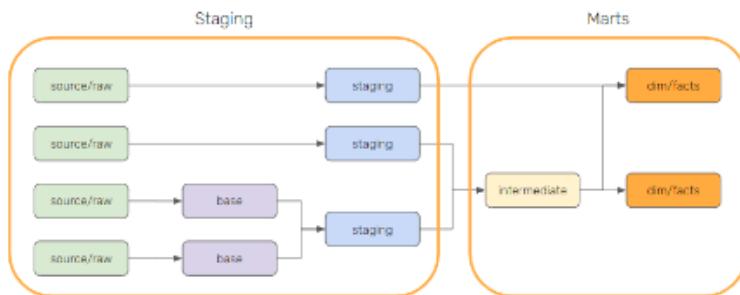


Analytics Engineering

Model Layers



จากรูป จะแบ่งเป็น 2 ส่วน คือ Staging กับ Marts ในส่วน Staging ไว้สำหรับพื้นที่ในการจัดการกับข้อมูล หรือ Source และในส่วน Marts ไว้สำหรับการนำข้อมูลจาก Staging ไปใช้งานต่อ เช่น การ Join การสร้าง Dashboard

1. การเตรียมเครื่อง

รัน คำสั่ง

```
cd 06-analytics-engineering/
```

```
docker-compose up
```

```
port 3000 —> SQLPad
```

```
SQLPAD_ADMIN: admin@swu.ac.th
```

```
SQLPAD_ADMIN_PASSWORD: admin
```

และรันคำสั่ง เตรียมEnvironment และสร้าง dbt project

```
cd 06-analytics-engineering/
```

```
python -m venv ENV
```

```
source ENV/bin/activate
```

```
dbt
```

```
dbt init
```

```
code ~/.dbt/profiles.yml
```

**** และสร้าง Folders marts/sales และ staging/jaffle ไว้สำหรับสร้างไฟล์

1. Staging

เริ่มจากการ setup source

1.1 สร้างไฟล์ stg_jaffle__customers.sql เพื่อดึงข้อมูลจากตาราง customers ใน staging/jaffle

with

```
source as (
    select * from {{ source('jaffle', 'jaffle_shop_customers') }}
)
, final as (
    select
        id
        , first_name || ' ' || last_name as name
    from source
)
select * from final
```

1.2 สร้างไฟล์เพิ่ม stg_jaffle__orders.sql เพื่อดึงข้อมูลจากตาราง orders ใน staging/jaffle

with

```
source as (
    select * from {{ source('jaffle', 'jaffle_shop_orders') }}
)
, final as (
    select
        id
        , user_id
        , order_date
        , status
    from source
)
select * from final
```

1.3 สร้างไฟล์ stg_jaffle__stripe_payments.sql เพื่อดึงข้อมูลจาก stripe_payments ใน staging/jaffle

with

```
source as (
    select * from {{ source('jaffle', 'stripe_payments') }}
)
```

, final as (

select

 id

 , order_id

 , payment_method

 , amount

 , status

 , created

 from source

)

select * from final

1.4 สร้างไฟล์ stg_models.yml เขียน Doc ของ staging ***** ใน staging/jaffle

version: 2

models:

- name: stg_jaffle_customers

description: Staging model for customers

columns:

 - name: id

 tests:

 - unique

 - not_null

 - name: name

- name: stg_jaffle_orders

description: Staging model for orders

columns:

 - name: id

 - name: user_id

 - name: order_date

 - name: status

- name: stg_jaffle_stripe_payments

description: Staging model for Stripe payments

columns:

 - name: id

 - name: order_id

 - name: payment_method

 - name: amount

 - name: status

 - name: created

2. Marts

2.1 นำไฟล์ `complete_orders.sql` ไปไว้ใน marts/sales และเขียน code ดังนี้

with

```
int_orders_customers_joined as (
    select * from {{ ref('int_orders_customers_joined') }}
)
, final as (
    select
        order_id
        , order_date
        , order_status
        , customer_name
    from int_orders_customers_joined
    where order_status = 'completed'
)
select * from final
```

2.2 สร้างไฟล์ `pending_orders.sql` ใน marts/sales

with

```
int_orders_customers_joined as (
    select * from {{ ref('int_orders_customers_joined') }}
)
, final as (
    select
        order_id
        , order_date
        , order_status
        , customer_name
    from int_orders_customers_joined
    where order_status = 'pending'
)
```

```
select * from final
```

2.3 สร้าง Folder intermediate ใน marts/sales และสร้างไฟล์

int_orders_customers_joined.sql

with

orders as (

```
    select * from {{ ref('stg_jaffle__orders') }}
```

)

, customers as (

```
    select * from {{ ref('stg_jaffle__customers') }}
```

)

, final as (

select

```
    o.id as order_id
```

```
    , o.order_date
```

```
    , o.status as order_status
```

```
    , c.name as customer_name
```

from orders as o

join customers as c

on

```
    o.user_id = c.id
```

)

```
select * from final
```

2.4 สร้างไฟล์ exposures.yml ใน mart/sales เอาไว้สร้าง Dashboard ได้

version: 2

exposures:

- name: weekly_jaffle_metrics

type: dashboard

maturity: high

url: <https://bi.tool/dashboards/1>

description: >

My cool dashboard

depends_on:

- ref('completed_orders')

owner:

name: Kan Ouivirach

email: kan@odds.team

2.5 ตรวจสอบไฟล์

assert_completed_orders_should_have_only_completed_status.sql

ใน test

select

 status

from "postgres"."public"."completed_orders"

where status != 'completed'

3. รัน DBT

To create models

dbt run

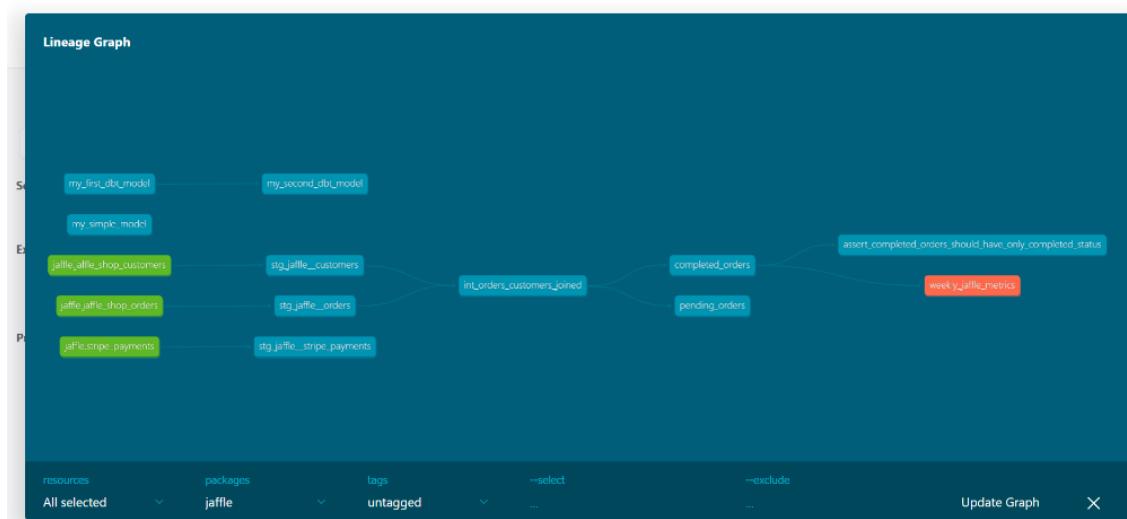
To test models

dbt test

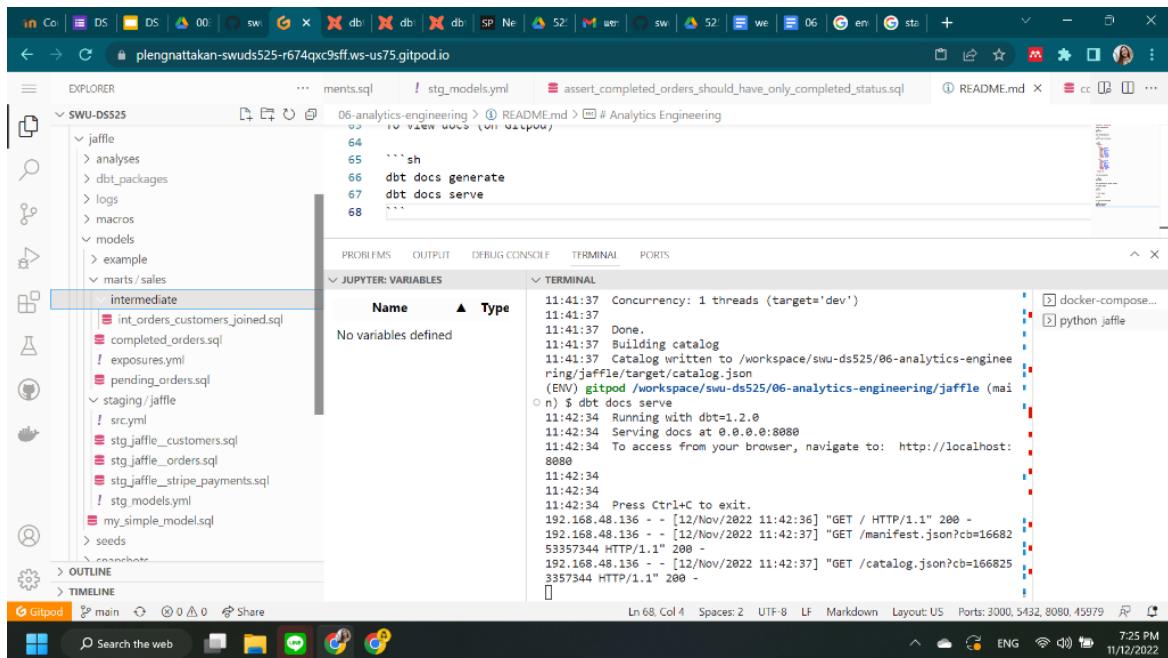
To view docs (on Gitpod)

dbt docs generate

dbt docs serve



Project : 06-analytics-engineering
Nattakan Towpunwong (64199130043)

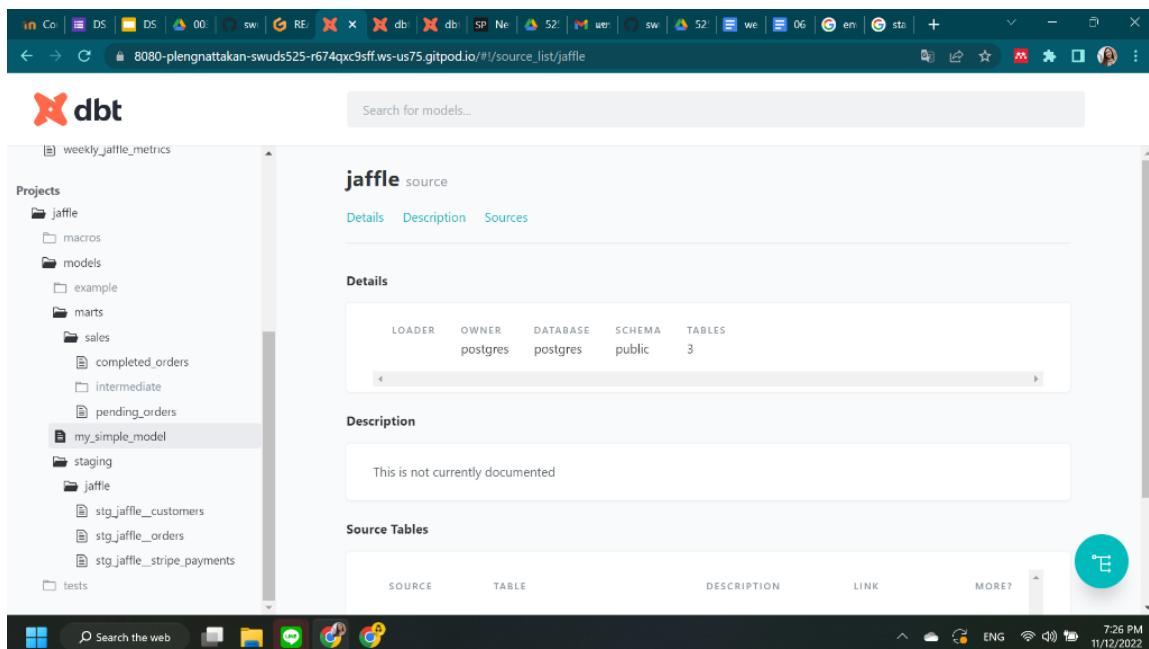


```

64
65     ``sh
66 dbt docs generate
67 dbt docs serve
68 ````

11:41:37 Concurrency: 1 threads (targets='dev')
11:41:37 Done.
11:41:37 Building catalog
11:41:37 Catalog written to /workspace/swu-ds525/06-analytics-engineering/jaffle/target/catalog.json
(ENV) gitpod /workspace/swu-ds525/06-analytics-engineering/jaffle (main)
n) $ dbt docs serve
11:42:34 Running with dbt=1.2.0
11:42:34 Serving docs at 0.0.0.0:8080
11:42:34 To access from your browser, navigate to: http://localhost:8080
11:42:34
11:42:34 Press Ctrl+C to exit.
192.168.48.136 - - [12/Nov/2022 11:42:36] "GET / HTTP/1.1" 200 -
192.168.48.136 - - [12/Nov/2022 11:42:37] "GET /manifest.json?cb=1668253357344 HTTP/1.1" 200 -
192.168.48.136 - - [12/Nov/2022 11:42:37] "GET /catalog.json?cb=1668253357344 HTTP/1.1" 200 -

```



LOADER	OWNER	DATABASE	SCHEMA	TABLES
postgres	postgres	postgres	public	3

Description

This is not currently documented

Source Tables

SOURCE	TABLE	DESCRIPTION	LINK	MORE?
--------	-------	-------------	------	-------