

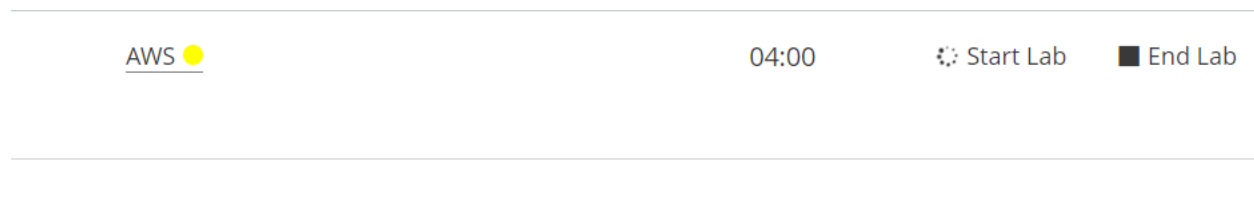
สาเหตุและปัญหาหลัก

เนื่องจากว่า ในปัจจุบันมีอัตราการเกิดอุบัติเหตุบนท้องถนนที่เกิดขึ้นบ่อยมาก แล้วสถานที่หลักๆส่วนใหญ่ ในการเกิดอุบัติเหตุบนท้องถนนคือทางด่วนที่ใช้ในจังหวัดกรุงเทพมหานคร ซึ่งเหตุผลหลักๆส่วนใหญ่ที่ทำให้เกิดอุบัติเหตุ คือ คนส่วนใหญ่ มีการขับรถเร็วที่เกินกว่าที่มาตรฐานได้มีการกำหนดหรือมีการเปลี่ยนแปลงเลนกระทันหัน โดยไม่ทันระวังหรือไม่ได้มองกระจกมองข้าง จึงทำให้เกิดอุบัติเหตุบนท้องถนน แต่อาจจะมีสาเหตุหลักๆ ที่เราไม่รู้หรือเรามองไม่เห็นสาเหตุหลักจริงๆ ที่ทำให้เกิดอุบัติเหตุบนท้องถนน เราจึงได้มีการทำ **Visualization** เพื่อหาสาเหตุหลักๆที่ทำให้เกิดอุบัติเหตุบนท้องถนน และหาแนวทางในการป้องกันและแก้ไขการเกิดอุบัติเหตุบนท้องถนน ให้มีอัตราการเกิดอุบัติเหตุในปีถัดๆ ไป ที่ลดน้อยลง

ซึ่งข้อมูลที่เรานำมาทำ **Visualization** มาจากเว็บไซต์ <https://gdcatalog.go.th/dataset/gdpublish-exat-accident>

ซึ่งมีขั้นตอนในการทำงาน ดังนี้

1. ให้ทำการคลิก **Start Lab** ขึ้นมาก่อน



2. แล้วทำการเขียนคำสั่งลงใน **console**

Write command: cat ~/.aws/credentials


```
ddd_v1_w_rsU_1383616@runweb69358:~$ cat ~/.aws/credentials
[default]
aws_access_key_id = ASIA46YTXNMJHYNM4G4F
aws_secret_access_key = 9CZvGeMh4IrAM2GJIzSmpF0nG/T0NQXIsYR3HeZA
aws_session_token = FwoGZXIvYXZlEIb////////wEaDJEjudpUhIt7vuGntiLIAWkw8i9swjQYhdnHffExsbOuZnBP
1Defgt9K049LK54Pb5k1K5E3qyIvMDgXbj3YDE0+h+1dCX0dxQ9P/T74a7MXi5VWsoZ+r23dfHQQZDn2a0VnM/Ot00Xuwm+S
riXwzfsaQQmmPdAAdzaL0XWmx2LSLCPXqkVo04aTpGQJq7vNXLt1SrkrOvkwe6npkkw5Z5Rr9DRqVjxKwJwxYSCw0gcJ9PbZ
hHKF22FcLTMyI/ER0Urjf5faGH+teZkirp3eNc9TYeRrGVOGKPPH2pwGmi3DpBuOCRNuyUtgtSKmxIujd3fB4gkOfz0NZ40s
OH0o9EwtqF83L8L11ATjAWY=
ddd_v1_w_rsU_1383616@runweb69358:~$
```

3. จากนั้นทำการคัดลอกข้อมูล `aws_access_key_id`, `aws_secret_access_key`, `aws_session_token` มาใส่ใน `code` ของเรา ในส่วนของ `function` ในการ Upload Files ขึ้นไปเก็บไว้บน S3
4. จากนั้นจะมี 2 เครื่องมือหลักๆ ที่เราต้องทำการสร้าง คือ S3 และ Redshift
5. เข้าไปที่เครื่องมือที่มีชื่อว่า S3

Console Home Info

Reset to default layout

+ Add widgets

 Introducing the new widget Applications. Find it at the bottom of your Console Home.



Recently visited Info



Amazon Redshift



S3



EMR



EC2

6. แล้วคลิกที่ปุ่ม Create a bucket

Create a bucket

Every object in S3 is stored in a bucket. To upload files and folders to S3, you'll need to create a bucket where the objects will be stored.

[Create bucket](#)

7. จากนั้นให้ใส่ชื่อ Bucket แล้วคลิกที่ปุ่ม Create bucket

Bucket Versioning

Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. [Learn more](#)

Bucket Versioning

☒ Disable
☐ Enable

Tags (0) - optional

You can use bucket tags to track storage costs and organize buckets. [Learn more](#)

No tags associated with this bucket.

Add tag

Default encryption

Automatically encrypt new objects stored in this bucket. [Learn more](#)

Server-side encryption

☒ Disable
☐ Enable

► Advanced settings

After creating the bucket you can upload files and folders to the bucket, and configure additional bucket settings.

Cancel

Create bucket

8. เมื่อทำการสร้าง Bucket เรียบร้อยแล้ว จะมี Bucket เกิดขึ้นมา ตามรูปภาพด้านล่าง

Amazon S3 > Buckets

▼ Account snapshot

View Storage Lens dashboard

Total storage
1.8 MB

Object count
223

Average object size
8.5 KB

You can enable advanced metrics in the "default-account-dashboard" configuration.

Buckets (1) Info

Copy ARN
Empty
Delete
Create bucket

Find buckets by name

< 1 >

	Name	AWS Region	Access	Creation date
<input type="radio"/>	junnieebucket	US East (N. Virginia) us-east-1	Objects can be public	December 17, 2022, 13:35:23 (UTC+07:00)

junnieebucket Info








Objects
Properties
Permissions
Metrics
Management
Access Points

Objects (7)


Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in you them permissions. [Learn more](#)

Copy S3 URI
Copy URL
Download
Open
Delete

Find objects by prefix

<input type="checkbox"/>	Name	Type	Last modified
<input type="checkbox"/>	 accident_2559.csv	csv	December 18, 2022, 08:30:30 (UTC+07:00)
<input type="checkbox"/>	 accident_2560.csv	csv	December 18, 2022, 08:30:30 (UTC+07:00)
<input type="checkbox"/>	 accident_2561.csv	csv	December 18, 2022, 08:30:30 (UTC+07:00)
<input type="checkbox"/>	 accident_2562.csv	csv	December 18, 2022, 08:30:30 (UTC+07:00)
<input type="checkbox"/>	 accident_2563.csv	csv	December 18, 2022, 08:30:31 (UTC+07:00)
<input type="checkbox"/>	 accident_2564.csv	csv	December 18, 2022, 08:30:31 (UTC+07:00)
<input type="checkbox"/>	 accident_2565.csv	csv	December 18, 2022, 08:30:31 (UTC+07:00)


9. จากนั้นเข้าไปที่เครื่องมือที่มีชื่อว่า Redshift

 Introducing the new widget Applications. Find it at the bottom of your Console Home.



Recently visited [Info](#)



 Amazon Redshift

 S3

 EMR

 EC2

10. แล้วคลิกที่ปุ่ม Create cluster

Provision and manage clusters

With a few clicks, you can create your first Amazon Redshift provisioned cluster in minutes.


[Create cluster](#)

11. ให้ทำการตั้งค่า Admin user name และ Admin user password ที่จะใช้ในระบบ Redshift

Database configurations

Admin user name

Enter a login ID for the admin user of your DB instance.

The name must be 1-128 alphanumeric characters, and it can't be a [reserved word](#) .

☐ Auto generate password

Amazon Redshift can generate a password for you, or you can specify your own password.

Admin user password

Must be 8-64 characters long. Must contain at least one uppercase letter, one lowercase letter and one number. Can be any printable ASCII character except `/`, `""`, or `@`.

☐ Show password

12. แล้วทำการคลิกที่ปุ่ม Create cluster

Additional configurations ☒ Use defaults

These configurations are optional, and default settings have been defined to help you get started with your cluster. Turn off "Use defaults" to modify these settings now.

Network Using default VPC (vpc-0d2028613da1f426d) and default subnet.	Security Using default (sg-0c05365e22a819e2f) cluster security group.
Backup Automated snapshots are created about every eight hours or following every 5 GB per node of data changes, whichever comes first.	Configuration Using default.redshift-1.0 parameter group with no database encryption.
Maintenance Using current maintenance track.	

CancelCreate cluster

13. เนื่องจากการเข้าถึง Redshift เป็นแบบ Public เราจึงต้องทำการ Modify publicly accessible setting ก่อน ถึง จะทำการเข้าใช้งาน Redshift จากภายนอกได้

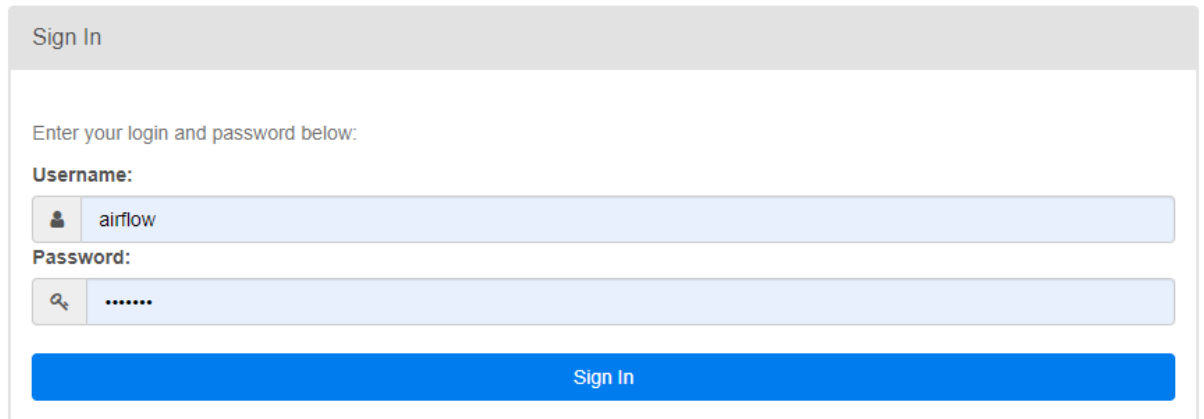
16. คลิก Open Browser โดยไปที่ port : 8080 แล้วเลือก DAGs ที่ชื่อว่า “etl” ที่เราทำการสร้างขึ้นมา

17. แล้วทำการใส่ข้อมูล ดังนี้

Username: airflow

Password: airflow

แล้วทำการคลิกปุ่ม Sign In

A screenshot of the Apache Airflow web interface's sign-in page. The page has a light gray header with the text "Sign In". Below the header, there is a white box containing the text "Enter your login and password below:". Underneath this text, there are two input fields. The first field is labeled "Username:" and contains the text "airflow". The second field is labeled "Password:" and contains a series of dots, indicating a masked password. Below these fields is a blue button with the text "Sign In".

Sign In

Enter your login and password below:

Username:

airflow

Password:

.....

Sign In

18. แล้วไปที่เมนูที่ชื่อว่า Admin แล้วเลือก Connections

19. แล้วทำการสร้าง Connections โดยการใส่ข้อมูล ดังนี้

Edit Connection

Connection Id *	my_redshift
Connection Type *	Postgres <small>Connection Type missing? Make sure you've installed the corresponding Airflow Provider Package.</small>
Description	
Host	redshift-cluster-1.cdagfejibaqn.us-east-1.redshift.amazonaws.com
Schema	dev
Login	awsuser
Password
Port	5439
Extra	

Save

Test

←

- Connection Id: ให้ใส่ชื่อ Connection Id ของเรา
- **Connection Type:** เลือกประเภท connection โดยเลือกเป็น Postgres
- **Host:** copy Endpoint ที่อยู่ใน Redshift มาใส่
- **Schema:** ให้ใส่คำว่า “dev”
- **Login:** ให้ใส่ Admin username ที่ได้มีการตั้งค่าเอาไว้ ตอนสร้าง Redshift
- **Password:** ให้ใส่ Admin user password ที่ได้มีการตั้งค่าเอาไว้ ตอนสร้าง Redshift
- **Port:** ให้ใส่เลข “5439”

20. แล้วทำการคลิกที่ปุ่ม **Test** เพื่อทดสอบการ **connection**

Connection successfully tested

Edit Connection

Connection Id *	my_redshift
Connection Type *	Postgres <small>Connection Type missing? Make sure you've installed the corresponding Airflow Provider Package.</small>
Description	
Host	redshift-cluster-1.cdagfejibaqn.us-east-1.redshift.amazonaws.com
Schema	dev
Login	awsuser
Password	*****
Port	5439
Extra	

Save Test ←

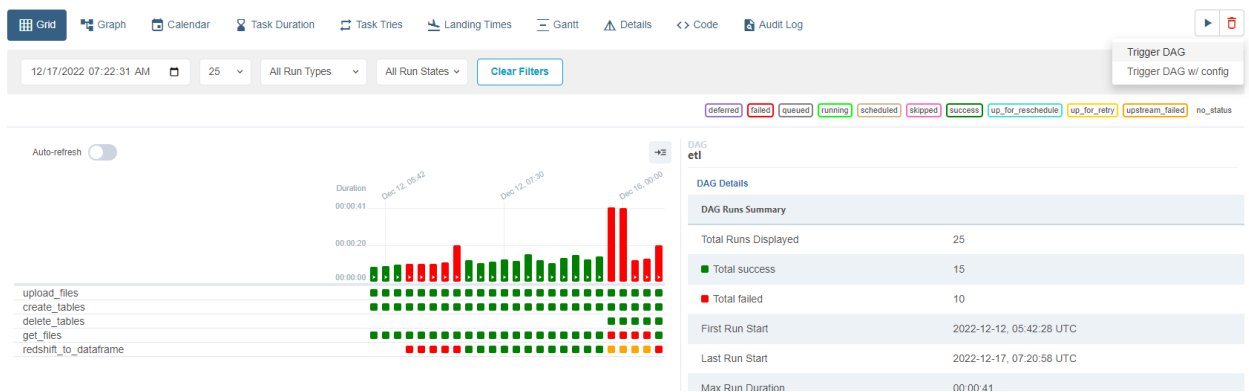
21. เมื่อทำการทดสอบ **Connection** ผ่านแล้ว ให้คลิกที่ปุ่ม **Save**

22. แล้วให้คลิกเข้าไปที่ **DAGs** ของเรา ที่มีชื่อว่า “etl”

DAGs

All 44		Active 1	Paused 43	Filter DAGs
DAG	Owner	Runs		
<div><div></div><div>dataset_consumes_1</div><div>consumes dataset-scheduled</div></div>	airflow	<div><div></div><div></div><div></div><div></div></div>		
<div><div></div><div>dataset_consumes_1_and_2</div><div>consumes dataset-scheduled</div></div>	airflow	<div><div></div><div></div><div></div><div></div></div>		
<div><div></div><div>dataset_consumes_1_never_scheduled</div><div>consumes dataset-scheduled</div></div>	airflow	<div><div></div><div></div><div></div><div></div></div>		
<div><div></div><div>dataset_consumes_unknown_never_scheduled</div><div>dataset-scheduled</div></div>	airflow	<div><div></div><div></div><div></div><div></div></div>		
<div><div></div><div>dataset_produces_1</div><div>dataset-scheduled produces</div></div>	airflow	<div><div></div><div></div><div></div><div></div></div>		
<div><div></div><div>dataset_produces_2</div><div>dataset-scheduled produces</div></div>	airflow	<div><div></div><div></div><div></div><div></div></div>		
<div><div></div><div>etl</div><div>workshop</div></div>	airflow	<div><div></div><div>21</div><div></div><div>57</div></div>		

23. แล้วลองทำการ run DAGs



```

upload_files = PythonOperator(
    task_id="upload_files",
    python_callable=_upload_files,
)

create_tables = PythonOperator(
    task_id="create_tables",
    python_callable=_create_tables,
)

delete_tables = PythonOperator(
    task_id="delete_tables",
    python_callable=_delete_tables,
)

get_files = PythonOperator(
    task_id="get_files",
    python_callable=_get_files,
)

redshift_to_dataframe = PythonOperator(
    task_id="redshift_to_dataframe",
    python_callable=_redshift_to_dataframe,
)

upload_files >> create_tables >> delete_tables >> get_files >> redshift_to_dataframe

```

ซึ่ง **Process** การทำงาน **DAGs** ของเรา จะประกอบด้วย 5 ขั้นตอนหลัก ๆ ดังนี้

1. Upload_files

คือ การ **Upload Files** ข้อมูลจากเครื่องคอมพิวเตอร์ของเราที่อยู่บน **Airflow** เอาขึ้นไปเก็บไว้บน **S3**

```

def _upload_files():

    aws_access_key_id = "ASIA46YTXNWJMAF6LMDS"
    aws_secret_access_key = "NMbmE1DJXoCuwt32ftTwKfq1Mpb05xYCN9wT0V"
    aws_session_token = "FwoGZXIvYXZlEBMaDM34gwp8FTEATq1UayLIAyA61/WNhVUuAmJ9WAHaDaf0j+Pi2IpMkzP3XtHU8hI2Lq/GGQ7LhWE

    s3 = boto3.resource(
        "s3",
        aws_access_key_id=aws_access_key_id,
        aws_secret_access_key=aws_secret_access_key,
        aws_session_token=aws_session_token
    )

    s3.meta.client.upload_file(
        "/opt/airflow/dags/data/accident_2559.csv",
        "junnieebucket",
        "accident_2559.csv",
    )

    s3.meta.client.upload_file(
        "/opt/airflow/dags/data/accident_2560.csv",
        "junnieebucket",
        "accident_2560.csv",
    )

    s3.meta.client.upload_file(
        "/opt/airflow/dags/data/accident_2561.csv",
        "junnieebucket",
        "accident_2561.csv",
    )

    s3.meta.client.upload_file(
        "/opt/airflow/dags/data/accident_2562.csv",
        "junnieebucket",
        "accident_2562.csv",
    )

    s3.meta.client.upload_file(
        "/opt/airflow/dags/data/accident_2563.csv",
        "junnieebucket",
        "accident_2563.csv",
    )

    s3.meta.client.upload_file(
        "/opt/airflow/dags/data/accident_2564.csv",
        "junnieebucket",
        "accident_2564.csv",
    )

```

2. Create_tables

คือ การสร้าง **Table Accidents** ต่าง ๆ รอไว้ใน **Redshift** เพื่อที่จะทำการ **insert** ข้อมูลจาก **S3** เข้ามาเก็บไว้ใน **Redshift**

```

def _create_tables():
    hook = PostgresHook(postgres_conn_id="my_redshift")
    conn = hook.get_conn()
    cur = conn.cursor()

    table_create_accident_2559 = """
        CREATE TABLE IF NOT EXISTS accident_2559 (
            accident_date VARCHAR(10),
            accident_time VARCHAR(10),
            expw_step VARCHAR(255),
            weather_state VARCHAR(255),
            injur_man int,
            injur_femel int,
            dead_man int,
            dead_femel int,
            cause VARCHAR(255)
        )
    """

    table_create_accident_2560 = """
        CREATE TABLE IF NOT EXISTS accident_2560 (
            accident_date VARCHAR(10),
            accident_time VARCHAR(10),
            expw_step VARCHAR(255),
            weather_state VARCHAR(255),
            injur_man int,
            injur_femel int,
            dead_man int,
            dead_femel int,
            cause VARCHAR(255)
        )
    """

    table_create_accident_2561 = """
        CREATE TABLE IF NOT EXISTS accident_2561 (
            accident_date VARCHAR(10),
            accident_time VARCHAR(10),
            expw_step VARCHAR(255),
            weather_state VARCHAR(255),
            injur_man int,
            injur_femel int,
            dead_man int,
            dead_femel int,
            cause VARCHAR(255)
        )
    """

```

3. delete_tables

คือ การลบข้อมูลที่อยู่ใน Table Accidents ต่าง ๆ ออกให้หมด หากมีข้อมูลเดิมที่มีอยู่แล้ว

```
def _delete_tables():
    hook = PostgresHook(postgres_conn_id="my_redshift")
    conn = hook.get_conn()
    cur = conn.cursor()

    table_drop_accident_2559 = "DELETE FROM accident_2559"
    table_drop_accident_2560 = "DELETE FROM accident_2560"
    table_drop_accident_2561 = "DELETE FROM accident_2561"
    table_drop_accident_2562 = "DELETE FROM accident_2562"
    table_drop_accident_2563 = "DELETE FROM accident_2563"
    table_drop_accident_2564 = "DELETE FROM accident_2564"
    table_drop_accident_2565 = "DELETE FROM accident_2565"

    drop_table_queries = [
        table_drop_accident_2559,
        table_drop_accident_2560,
        table_drop_accident_2561,
        table_drop_accident_2562,
        table_drop_accident_2563,
        table_drop_accident_2564,
        table_drop_accident_2565,
    ]

    for query in drop_table_queries:
        cur.execute(query)
        conn.commit()
```

4. get_files

คือ การ copy ไฟล์ Accidents ต่าง ๆ ที่ถูกเก็บอยู่ใน S3 ทั้งหมด ขึ้นไปเก็บไว้บน Redshift


```

def _get_files():
    hook = PostgresHook(postgres_conn_id="my_redshift")
    conn = hook.get_conn()
    cur = conn.cursor()

    copy_table_accident_2559 = """
        COPY accident_2559 FROM 's3://junnieebucket/accident_2559.csv'
        ACCESS_KEY_ID 'ASIA46YTXNWJMAF6LMDS'
        SECRET_ACCESS_KEY 'NMBmnE1DJXoCuwt32ftTwKfq1Mpbr05xYCN9wT0V'
        SESSION_TOKEN 'FwoGZXIvYXdzEBMaDM34gwp8FTEATq1UayLIAYA6l/WNhVUhAmJ9WAHaDaf0j+Pi2IpMkzP3XtHU8hI2Lq.
        CSV
        IGNOREHEADER 1
        REGION 'us-east-1'
        """

    copy_table_accident_2560 = """
        COPY accident_2560 FROM 's3://junnieebucket/accident_2560.csv'
        ACCESS_KEY_ID 'ASIA46YTXNWJMAF6LMDS'
        SECRET_ACCESS_KEY 'NMBmnE1DJXoCuwt32ftTwKfq1Mpbr05xYCN9wT0V'
        SESSION_TOKEN 'FwoGZXIvYXdzEBMaDM34gwp8FTEATq1UayLIAYA6l/WNhVUhAmJ9WAHaDaf0j+Pi2IpMkzP3XtHU8hI2Lq.
        CSV
        IGNOREHEADER 1
        REGION 'us-east-1'
        """

    copy_table_accident_2561 = """
        COPY accident_2561 FROM 's3://junnieebucket/accident_2561.csv'
        ACCESS_KEY_ID 'ASIA46YTXNWJMAF6LMDS'
        SECRET_ACCESS_KEY 'NMBmnE1DJXoCuwt32ftTwKfq1Mpbr05xYCN9wT0V'
        SESSION_TOKEN 'FwoGZXIvYXdzEBMaDM34gwp8FTEATq1UayLIAYA6l/WNhVUhAmJ9WAHaDaf0j+Pi2IpMkzP3XtHU8hI2Lq.
        CSV
        IGNOREHEADER 1
        REGION 'us-east-1'
        """

    copy_table_accident_2562 = """
        COPY accident_2562 FROM 's3://junnieebucket/accident_2562.csv'
        ACCESS_KEY_ID 'ASIA46YTXNWJMAF6LMDS'
        SECRET_ACCESS_KEY 'NMBmnE1DJXoCuwt32ftTwKfq1Mpbr05xYCN9wT0V'
        SESSION_TOKEN 'FwoGZXIvYXdzEBMaDM34gwp8FTEATq1UayLIAYA6l/WNhVUhAmJ9WAHaDaf0j+Pi2IpMkzP3XtHU8hI2Lq.
        CSV
        IGNOREHEADER 1
        REGION 'us-east-1'
        """

```

5. redshift_to_dataframe

แล้วเมื่อทำการ **transform** ข้อมูลใน **Dbt** เรียบร้อยแล้ว

ขั้นตอนสุดท้าย ก็จะมีการ **Export** ข้อมูลออกมาเป็นไฟล์ **.csv** เพื่อนำไปใช้ในการสร้าง **Visualization**

ต่อไป

```
def _redshift_to_dataframe():

    # Get data from Redshift
    hook = PostgresHook(postgres_conn_id="my_redshift")
    conn = hook.get_conn()
    cur = conn.cursor()

    table_select_events_accidents_total = """ SELECT * FROM events_accidents_total """
    cur.execute(table_select_events_accidents_total)
    df = pd.DataFrame(cur.fetchall())
    df.to_csv (r'/opt/airflow/dags/data/download/events_accidents_total.csv', index = False)

    table_select_events_accidents_count_total = """ SELECT * FROM events_accidents_count_total """
    cur.execute(table_select_events_accidents_count_total)
    df = pd.DataFrame(cur.fetchall())
    df.to_csv (r'/opt/airflow/dags/data/download/events_accidents_count_total.csv', index = False)
```

24. หลังจากนั้น เมื่อกระบวนการทำงานของ **Airflows** เสร็จสิ้น เราจะมาทำการ **transform** ข้อมูลของเราใน

Dbt

25. โดยไปที่ **Gitpod.io** ที่อยู่บน **Browser**

Write command: **pip install dbt-core dbt-redshift**

Write command: **dbt init**

```
• (EHW) gitpod /workspace/swu-ds525/08-capstone-project (main) $ dbt init
05:55:10 Running with dbt=1.3.1
Enter a name for your project (letters, digits, underscore): accident
Which database would you like to use?
[1] postgres
[2] redshift

(Don't see the one you want? https://docs.getdbt.com/docs/available-adapters)

Enter a number: 2
host (hostname.region.redshift.amazonaws.com): redshift-cluster-1.cdagfejibagn.us-east-1.redshift.amazonaws.com
port [5439]:
user (dev username): awsuser
[1] password
[2] iam
Desired authentication method option (enter a number): 1
password (dev password):
dbname (default database that dbt will build objects in): dev
schema (default schema that dbt will build objects in): public
threads (1 or more) [1]: 1
05:55:59 Profile accident written to /home/gitpod/.dbt/profiles.yml using target's profile_template.yml and your supplied values. Run 'dbt debug' to validate the connection.
05:55:59
Your new dbt project "accident" was created!

For more information on how to configure the profiles.yml file,
please consult the dbt documentation here:

https://docs.getdbt.com/docs/configure-your-profile

One more thing:

Need help? Don't hesitate to reach out to us via GitHub issues or on Slack:

https://community.getdbt.com/

Happy modeling!
```

```
etl.py M ! profiles.yml X
home > gitpod > .dbt > ! profiles.yml > {} accident
1 accident:
2   outputs:
3     dev:
4       dbname: dev
5       host: redshift-cluster-1.cdagfejibaqn.us-east-1.redshift.amazonaws.com
6       password: Krue#55zz
7       port: 5439
8       schema: public
9       threads: 1
10      type: redshift
11      user: awsuser
12    target: dev
13
```

ภาพนี้ จะเป็นภาพหน้าต่างไฟล์ที่มีชื่อว่า “profiles.yml” ที่สร้างขึ้นมาจาก Dbt

Write command: dbt debug

```
• (ENV) gitpod /workspace/swu-ds525/08-capstone-project/accident (main) $ dbt debug
06:01:05 Running with dbt=1.3.1
dbt version: 1.3.1
python version: 3.8.13
python path: /workspace/swu-ds525/08-capstone-project/ENV/bin/python
os info: Linux-5.15.0-47-generic-x86_64-with-glibc2.29
Using profiles.yml file at /home/gitpod/.dbt/profiles.yml
Using dbt_project.yml file at /workspace/swu-ds525/08-capstone-project/accident/dbt_project.yml

Configuration:
  profiles.yml file [OK found and valid]
  dbt_project.yml file [OK found and valid]

Required dependencies:
- git [OK found]

Connection:
  host: redshift-cluster-1.cdagfejibaqn.us-east-1.redshift.amazonaws.com
  port: 5439
  user: awsuser
  database: dev
  schema: public
  search_path: None
  keepalives_idle: 240
  sslmode: None
  method: database
  cluster_id: None
  iam_profile: None
  iam_duration_seconds: 900
  Connection test: [OK connection ok]

All checks passed!
```

Write command: dbt run

```

● (ENV) gitpod /workspace/swu-ds525/08-capstone-project/accident (main) $ dbt run
06:00:14 Running with dbt=1.3.1
06:00:14 Found 2 models, 4 tests, 0 snapshots, 0 analyses, 327 macros, 0 operations, 0 seed files, 0 sources, 0 exposures, 0 metrics
06:00:14 Concurrency: 1 threads (target='dev')
06:00:16
06:00:16 1 of 2 START sql table model public.my_first_dbt_model ..... [RUN]
06:00:19 1 of 2 OK created sql table model public.my_first_dbt_model ..... [SELECT in 2.61s]
06:00:19 2 of 2 START sql view model public.my_second_dbt_model ..... [RUN]
06:00:20 2 of 2 OK created sql view model public.my_second_dbt_model ..... [CREATE VIEW in 1.79s]
06:00:21
06:00:21 Finished running 1 table model, 1 view model in 0 hours 0 minutes and 7.41 seconds (7.41s).
06:00:21
06:00:21 Completed successfully
06:00:21
06:00:21 Done. PASS=2 WARN=0 ERROR=0 SKIP=0 TOTAL=2
○ (ENV) gitpod /workspace/swu-ds525/08-capstone-project/accident (main) $ 

```

26. เมื่อทำการสร้างไฟล์ Dbt เสร็จเรียบร้อยแล้ว ต่อมาจะมาทำการสร้าง Staging ต่างๆ โดยใช้ command ดังนี้

```

select
    to_date(accident_date, 'YYYYMMDD', FALSE) AS accident_date ,
    accident_time ,
    expw_step ,
    weather_state ,
    injur_man ,
    injur_femel ,
    dead_man ,
    dead_femel ,
    cause
from accident_2559
where accident_date IS NOT NULL
and accident_time IS NOT NULL
and expw_step IS NOT NULL

```

```

✓ select
    to_date(accident_date, 'YYYYMMDD', FALSE) AS accident_date ,
    accident_time ,
    expw_step ,
    weather_state ,
    injur_man ,
    injur_femel ,
    dead_man ,
    dead_femel ,
    cause
from accident_2560
where accident_date IS NOT NULL
and accident_time IS NOT NULL
and expw_step IS NOT NULL

```

```
select
    to_date(accident_date, 'YYYYMMDD', FALSE) AS accident_date ,
    accident_time ,
    expw_step ,
    weather_state ,
    injur_man ,
    injur_femal ,
    dead_man ,
    dead_femal ,
    cause
from accident_2561
where accident_date IS NOT NULL
and    accident_time IS NOT NULL
and    expw_step IS NOT NULL
```

```
select
    to_date(accident_date, 'YYYYMMDD', FALSE) AS accident_date ,
    accident_time ,
    expw_step ,
    weather_state ,
    injur_man ,
    injur_femal ,
    dead_man ,
    dead_femal ,
    cause
from accident_2562
where accident_date IS NOT NULL
and    accident_time IS NOT NULL
and    expw_step IS NOT NULL
```

```
select
    to_date(accident_date, 'YYYYMMDD', FALSE) AS accident_date ,
    accident_time ,
    expw_step ,
    weather_state ,
    injur_man ,
    injur_femel ,
    dead_man ,
    dead_femel ,
    cause
from accident_2563
where accident_date IS NOT NULL
and    accident_time IS NOT NULL
and    expw_step IS NOT NULL
```

```
select
    to_date(accident_date, 'YYYYMMDD', FALSE) AS accident_date ,
    accident_time ,
    expw_step ,
    weather_state ,
    injur_man ,
    injur_femel ,
    dead_man ,
    dead_femel ,
    cause
from accident_2564
where accident_date IS NOT NULL
and    accident_time IS NOT NULL
and    expw_step IS NOT NULL
```

```
✓ select
    to_date(accident_date, 'YYYYMMDD', FALSE) AS accident_date ,
    accident_time ,
    expw_step ,
    weather_state ,
    injur_man ,
    injur_femel ,
    dead_man ,
    dead_femel ,
    cause
from accident_2565
where accident_date IS NOT NULL
and    accident_time IS NOT NULL
and    expw_step IS NOT NULL
```

Write command: dbt run

```
• (ENV) gitpod /workspace/swu-ds525/08-capstone-project/accidents (main) $ dbt run
02:22:28 Running with dbt=1.3.1
02:22:28 Found 10 models, 4 tests, 0 snapshots, 0 analyses, 327 macros, 0 operations, 0 seed files, 0 sources, 0 exposures, 0 metrics
02:22:28
02:22:30 Concurrency: 1 threads (target='dev')
02:22:30
02:22:30 1 of 9 START sql table model public.my_first_dbt_model ..... [RUN]
02:22:33 1 of 9 OK created sql table model public.my_first_dbt_model ..... [SELECT in 2.72s]
02:22:33 2 of 9 START sql view model public.stg_accident_2559 ..... [RUN]
02:22:36 2 of 9 OK created sql view model public.stg_accident_2559 ..... [CREATE VIEW in 2.54s]
02:22:36 3 of 9 START sql view model public.stg_accident_2560 ..... [RUN]
02:22:38 3 of 9 OK created sql view model public.stg_accident_2560 ..... [CREATE VIEW in 2.36s]
02:22:38 4 of 9 START sql view model public.stg_accident_2561 ..... [RUN]
02:22:40 4 of 9 OK created sql view model public.stg_accident_2561 ..... [CREATE VIEW in 2.34s]
02:22:40 5 of 9 START sql view model public.stg_accident_2562 ..... [RUN]
02:22:43 5 of 9 OK created sql view model public.stg_accident_2562 ..... [CREATE VIEW in 2.45s]
02:22:43 6 of 9 START sql view model public.stg_accident_2563 ..... [RUN]
02:22:45 6 of 9 OK created sql view model public.stg_accident_2563 ..... [CREATE VIEW in 2.30s]
02:22:45 7 of 9 START sql view model public.stg_accident_2564 ..... [RUN]
02:22:47 7 of 9 OK created sql view model public.stg_accident_2564 ..... [CREATE VIEW in 2.41s]
02:22:47 8 of 9 START sql view model public.stg_accident_2565 ..... [RUN]
02:22:50 8 of 9 OK created sql view model public.stg_accident_2565 ..... [CREATE VIEW in 2.42s]
02:22:50 9 of 9 START sql view model public.my_second_dbt_model ..... [RUN]
02:22:52 9 of 9 OK created sql view model public.my_second_dbt_model ..... [CREATE VIEW in 1.79s]
02:22:52
02:22:52 Finished running 1 table model, 8 view models in 0 hours 0 minutes and 24.47 seconds (24.47s).
02:22:52 Completed successfully
02:22:52
02:22:52 Done. PASS=9 WARN=0 ERROR=0 SKIP=0 TOTAL=9
```

24. แล้วมาดูผลลัพธ์ของ Staging ทั้งหมด ที่ถูกสร้างขึ้นมาใน Redshift

The screenshot shows the AWS Redshift console interface. On the left, there's a sidebar with 'Select database' (dev) and 'Select schema' (public). Below that, a list of tables is shown, including staging tables like 'stg_accident_2559' through 'stg_accident_2564'. The main panel displays the SQL query for 'stg_accident_2565' and the resulting data table.

Query:

```
1 select
2   accident_date ,
3   accident_time ,
4   expw_step ,
5   weather_state ,
6   injur_man ,
7   injur_femel ,
8   dead_man ,
9   dead_femel ,
10  cause
11 from stg_accident_2565
```

Query results: Query 925, Completed, started on December 18, 2022 at 09:18:19, ELAPSED TIME: 00 m 08 s

Rows returned (667)

accident_date	accident_time	expw_step	weather_state	injur_man	injur_femel	dead_man	dead_femel	cause
2022-01-05	05:15:15	ตำรวจ	ปกติ	0	0	0	0	ผลสาธิตจากการขับขี่
2022-01-05	06:44:44	ตำรวจ	ปกติ	0	0	0	0	ลื่นหลุด
2022-01-05	20:33:33	ตำรวจ	ปกติ	0	1	0	0	ชนรถกระบะคันนี้

25. แล้วทำการสร้าง Model ที่มีชื่อว่า “events_accidents_total” โดยใช้ command ดังนี้

```

select
    accident_date ,
    accident_time ,
    expw_step ,
    weather_state ,
    injur_man ,
    injur_femel ,
    dead_man ,
    dead_femel ,
    cause
from {{ ref('stg_accident_2559') }}

union

select
    accident_date ,
    accident_time ,
    expw_step ,
    weather_state ,
    injur_man ,
    injur_femel ,
    dead_man ,
    dead_femel ,
    cause
from {{ ref('stg_accident_2560') }}

union

select
    accident_date ,
    accident_time ,
    expw_step ,
    weather_state ,
    injur_man ,
    injur_femel ,
    dead_man ,
    dead_femel ,
    cause
from {{ ref('stg_accident_2561') }}

```

26. แล้วทำการสร้าง model ที่มีชื่อว่า “events_accidents_count_total” โดยใช้ command ดังนี้


```

select
    accident_date ,
    accident_time ,
    expw_step ,
    weather_state ,
    count(*) AS events_accidents_total ,
    cause
from {{ ref('events_accidents_total') }}
group by accident_date , accident_time , expw_step , weather_state , cause

```

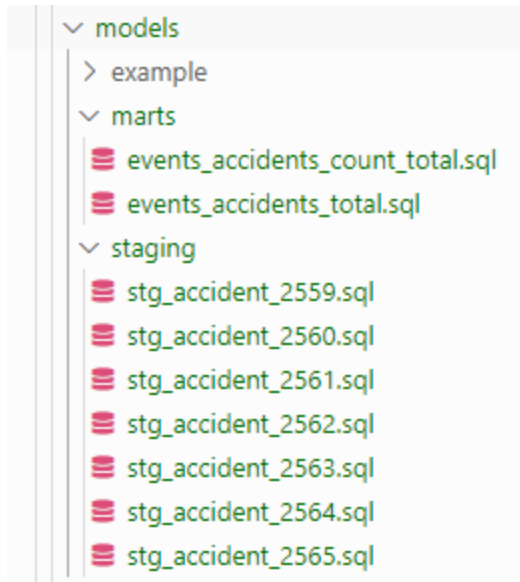
Write command: **dbt run**

```

• (ENV) gitpod /workspace/swu-ds525/08-capstone-project/accidents (main) $ dbt run
02:54:35 Running with dbt=1.3.1
02:54:35 Found 11 models, 4 tests, 0 snapshots, 0 analyses, 327 macros, 0 operations, 0 seed files, 0 sources, 0 exposures, 0 metrics
02:54:35
02:54:38 Concurrency: 1 threads (target='dev')
02:54:38
02:54:38 1 of 11 START sql table model public.my_first_dbt_model ..... [RUN]
02:54:40 1 of 11 OK created sql table model public.my_first_dbt_model ..... [SELECT in 2.71s]
02:54:40 2 of 11 START sql view model public.stg_accident_2559 ..... [RUN]
02:54:43 2 of 11 OK created sql view model public.stg_accident_2559 ..... [CREATE VIEW in 2.33s]
02:54:43 3 of 11 START sql view model public.stg_accident_2560 ..... [RUN]
02:54:45 3 of 11 OK created sql view model public.stg_accident_2560 ..... [CREATE VIEW in 2.29s]
02:54:45 4 of 11 START sql view model public.stg_accident_2561 ..... [RUN]
02:54:47 4 of 11 OK created sql view model public.stg_accident_2561 ..... [CREATE VIEW in 2.34s]
02:54:47 5 of 11 START sql view model public.stg_accident_2562 ..... [RUN]
02:54:50 5 of 11 OK created sql view model public.stg_accident_2562 ..... [CREATE VIEW in 2.45s]
02:54:50 6 of 11 START sql view model public.stg_accident_2563 ..... [RUN]
02:54:52 6 of 11 OK created sql view model public.stg_accident_2563 ..... [CREATE VIEW in 2.46s]
02:54:52 7 of 11 START sql view model public.stg_accident_2564 ..... [RUN]
02:54:55 7 of 11 OK created sql view model public.stg_accident_2564 ..... [CREATE VIEW in 2.36s]
02:54:55 8 of 11 START sql view model public.stg_accident_2565 ..... [RUN]
02:54:57 8 of 11 OK created sql view model public.stg_accident_2565 ..... [CREATE VIEW in 2.26s]
02:54:57 9 of 11 START sql view model public.my_second_dbt_model ..... [RUN]
02:54:59 9 of 11 OK created sql view model public.my_second_dbt_model ..... [CREATE VIEW in 1.79s]
02:54:59 10 of 11 START sql view model public.events_accidents_total ..... [RUN]
02:55:00 10 of 11 OK created sql view model public.events_accidents_total ..... [CREATE VIEW in 1.72s]
02:55:00 11 of 11 START sql view model public.events_accidents_count_total ..... [RUN]
02:55:02 11 of 11 OK created sql view model public.events_accidents_count_total ..... [CREATE VIEW in 1.71s]
02:55:03
02:55:03 Finished running 1 table model, 10 view models in 0 hours 0 minutes and 27.52 seconds (27.52s).
02:55:03
02:55:03 Completed successfully
02:55:03
02:55:03 Done. PASS=11 WARN=0 ERROR=0 SKIP=0 TOTAL=11

```

ซึ่งผลลัพธ์จะได้ทั้งหมด 7 Staging และ 2 Modeling



ภาพนี้เป็นภาพรวมของ **Staging** และ **Modeling** ทั้งหมดของเรา

27. เรามาลองทำการ Query ข้อมูล ที่ถูกเก็บอยู่ใน Redshift

Select database [Info](#)
To view schemas, select a database.

dev

Select schema [Info](#)
To view tables, select a schema.

public

Filter tables

1 select

2 *

3 from events_accidents_count_total

4

Run Save Schedule Clear

Send feedback

Query results Table details

Query 1417 [🔗](#)

Completed, started on December 18, 2022 at 09:55:59
ELAPSED TIME: 00 m 02 s

Rows returned (5582)

Export

Search rows

accident_date	accident_time	expw_step	weather_state	events_accidents_total	cause
2016-01-07	12:22:22	ศรัทธ	ปกติ	1	-
2014-11-02	14:40:40	ศรัทธ	ปกติ	1	ทำโทรศัพท์ชนแล้วกันเหย
2016-01-07	05:17:17	ฉลอสัย	ปกติ	1	หลบมือช้อนทาง
2016-01-06	22:59:59	ฉลอสัย	ปกติ	1	ชนยานที่สวนทางเดินรถ
2016-01-01	16:36:36	ฉลอสัย	ปกติ	1	ปกติ

28. ทำการสร้าง Document โดยการเขียน command ดังนี้

Write command: **dbt docs generate**

```

● (ENV) gitpod /workspace/swu-ds525/08-capstone-project/accidents (main) $ dbt docs generate
02:56:50 Running with dbt=1.3.1
02:56:51 Found 11 models, 4 tests, 0 snapshots, 0 analyses, 327 macros, 0 operations, 0 seed files, 0 sources, 0 exposures, 0 metrics
02:56:51
02:56:52 Concurrency: 1 threads (target='dev')
02:56:52
02:56:52 Done.
02:56:52 Building catalog
02:56:55 Catalog written to /workspace/swu-ds525/08-capstone-project/accidents/target/catalog.json

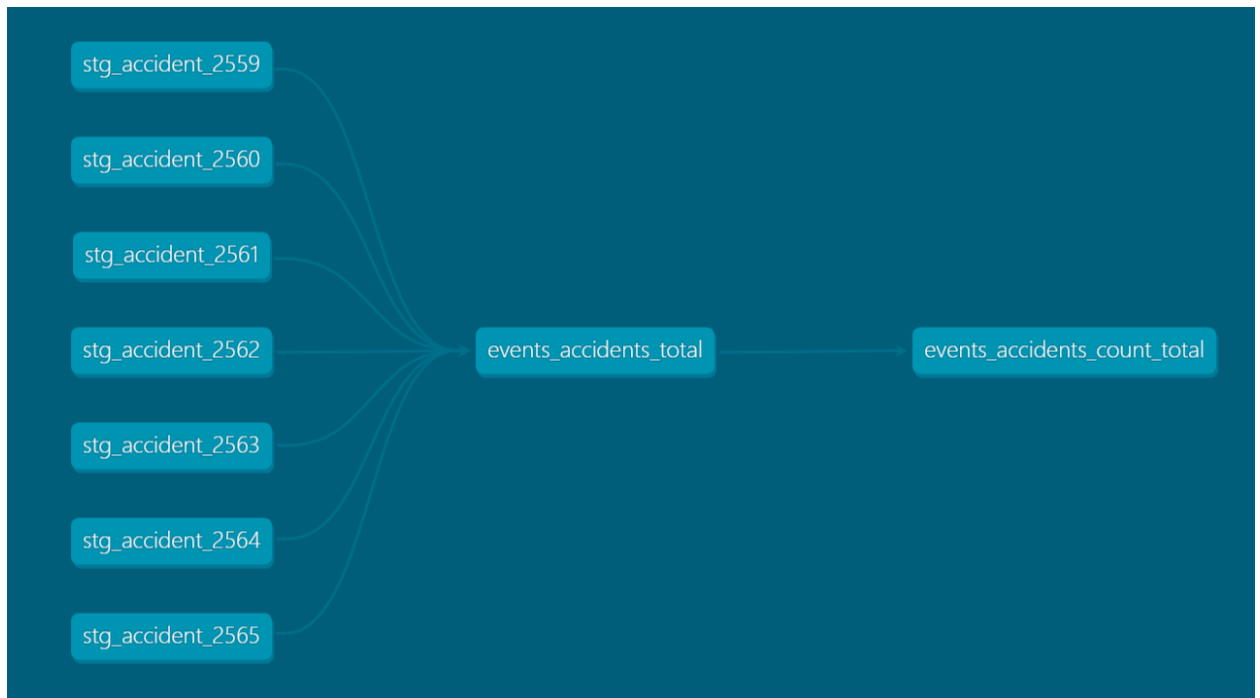
```

Write command: **dbt docs serve --port 8001**

```

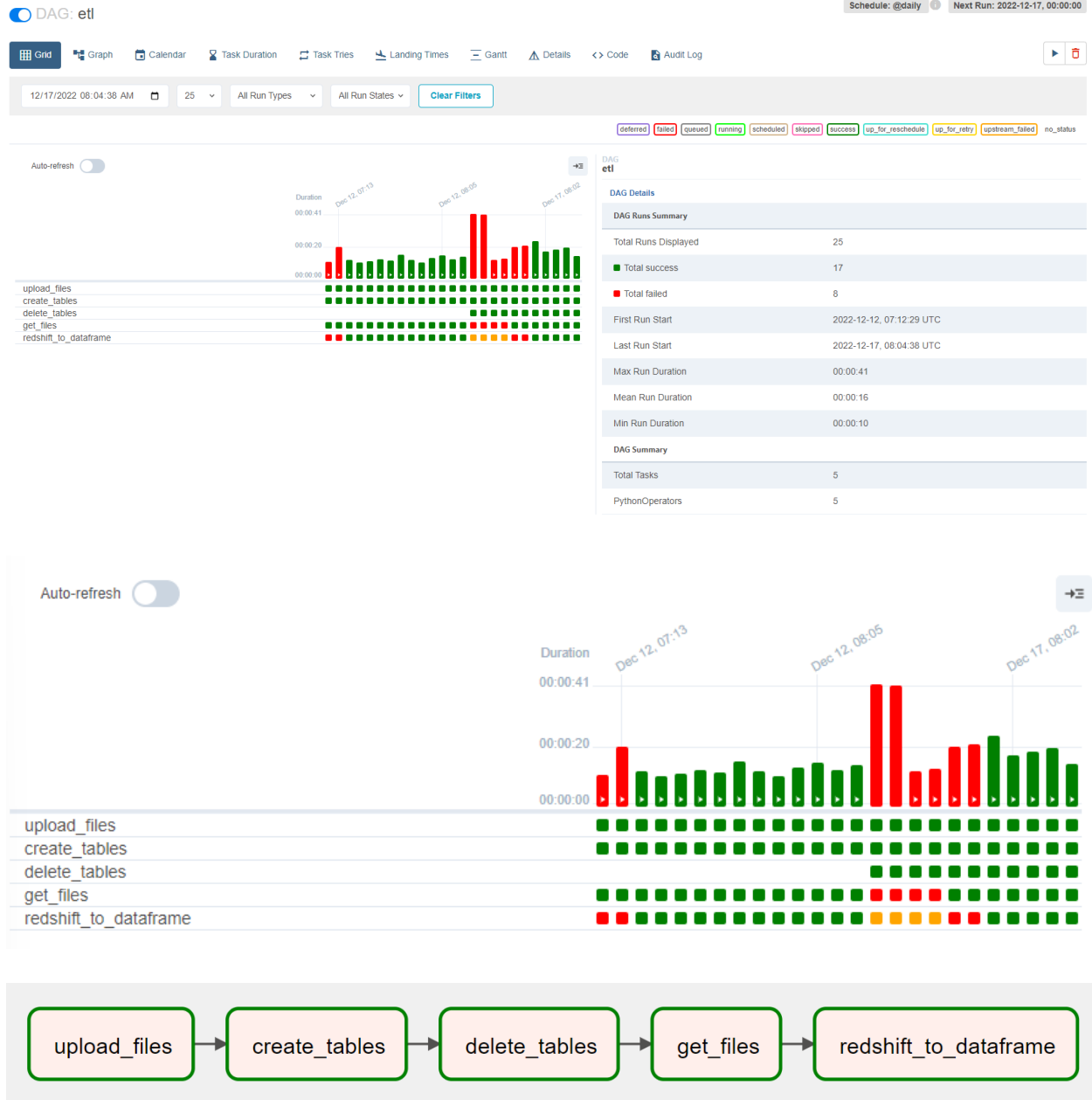
○ (ENV) gitpod /workspace/swu-ds525/08-capstone-project/accidents (main) $ dbt docs serve --port 8001
02:57:23 Running with dbt=1.3.1
02:57:23 Serving docs at 0.0.0.0:8001
02:57:23 To access from your browser, navigate to: http://localhost:8001
02:57:23
02:57:23 Press Ctrl+C to exit.
192.168.234.70 - - [18/Dec/2022 02:57:24] "GET / HTTP/1.1" 200 -
192.168.234.70 - - [18/Dec/2022 02:57:24] "GET /manifest.json?cb=1671332244764 HTTP/1.1" 200 -
192.168.234.70 - - [18/Dec/2022 02:57:24] "GET /catalog.json?cb=1671332244764 HTTP/1.1" 200 -

```



ซึ่งจากภาพ จะเห็นการทำงานของ **Staging** เหตุการณ์ของปีต่าง ๆ นำมาสร้าง **Modeling** ที่มีชื่อว่า “events_accidents_total” แล้วจาก **Modeling** ที่มีชื่อว่า “events_accidents_total” นำไปสรุปผลรวมของเหตุการณ์เป็น **Modeling** ที่มีชื่อว่า “events_accidents_count_total”

29. จากนั้นทำการ **Export Model** เหล่านี้ ออกมาเป็นไฟล์ **.csv**



30. จากนั้น เราจะนำไฟล์ที่ทำการ **Export Model** ออกมาเป็นไฟล์ **.csv** นำมาใช้ในการสร้าง **Visualization**

