Masukkan informasi data warehouse yang akan kita connectkan dengan DBT sesuai dengan format diatas

```
models:
  my_project:
    # Config indicated by + and applies to all files under models/example/
    store:
      +materialized: table
      +schema: _staging
      +database: store
    store_analytics:
      +materialized: table
      +schema: _intermediate
      +database: store
```



membuat file dbt\_project.yml pada folder my-project, pada file dbt\_project.yml berisi konfigurasi yang berfungsi untuk menentukan letak hasil model yang telah dibuat pada data warehouse kita. Berdasarkan konfigurasi diatas terdapat 2 lokasi untuk menampilkan hasil models pada data warehouse kita yaitu: public\_staging dan public\_intermediate




Setelah membuat konfigurasi, pastikan sudah ada folder yang bertugas untuk menyimpan data tranformasi di DBT kita, yaitu store dan store\_analytics. kedua folder tersebut ditaruh di sebuah folder bernama models

```

my_project > models > store > ! source_postgres.yml
1  version: 2
2
3  sources:
4    - name: store
5      database: store
6      schema: public
7
8    tables:
9      - name: brands
10        columns:
11          - name: brand_id
12            description: "Unique identifier for each brand"
13            tests:
14              - unique
15              - not_null
16          - name: name
17            description: "Name of the brand"
18            tests:
19              - not_null
20
21      - name: products
22        columns:
23          - name: product_id
24            description: "Unique identifier for each product"
25            tests:
26              - unique
27              - not_null
28          - name: brand_id
29            description: "Foreign key referencing brands"
30            tests:
31              - relationships:
32                to: source('store', 'brands')
33                field: brand_id
34          - name: name
35            description: "Name of the product"
36            tests:
37              - not_null

```

Hal pertama yang dilakukan sebelum membuat model adalah membuat file bernama `source_postgres.yml` dan ditaruh di folder `store`. File ini berfungsi untuk mendefinisikan tabel beserta setiap kolom yang akan kita transformasikan, tabel ini diambil dari data warehouse

```
my_project > models > store >  stg_brands.sql
1  select
2    brand_id::int as brand_id
3    , name as brand_name
4  from {{source('store', 'brands')}}
```

Setelah menentukan source, kita mulai membuat model. Pembentukan model dapat dilakukan dengan query sederhana seperti perintah select. Pada query diatas, kita melakukan transformasi data dari tabel brands pada file stg\_brand, kita mengganti data type kolom brand\_id menjadi interger dan mengganti nama kolom name menjadi brand\_name

```
my_project > models > store > ! stg_schema.yml
1  version: 2
2
3  models:
4    - name: stg_brands
5      description: "brand staging area "
6      columns:
7        - name: brand_id
8          tests:
9            - unique
10           - not_null
```

Setelah itu kita harus mendefinisikan model yang kita buat, selain mendefinisikan model yang telah dibuat, file ini juga akan melakukan pengetesan terhadap model. Dari contoh diatas, kita melakukan tes terhadap model stg\_brands, tes tersebut memastikan bahwa data yang terdapat di tabel tersebut selalu unik dan tidak ada yang kosong