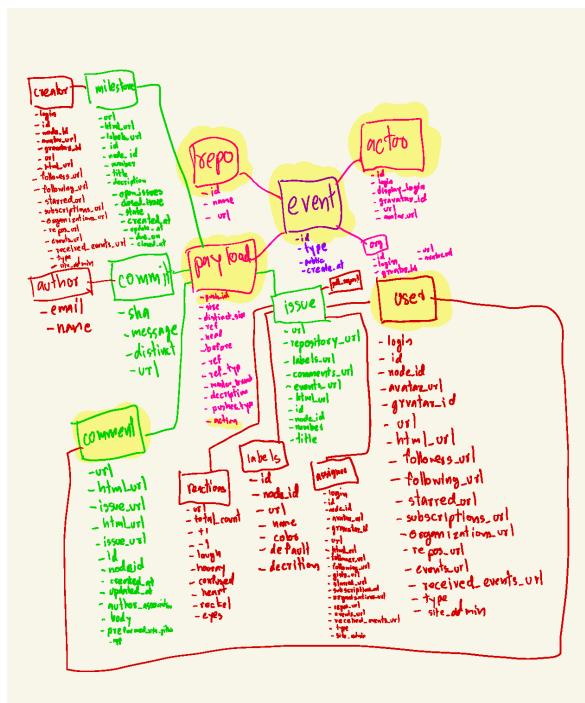


Building an ETL Pipeline for a Cloud Data Warehouse (Redshift/Google BigQuery)

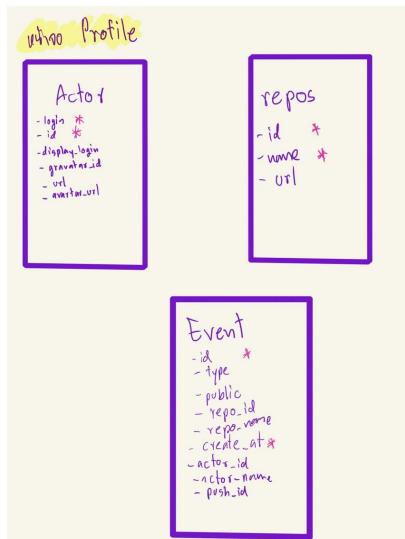
วิธีการดำเนินการ

1. วิเคราะห์โครงสร้างข้อมูลของ JSON ไฟล์ ได้ดังรูป



จากภาพ ข้อมูลใน Json เป็นข้อมูล เหตุการณ์ หรือ Event ของ Github โดยมีโครงสร้างที่ยืดหยุ่น โดยแต่ละ record จะมีข้อมูลที่ไม่เท่ากัน และไม่เหมือนกัน

2. ดำเนินการออกแบบตาราง “ได้ตารางดังนี้ (โดยทุกตารางไม่มีการเชื่อมโยงกัน และทุกตารางมีการกำหนด Partition Key เพื่อการจัดกลุ่มข้อมูล และ Clustering Key เพื่อการจัดเรียงข้อมูลที่อยู่ใกล้เคียงกัน รวมเป็น Primary Key เพื่อทำให้การ Query เข้าถึงได้อย่างรวดเร็ว)



โดย ตาราง event คือ ตารางที่เก็บข้อมูลการดำเนินการหรือ Activity ต่างๆ ของผู้ใช้งาน เช่น มีการ push สิ่งใด จำนวนกี่ครั้ง โดย Partition Key คือ type เนื่องจาก Type เป็นลักษณะของหมวดหมู่อยู่แล้ว และ Clustering Key คือ Create_at เป็นลักษณะของวันและเวลา หมายความว่าการนำมารอต์ข้อมูล ตาราง repo คือ ตารางที่เก็บข้อมูลในส่วนของพื้นที่ workspace ของแต่ละคน โดย Partition Key คือ id เนื่องจากตารางนี้เป็นตารางเก็บข้อมูลพื้นที่การทำงานของแต่ละ actor 1 actor สามารถมีได้หลาย repo 1 repo สามารถแชร์การทำงานร่วมกันได้ และ Clustering Key คือ name จากในตารางไม่มีข้อมูล Datetime จึงใช้ name ควบคู่ไปกับ id ตาราง actor คือ ตารางที่เก็บข้อมูลเจ้าของ Profile โดย Partition Key คือ id เนื่องจากตารางนี้เป็นตารางเก็บข้อมูลต่างๆ ของแต่ละ actor 1 และ Clustering Key คือ login จากในตารางไม่มีข้อมูลDatetime จึงใช้ Login ควบคู่ไปกับ id

- ทำการเตรียม Gitpod สำหรับการเขียนโปรแกรม โดยทำการ Sync Github account และ Gitpod เข้าด้วยกัน และทำการเตรียม environment

```

14
15     create_table_queries = [
16         """
17             CREATE TABLE IF NOT EXISTS staging_events (
18                 id varchar,
19                 type varchar,
20                 actor_id bigint,
21                 actor_login varchar,
22                 actor_url varchar,
23                 repo_id bigint,
24                 repo_name varchar,
25                 repo_url varchar,
    
```

JUPITER: VARIABLES

Name	Type
No variables defined	

TERMINAL

```

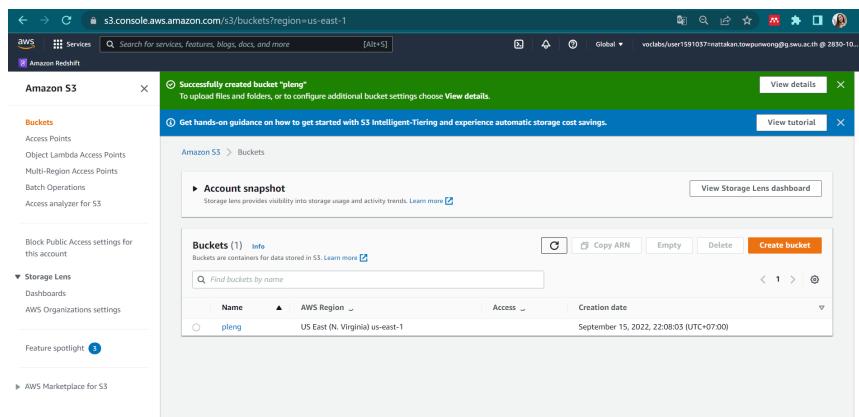
gitpod /workspace/swu-ds525/03-building-a-data-warehouse (main) $ python -m venv ENV
(ENV) gitpod /workspace/swu-ds525/03-building-a-data-warehouse (main) $ pip install -r requirements.txt
Requirement already satisfied: numpy==1.23.2 in ./ENV/lib/python3.8/site-packages (from -r requirements.txt (line 1)) (1.23.2)
Requirement already satisfied: psycopg2==2.9.3 in ./ENV/lib/python3.8/site-packages (from -r requirements.txt (line 2)) (2.9.3)
Requirement already satisfied: python-dateutil==2.8.2 in ./ENV/lib/python3.8/site-packages (from -r requirements.txt (line 3)) (2.8.2)
Requirement already satisfied: pytz==2022.2.1 in ./ENV/lib/python3.8/site-packages (from -r requirements.txt (line 4)) (2022.2.1)
Requirement already satisfied: six==1.16.0 in ./ENV/lib/python3.8/site-packages (from -r requirements.txt (line 5)) (1.16.0)
WARNING: You are using pip version 22.0.4; however, version 22.2.1 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
(ENV) gitpod /workspace/swu-ds525/03-building-a-data-warehouse (main) $
    
```

4. ทำการแก้ไขข้อมูลในไฟล์ events_json_path.json ให้ตรงกับที่ออกแบบ

```

1 { "jsonpaths": [
2     "$.id",
3     "$.type",
4     "$.actor.id",
5     "$.actor.login",
6     "$.actor.url",
7     "$.repo.id",
8     "$.repo.name",
9     "$.repo.url",
10    "$.public",
11    "$.created_at",
12    "$.actor.display_login",
13    "$.actor.gravatar_id",
14    "$.actor.url",
15    "$.payload.push_id",
16 ]
    }
    
```

5. เข้าไปที่ Amazon S3 สร้าง Bucket สำหรับเก็บไฟล์ต่างๆ พร้อมทั้ง Upload ไฟล์ events_json_path.json และ github_events_01.json



The screenshot shows the AWS S3 console interface. At the top, it says "Upload succeeded". Below that, there's a summary table with columns for Destination, Status, and Failed. Under "Destination" is "s3://pleng". Under "Status" are "Succeeded" and "Failed". Under "Failed" is "0 files, 0 B (0%)". Below this is a "Files and folders" section showing two items: "events_json_path.json" and "github_events_01.json", both of which have a status of "Succeeded".

5. เข้าไปที่ Amazon Redshift เพื่อสร้าง Cluster รายละเอียดดังรูป

The screenshots show the "Create cluster" wizard in the AWS Redshift console.

- Cluster configuration:** Shows the "Cluster identifier" as "redshift-cluster-1", "Node type" as "ra3.xlarge", "Number of nodes" as "1", and a "Free trial" option selected.
- Admin user setup:** Shows the "Admin user name" as "password" and "Admin user password" as "Plano65720900".
- Cluster permissions:** Shows the "Associated IAM roles" section where "Create an IAM role as the default for this cluster that has the AmazonRedshiftAllCommandsFullAccess policy attached" is selected.
- Additional configurations:** Shows network settings (using default VPC), backup settings (automated snapshots every 7 hours), and maintenance settings (using current maintenance track). A "Create cluster" button is at the bottom.

6. เมื่อได้สถานะ Available ให้ทำการกำหนดให้ Cluster สามารถเข้าถึงได้ โดย Enable publicly accessible

The first screenshot shows the 'General information' tab for 'redshift-cluster-1'. The cluster identifier is 'redshift-cluster-1', status is 'Available', node type is 'ra3.xlplus', number of nodes is 1, endpoint is 'redshift-cluster-1.cj0baeqdep.us-east-1.redshift.amazonaws.com', JDBC URL is 'jdbc:redshift://redshift-cluster-1.cj0baeqdep...', ODBC URL is 'Driver=(Amazon Redshift (x64)); Server=redshift-cluster-1.cj0baeqdep.us-east-1.redshift.amazonaws.com; Port=5439; Database=dev; User=nattakan.towpunwong; Password='.

The second screenshot shows the same cluster page with the 'Actions' dropdown open. The 'Modify publicly accessible setting' option is highlighted.

The third screenshot shows the 'Edit publicly accessible' dialog box. The 'Enable' radio button is selected. A warning message states: '⚠ Your cluster might be unavailable for up to 10 minutes while this change to public accessibility is processed.' There are 'Cancel' and 'Save changes' buttons at the bottom.

7. ทำการสร้างการเชื่อมต่อกับ Database

The screenshot shows the 'Connect to database' dialog box. Under 'Connection', 'Create a new connection' is selected. Under 'Authentication', 'Temporary credentials' is selected. In the 'Database' section, 'Database name' is set to 'dev' and 'Database user' is set to 'awesusef'. At the bottom right are 'Cancel' and 'Connect' buttons.

8. ทำการ Copy S3 ไฟล์ทั้ง 2 ไฟล์ และ ARN ของ IAM ไปวางที่ไฟล์ etl.py

The first screenshot shows the AWS S3 console with the file 'github_events_01.json' selected. A 'Copy S3 URI' button is highlighted, indicating the action to copy the file's URL.

The second screenshot shows the AWS IAM Roles page for 'LabRole'. An 'ARN Copied' message is visible, and the ARN of the role is displayed in the summary section.

The third screenshot shows a code editor for 'etl.py' containing the following code:

```

swuds525-ruzbqf73rkj.ws-us65.gitpod.io
 README.md requirements.txt etl.py M events_json_path.json gitl
 3-building-a-data-warehouse > etl.py > [e] insert_table_queries

21     '',
22 ]
23 copy_table_queries = [
24     '',
25     COPY staging_events FROM 's3://pleng/github_events_01.json'
26     CREDENTIALS 'aws_iam_role=arn:aws:iam::283010062450:role/LabRole'
27     JSON 's3://pleng/events_json_path.json'
28     REGION 'us-east-1'
29     '',
30
31 insert_table_queries = []
32     '',
33     INSERT INTO
34         events (

```

9. ทำการแก้ไขรายละเอียดส่วน Function Main ของไฟล์ etl.py เพื่อเชื่อมโยงกับ Amazon Redshift

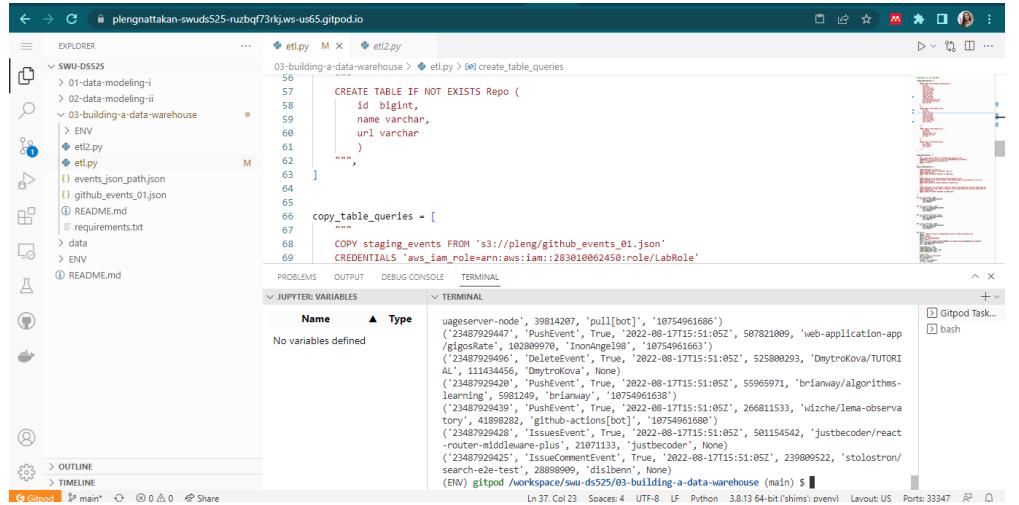
The code editor shows the 'etl.py' file with the following modified 'main' function:

```

1 READMe.md 2 requirements.txt 3 etl.py M 4 events_json_path.json 5 github_events_01.json
 3-building-a-data-warehouse > etl.py > ...
119
120
121 def main():
122     host = "redshift-cluster-1.ci0boaeqvdep.us-east-1.redshift.amazonaws.com"
123     dbname = "dev"
124     user = "awsuser"
125     password = "Pleng056720990"
126     port = "5439"
127     conn_str = f"host={host} dbname={dbname} user={user} password={password} port={port}"
128     conn = psycopg2.connect(conn_str)
129     cur = conn.cursor()
130
131     drop_tables(cur, conn)

```

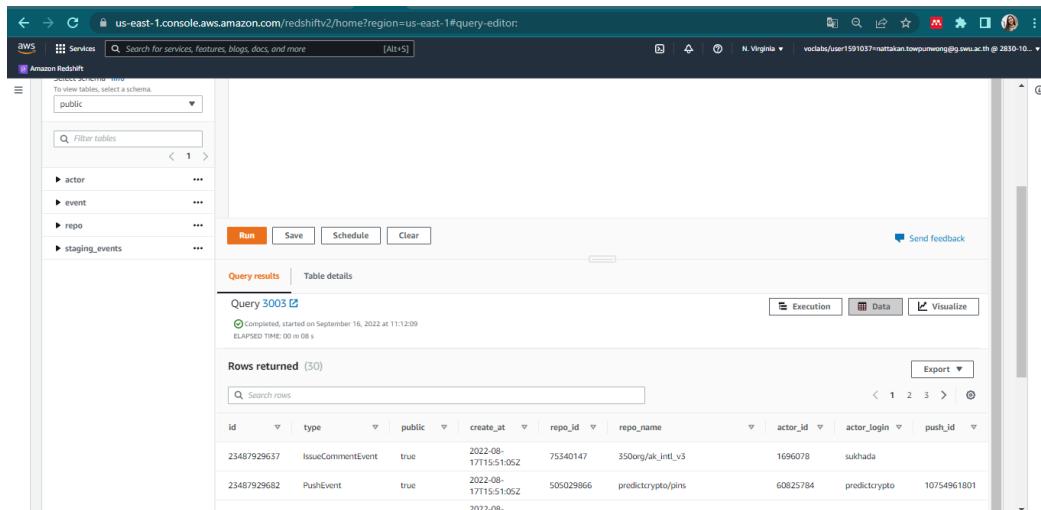
10. ทำการเขียนโปรแกรมเพื่อสร้างตาราง Insert ข้อมูล และ Query ข้อมูลจากไฟล์ JSON แล้ว Run โดยไปตรวจสอบที่หน้าจอ Amazon



```

CREATE TABLE IF NOT EXISTS Repo (
    id bigint,
    name varchar,
    url varchar
)
;

copy_table_queries = [
    """
    COPY staging_events FROM 's3://pleng/github_events_01.json'
    CREDENTIALS 'aws_iam_role=arn:aws:iam::283010062450:role/LabRole'
    """
]
    
```



Query results

id	type	public	create_at	repo_id	repo_name	actor_id	actor_login	push_id
23487929637	IssueCommentEvent	true	2022-08-17T15:51:05Z	75340147	3500org/ak_intl_v\$	1696078	sukhada	
23487929682	PushEvent	true	2022-08-17T15:51:05Z	505029866	predictcrypto/pins	60825784	predictcrypto	10754961801