

## **ปัญหาและที่มาของงาน**

PM2.5 หรือชื่อเต็มคือ Particulate Matter with diameter of less than 2.5 micron เป็นฝุ่นละอองขนาดจิ๋วที่มีขนาดไม่เกิน 2.5 ไมครอน ซึ่งมีขนาดประมาณ 1 ใน 25 ส่วนของเส้นผ่าศูนย์กลางเส้นผ่านศูนย์กลาง เดียวกันจะมีขนาดของฝุ่น PM2.5 ที่ทำให้เกิดจากฝุ่น PM2.5 สามารถแพร่กระจายเข้าสู่ทางเดินหายใจ กระแสเลือด และเข้าสู่อวัยวะอื่นๆ ในร่างกายได้

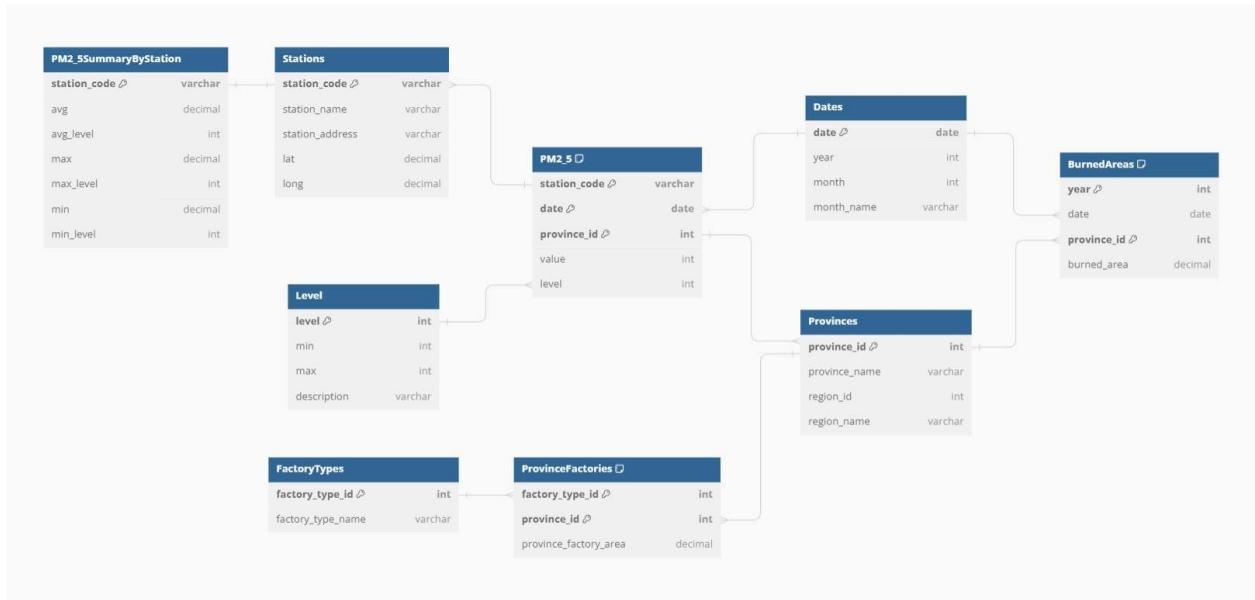
PM2.5 ได้เริ่มนีบทบาทในประเทศไทยอย่างมากในช่วงต้นปี 2562 ในตอนนั้นประเทศไทยประสบกับสภาพอากาศที่มีฝุ่นปกคลุมอย่างหนาแน่น ทำให้ทุกภาคส่วนทั้งรัฐบาล เอกชน ตลอดจนประชาชนล้วนให้ความสนใจเป็นอย่างมากพร้อมกับหัวคิดว่าเกิดอะไรขึ้น จนสุดท้ายก็ทำให้เราทุกคนได้รู้จักกับค่าฝุ่น PM2.5 นี้ นับตั้งแต่นั้นมาจึงมีการตรวจสอบปริมาณค่าฝุ่น PM2.5 ตามจุดบริเวณต่างๆ ของประเทศไทย เพื่อตรวจสอบว่าเป็นอันตรายต่อสุขภาพหรือไม่

PM2.5 สามารถเกิดขึ้นได้จากหลายปัจจัย ทั้งจากการผลิตในโรงงานอุตสาหกรรม จากการเผาไหม้ในกิจกรรมในครัวเรือน จากการเผาทางการเกษตร จากการเผาป่า จากการการจราจร รวมถึงสาเหตุอื่น เช่น การรวมตัวของก้าชอื่นๆ ในบรรยายกาศ

ด้วยเหตุนี้ทำให้ทางกลุ่มของเรามีความสนใจที่จะศึกษาเกี่ยวกับแนวโน้มของปริมาณค่าฝุ่น PM2.5 ในประเทศไทยตามบริเวณต่างๆ รวมถึงต้องการศึกษาว่า ปัจจัยใดที่จะมีแนวโน้มในการส่งผลกระทบต่อปริมาณค่าฝุ่น PM2.5 มากกว่ากัน โดยในปัจจุบันนี้ขอทำการศึกษาทั้งหมด 2 ปัจจัย ได้แก่ 1. โรงงานอุตสาหกรรม: พิจารณาจากพื้นที่ของโรงงานอุตสาหกรรมในแต่ละภูมิภาค และ 2. ไฟป่า: พิจารณาพื้นที่ที่เกิดไฟป่าในภูมิภาคนั้นๆ

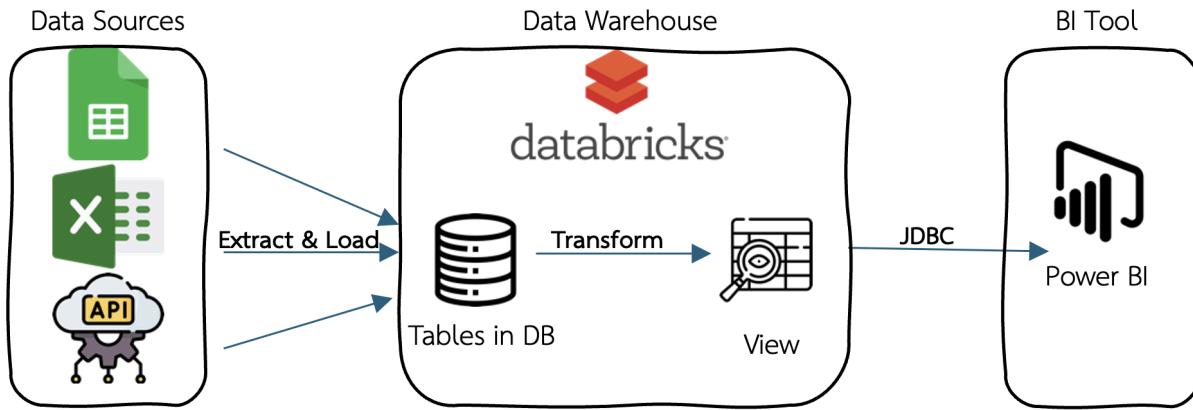
โดยการใช้งานจริงนั้นคาดว่าสามารถใช้แนวโน้มค่าฝุ่นที่เปลี่ยนไปเป็นตัวชี้วัดคุณภาพการแก้ปัญหาไฟป่าและมาตรการจัดการฝุ่นละอองที่เกิดจากโรงงานอุตสาหกรรมในแต่ละภูมิภาคหรือระดับจังหวัด

# Data Model



# ขั้นตอนในการทำงาน

## ภาพรวมของ workflow



### 1. Data Sources

#### 1.1 ข้อมูลที่ไม่มีการอัพเดท หรือมีการอัพเดทด้านๆ ที่ (Static data)

- ข้อมูล PM2.5 ตั้งแต่ปี 2020 ถึงปี 2024 รายวันจากแต่ละสถานีวัดคุณภาพอากาศที่ ถูกบันทึกไว้ในเว็บไซต์ในรูปแบบตาราง excel ดังตัวอย่าง

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	Date	02T	05T	10T	11T	12T	59T	61T	03T	50T	52T	53T	54T	08T	13T	14T
2	1/1/2020	20	22	18	23	17	20	20	21	22	21	21	23	29	20	32
3	2/1/2020	26	24	20	26	21	24	26	28	27	27	28	30	32	25	28
4	3/1/2020	32	28	24	31	24	27	32	32	29	33	30	36	38	30	33
5	4/1/2020	37	35	31	35	35	34	37	38	39	39	35	44	48	35	43
6	5/1/2020	45	41	33	40	36	38	42	51	44	45	40	46	56	39	53
7	6/1/2020	51	49	39	46	49	40	46	62	50	52	45	52	63	45	66
8	7/1/2020	51	57	39	47	45	38	57	64	52	53	57	64	66	48	71
9	8/1/2020	53	64	47	53	52	49	60	58	55	55	59	63	76	49	59
10	9/1/2020	60	55	47	49	49	47	53	74	56	63	56	70	67	60	81
11	10/1/2020	97	100	63	72	78	70	78	96	82	101	77	102	112	82	94
12	11/1/2020	46	46	35	46	35	42	47	57	49	49	46	65	55	47	61
13	12/1/2020	36	37	26	35	33	33	38	38	39	39	38	55	47	42	40
14	13/1/2020	33	33	26	36	33	30	34	39	42	39	38	45	47	40	41
15	14/1/2020	34	36	27	35	30	29	35	42	37	34	38	47	44	37	44
16	15/1/2020	35	38	30	37	31	31	39	45	39	36	38	49	45	40	45
17	16/1/2020	37	41	33	39	37	37	40	47	42	38	42	56	47	40	46
18	17/1/2020	45	46	36	47	43	42	47	58	50	45	48	60	54	51	56
19	18/1/2020	58	58	48	55	52	52	55	59	59	54	56	79	64	61	63
20	19/1/2020	56	58	50	56	53	52	57	58	58	55	56	74	65	59	57

ไฟล์ .xlsx นี้ถูกอัปโหลดขึ้น databricks file system (DBFS) โดยตรง ผ่าน UI ของ databricks เนื่องจากข้อมูลส่วนนี้มีความน่าจะเป็นต่าที่จะเกิดการแก้ไขในอนาคต

- ข้อมูลเกี่ยวกับไฟป่าและโรงงาน ถูกบันทึกไว้ใน google sheet เนื่องจากข้อมูลส่วนนี้มักมีการอัพเดตรายปีในรูปแบบ .xlsx จากหน่วยงานรัฐ เมื่อต้องการจะอัพเดทก็สามารถคัดลอกมาใส่โดยง่าย อีกทั้งเพื่อต้องการแก้ไขข้อ้งงาน หรือแก้ไข typo ต่างๆ ก็สามารถแก้ใน google sheet ได้ ดังตัวอย่างด้านล่าง

	A	B	C	D	E	F	G	H	I
1	province_id	factory_type_id	province_factory_area						
2	50	1	6,400.00						
3	50	1	9,600.00						
4	50	1	32,000.00						
5	50	1	81,416.00						
6	51	1	13,200.00						
7	52	1	24,000.00						
8	52	1	12,000.00						
9	52	1	1,810.00						
10	54	1	54,504.00						
11	54	1	52,960.00						
12	54	1	13,884.00						
13	54	1	67,200.00						
14	55	1	3,200.00						
15	55	1	12,800.00						
16	55	1	43,200.00						

	A	B	C	D	E	F
1	station_code	station_name	station_address	lat	long	province_id
2	02t	Bansomdejchaopraya Rajabhat University	Hiran Ruchi Khet Thor Buri, Bangkok	13.732846	100.487662	10
3	03t	Highway NO.3902 km.13 +600	Kanchanaphisek Rd, Bang Khun Thian, Bangkok	13.636514	100.414262	10
4	05t	Thai Meteorological Department	Bang Na, Khet Bang Na, Bangkok	13.666183	100.605742	10
5	08t	Prabangdang Rehabilitation Center	Song Kanong, Phra Pradaeng, Samut Prakan	13.664087	100.543436	11
6	11t	National Housing Huaykwang	Din Daeng, Khet Din Daeng, Bangkok	13.77553	100.569195	10
7	12t	Nonsi Witthaya School	Chong Nonsi, Khet Yannawa, Bangkok	13.70806667	100.5473333	10
8	13t	Department of Disease Control	Talad Kwan, Muang, Nonthaburi	13.852778	100.529444	12
9	14t	Highway District	Om Noi, Krathum Baen, Samut Sakhon	13.7054879	100.315622	74
10	16t	South Bangkok Power Plant	Bang Prong, Meuang, Samut Prakan	13.618	100.5562	11
11	17t	Residence for Dept. of Mineral Resources	Talat, Phra Pradaeng, Samut Prakan	13.652141	100.531805	11
12	18t	City Hall, Samut Prakan	Pak Nam, Meuang, Samut Prakan	13.599149	100.597345	11
13	19t	National Housing Authority Bangphee	Bang Sao Thong, Bang Sao Thong, Samut Prakan	13.5705	100.7862867	11
14	20t	Bangkok University Rangsit Campus	Khlong Nueng, Khlong Luang, Pathum Thani	14.03742	100.605216	13
15	21t	Ayutthaya Witthayalai School	Pratu Chai, Phra Nakhon Si Ayutthaya	14.35222	100.565325	14
16	22t	Sukhothai Thammathirat Open University	Bang Phut, Pak Kret, Nonthaburi	13.907861	100.535641	12
17	24t	Na Phralan Police Station	Na Phra Lan, Chaloem Phra Kiat, Saraburi	14.685833	100.871996	19
18	25t	Khao Nai Fire Station	Pak Phria, Mueang Saraburi, Saraburi	14.526343	100.926047	19

ข้อมูลเหล่านี้จะถูกบันทึกลง DBFS โดยการ create python code (ในที่นี้ตั้งชื่อว่า `data-api-upsert.py`) ใน workspace และ run ผ่าน pipeline ซึ่ง

## จะกล่าวถึงต่อไป

The screenshot shows the Microsoft Azure Databricks workspace interface. On the left, there's a sidebar with navigation links like 'New', 'Workspace', 'Recent', 'Catalog', 'Workflows', 'Compute', 'Data Engineering', 'Job Runs', 'Machine Learning', 'Playground', 'Experiments', 'Features', 'Models', and 'Serving'. The main area displays a table of workspace items:

Name	Type	Owner	Created at
create_view_to.bi	Notebook	Jirasak Sai-mek	2024-05-07 16:35:30
create_view_to_bu.py	File	Jirasak Sai-mek	2024-05-07 16:18:17
credentials.json	File	Jirasak Sai-mek	2024-05-05 10:26:51
data-spn-upsert.py	File	Jirasak Sai-mek	2024-04-24 22:48:12
get_spn-source	Notebook	Jirasak Sai-mek	2024-04-20 17:05:50
pme_2024-05-07 13:43:37	Dashboard	Jirasak Sai-mek	2024-05-07 13:43:39
token.json	File	Jirasak Sai-mek	2024-05-05 10:27:04

```
449
130 # Fetch Google Sheets data and convert to PySpark DataFrames
131 data_processed = []
132 creds = None
133
134 # Check if token file exists and credentials are valid
135 SCOPES = ["https://www.googleapis.com/auth/spreadsheets.readonly"]
136 SAMPLE_SPREADSHEET_ID = "19aFrlAgtPn6Uv1CjUiqTUhbvZSRnZUBZ_hi8jG8g2c" [1]
137
138 # columns to change
139 long_cols = ['year', 'province_id', 'factory_type_id', 'region_id', 'Level', 'Min', 'Max']
140 str_cols = ['factory_type_name', 'province_factory_area', 'province_name', 'region_name', 'station_code', 'station_name', 'station_address']
141 date_cols = ['date', 'string', 'date']
142 double_cols = ['value', 'lat', 'long']
143
144 for station_id in data:
145     processed_data = process_station_data(data[station_id])
146     data_processed.append(processed_data)
147
148 raw_columns = ["station_code", "station_name", "lat", "long", "province_id", "date", "value", "station_address"]
149 raw_data = spark.createDataFrame(data_processed, raw_columns)
150 pm_25 = raw_data.select("date", "station_code", "value")
151
152 # Fetch Google Sheets data and convert to PySpark DataFrames
153
154 # Check if token file exists and credentials are valid
155 if os.path.exists("/Workspace/Users/jirasak.sai-mek/.databricks/python/lib/token.json"): [2]
156     creds = Credentials.from_authorized_user_file("/Workspace/Users/jirasak.sai-mek/.databricks/python/lib/token.json", SCOPES)
157 if not creds or not creds.valid:
158     if creds and creds.expired and creds.refresh_token:
159         creds.refresh(Request())
160     else:
161         flow = InstalledAppFlow.from_client_secrets_file("credentials.json", SCOPES)
162         creds = flow.run_local_server(port=0)
163
164 with open("/Workspace/Users/jirasak.sai-mek/.databricks/python/lib/token.json", "w") as token:
165     token.write(creds.to_json())
166
167 # Build Google Sheets service
168 service = build("sheets", "v4", credentials=creds)
169
170 # Read data from Google Sheets into Pandas DataFrames
171 burned_areas_df = read_google_sheets(service, SAMPLE_SPREADSHEET_ID, "burned_areas!A:C")
172 region_df = read_google_sheets(service, SAMPLE_SPREADSHEET_ID, "region!A:B")
173 province_factories_df = read_google_sheets(service, SAMPLE_SPREADSHEET_ID, "province_factories!A:C")
```

ในส่วนของ (1) จะต้องใส่ sheet ID ขึ้งๆได้จาก URL ของ google sheet (highlighted) ดังภาพ

station_code	station_name	station_address	lat	long	province_id
1					

และ (2) เป็น credential ที่ generate จาก GCP

ผลลัพธ์ของตารางที่ดึงข้อมูลสำเร็จสามารถดูจาก databricks UI ได้ดังตัวอย่างนี้

	id	year	province_id	burned_area
1	2553	82	4000	
2	2553	62	4800	
3	2553	50	3200	
4	2553	57	4800	
5	2553	63	400	
6	2553	60	4800	
7	2553	67	3200	
8	2553	54	4800	
9	2553	56	4800	
10	2553	69	3200	
11	2553	58	480	
12	2553	52	3200	
13	2553	51	1600	
14	2553	64	948800	
15	2553	53	1789440	
16	2553	61	1579008	
17	2553	46	3030400	
18	2553	49	640000	
19	2553	39	827200	
20	2553	48	1444800	
21	2553	30	599600	
22	2553	31	857600	
23	2553	44	4468000	
24	2553	49	3345600	
25	2553	35	2630800	
76	2553	45	4079600	

1.2 ข้อมูลที่มีการอัปเดททุกวัน ได้แก่ข้อมูลปริมาณ PM2.5 จากแต่ละสถานีในแต่ละวัน ซึ่งจะถูกดึงมาจาก API เว็บไซต์ air4thai.com โดยใช้โค้ดจาก notebook data-api-upsert.py เช่นกัน ตัวอย่างดังภาพ

```

# Function to process station data fetched from API
def process_station_data(data):
    stationid = data['stationID']
    name_eng = data['nameEN']
    loc_lat = data['lat']
    loc_long = data['long']
    area_eng = data['areaEN']
    parts = area_eng.split(' ', ' ')
    # Split area_eng to extract station and province information
    if len(parts) >= 3:
        station = parts[0] + ', ' + parts[1]
        province = parts[2].rstrip(' ')
    else:
        province = None

    prov_id = None
    # Lookup province_id using Spark DataFrame
    if province:
        try:
            prov_id = prov_sql.filter(prov_sql['province_name'] == province).collect()[0]['province_id']
        except IndexError:
            pass

date_ori = data['AQILast']['date']
aqi = data['AQILast']['PM25']['value']

return stationid, name_eng, loc_lat, loc_long, prov_id, date_ori, aqi, area_eng

# Configuration and setup
base_url = "http://air4thai.com/forweb/getAQI_JSON.php?stationID="
headers = {
    "Accept": "application/json",
    "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.127 Safari/537.36"
}
# Spark SQL DataFrame for province information
prov_sql = spark.sql("SELECT province_name, province_id FROM default.provinces")
# Fetch data for all stations
data = {}
station_ids = ['02t', '03t', '05t', '08t', '11t', '12t', '13t', '14t', '16t', '17t', '18t', '19t', '20t', '21t', '22t', '24t', '25t',
               '26t', '27t', '28t', '29t', '31t', '33t', '34t', '35t', '36t', '37t', '38t', '39t', '40t', '41t', '42t', '43t',
               '44t', '46t', '47t', '50t', '52t', '53t', '54t', '57t', '58t', '59t', '60t', '61t', '62t', '63t', '67t', '68t', '69t', '70t',
               '71t', '72t', '73t', '74t', '75t', '76t', '77t', '78t', '79t', '80t', '81t']
for station_id in station_ids:
    try:
        station_data = fetch_station_data(base_url, station_id, headers)
        data[station_id] = station_data
    except Exception as e:
        print(f"Error fetching data for station {station_id}: {e}")
# Collect all results outside the loop
station_results = []
for station_id in data:
    result = find_region_info(data[station_id])
    if result:
        station_results.append(result)
# Create DataFrame once outside the loop
df_stations = spark.createDataFrame(station_results, schema=["station_code", "station_name", "station_address", "lat", "long", "province_id"])
# Fetch Google Sheets data and convert to PySpark DataFrames
data_processed = []
creds = None

```

```
{
  "stationID": "02t",
  "nameTH": "มหาวิทยาลัยราชภัฏบ้านสมเด็จเจ้าพระยา",
  "nameEN": "Bansomdejchaopraya Rajabhat University",
  "areaTH": "แขวงทิรัญธรวิชัย เขตธนบุรี, กรุงเทพฯ",
  "areaEN": "Hiran Ruchi, Khet Thon Buri, Bangkok",
  "stationType": "GROUND",
  "lat": "13.732846",
  "long": "100.487662",
  "forecast": [],
  "AQILast": {
    "date": "2024-05-09",
    "time": "21:00",
    "PM25": {
      "color_id": "2",
      "aqi": "34",
      "value": "18.4"
    },
    "PM10": {
      "color_id": "0",
      "aqi": "-1",
      "value": "-1"
    },
    "O3": {
      "color_id": "0",
      "aqi": "-1",
      "value": "-1"
    },
    "CO": {
      "color_id": "0",
      "aqi": "-1",
      "value": "-1"
    },
    "NO2": {
      "color_id": "0",
      "aqi": "-1",
      "value": "-1"
    },
    "SO2": {
      "color_id": "0",
      "aqi": "-1",
      "value": "-1"
    },
    "AQI": {
      "color_id": "2",
      "aqi": "34",
      "param": "PM25"
    }
  }
}
```

## 2. Orchestration

Task หลักในงานถูกรวมอยู่ใน 1 notebook ชื่อ **data-api-upsert.py** ซึ่งมีทั้งการสร้างตาราง การโหลดข้อมูลลงตาราง และการสร้าง view อญญาณใน notebook เดียวกัน

### 2.1 ตัวอย่างการ create table (if not exists) & insert data

สร้างตารางต่างๆ

```

220  #create table first
221  try:
222  |   region_spark_df.write.format("delta").saveAsTable("default.regions")
223  except:
224  |   pass
225  try:
226  |   factory_types_spark_df.write.format("delta").saveAsTable("default.factorytypes")
227  except:
228  |   pass
229  try:
230  |   provinces_spark_df.write.format("delta").saveAsTable("default.provinces")
231  except:
232  |   pass
233  try:
234  |   df_stations.write.format("delta").saveAsTable("default.stations")
235  except:
236  |   pass
237  try:
238  |   levels_spark_df.write.format("delta").saveAsTable("default.levels")
239  except:
240  |   pass
241

```

## Append

```
# Write data to Delta tables
# Append PM2.5 data to existing 'pm2_5' Delta table
pm_25.write.format("delta").mode("append").saveAsTable("default.pm2_5")
```

```
# Upsert data into 'regions', 'factorytypes', 'provinces', and 'stations' Delta tables
# Upsert 'regions' table
region_spark_df.createOrReplaceTempView('region')
Update_region_province = f"""
    MERGE INTO default.regions AS target
    USING region AS source
    ON source.region_id = target.region_id
    WHEN MATCHED THEN UPDATE SET *
    WHEN NOT MATCHED THEN INSERT *
    .....
spark.sql(Update_region_province)

# Upsert 'factorytypes' table
factory_types_spark_df.createOrReplaceTempView('factory_types')
Update_region_province = f"""
    MERGE INTO default.factorytypes AS target
    USING factory_types AS source
    ON source.factory_type_id = target.factory_type_id
    WHEN MATCHED THEN UPDATE SET *
    WHEN NOT MATCHED THEN INSERT *
    .....
spark.sql(Update_region_province)
```

## 2.2 Create view

ในตัวอย่างจะเป็นการเขียนคำสั่ง sql และ run ด้วย spark.sql เพื่อ create view สำหรับการนำไปใช้งานบน BI tools ต่อไป

```

# code create view v_pm2_5
create_view_v_pm2_5 = """Create view v_pm2_5 as
SELECT p.station_code, s.province_id, p.date, p.value, l.Level as level
FROM default.pm2_5 p
LEFT JOIN default.levels l ON p.value BETWEEN l.Min AND l.Max
LEFT JOIN default.stations s ON p.station_code = s.station_code
"""

# code create view v_burnedareas01
create_view_v_burnedareas = """Create view v_burnedareas01 as
SELECT *, CONCAT(CASE
    WHEN year > 2500 THEN year - 543
    ELSE year
END, '-1-1') AS date_string
FROM default.burnedareas
"""

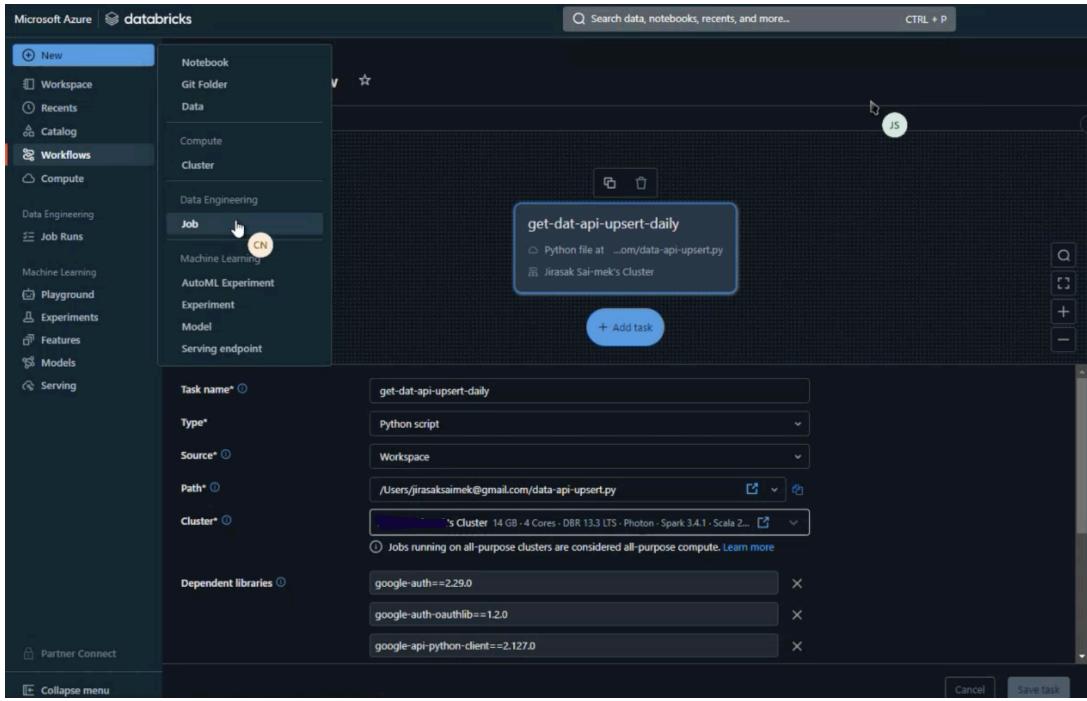
# create view in first
try:
    spark.sql(create_view_v_pm2_5bystation)
except:
    pass
try:
    spark.sql(create_view_v_date)
except:
    pass
try:
    spark.sql(create_view_v_pm2_5)
except:
    pass
try:
    spark.sql(create_view_v_burnedareas)
except:
    pass

```

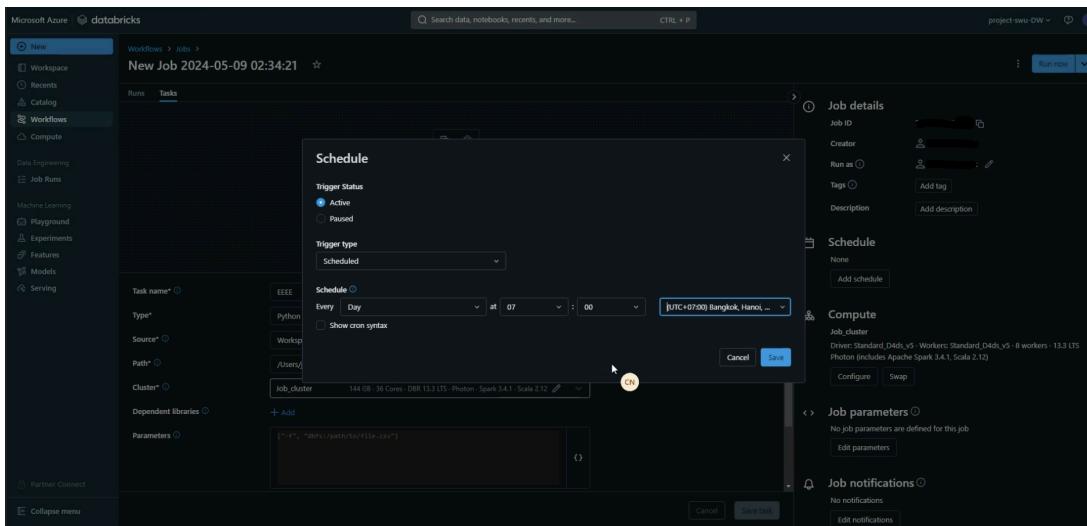
### 2.3 การสร้าง workflow และ job scheduling

บน databricks ให้ import ไฟล์ **data-api-upsert.py** ไปวางใน Workspace หลังจากนั้น ไปที่ New -> Job -> Add task

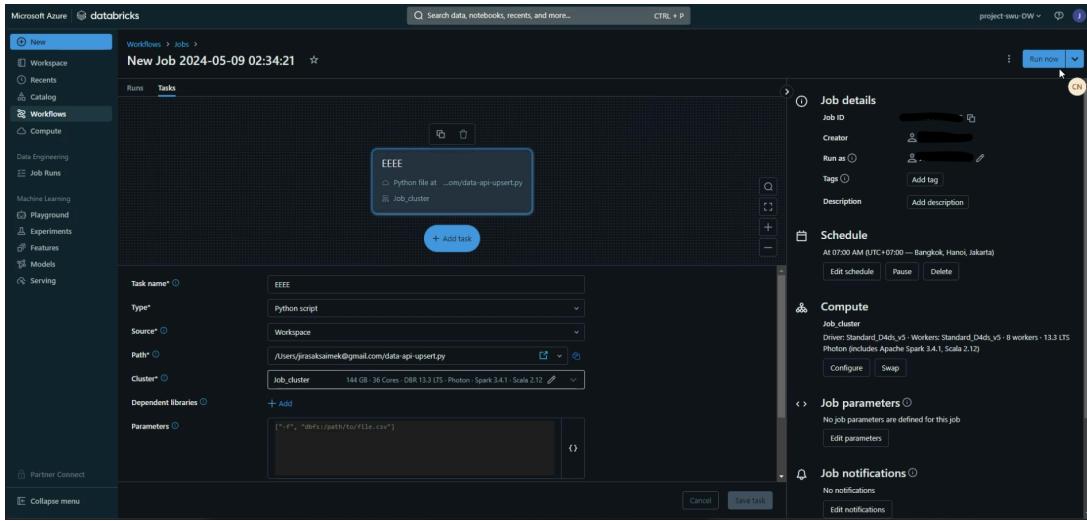
กำหนด Task name – เลือก Type : Python script – Source : Workspace – Path : path ของไฟล์ – ใส่ Dependent libraries โดยใช้ Library source : PyPI และกรอกดังภาพ



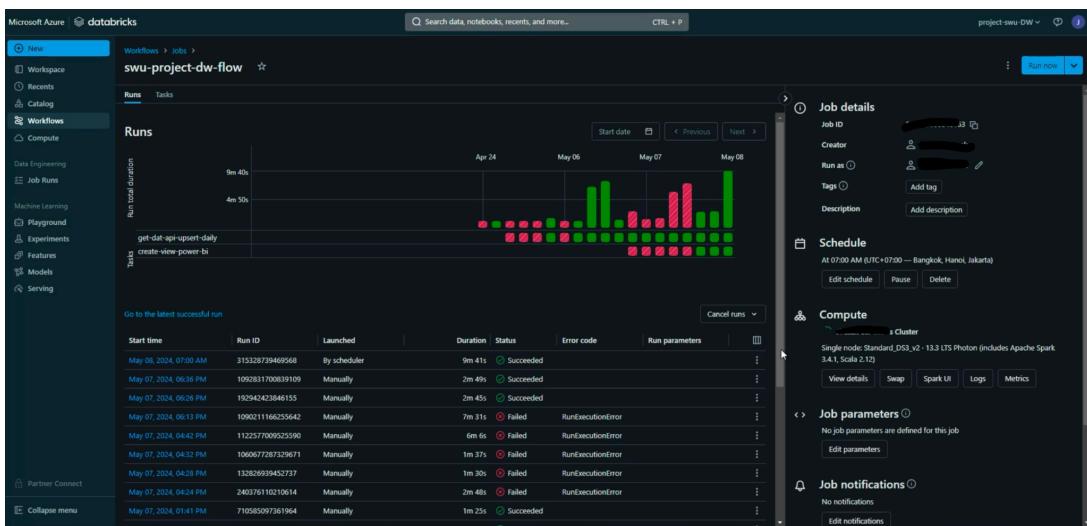
เมื่อ Save Task แล้ว คลิก Add Schedule ด้านขวามือ แล้วเลือกเวลาที่ต้องการให้รัน task นี้ โดยในโปรเจคนี้ตั้งให้ run daily เวลา 7.00 น. ประเทศไทย → save



เมื่อ save แล้วสามารถกด Run Now ที่มุมขวาบนเพื่อรันโนนตบูคตามความถี่ที่เราตั้งไว้



เมื่อ workflow ทำงานจะขึ้นโพว์ดังภาพ

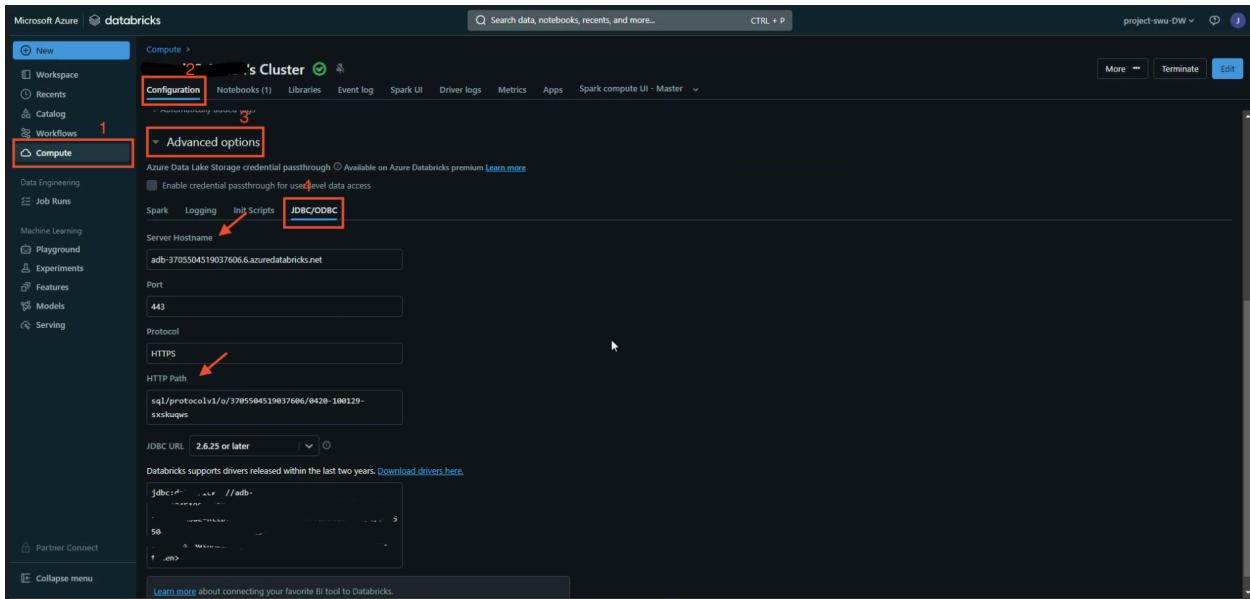


### 3. Power BI connection

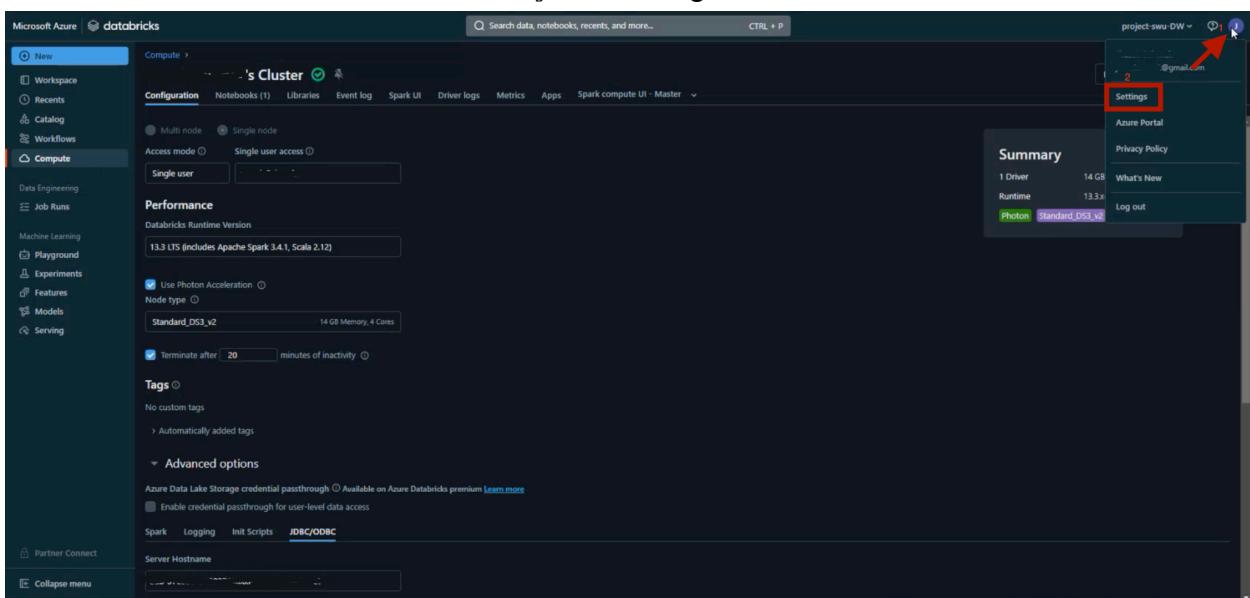
การจะเชื่อมต่อกับ Power BI ได้นั้น จะต้องใช้ Server Hostname, HTTP Path, และ Personal Access Token

- Server Hostname และ HTTP Path

ในหน้า databricks ให้คลิก compute -> configuration -> Advanced option -> JDBC/ODBC จะเจอ service hostname และ HTTP Path



- Personal Access Token  
ในหน้า databricks ให้กด icon ข้างบนตั้งรูป -> settings



ใน section “user” ให้เลือก developer -> ใน section “Access tokens” คลิก Manage แล้วสร้าง personal access token ขึ้นมาเพื่อใช้งาน

- เมื่อเปิดโปรแกรม Power BI → New Report → Get data → Azure → Azure Databricks → Connect

ใส่ Server Hostname และ HTTP Path → OK → ใส่ Personal Access Token → hive\_metastore → default → คลิกเลือกตาราง → load จะได้ตารางที่พร้อมสำหรับการทำ analytics

# Dashboard summarization



จาก dashboard สรุปได้ว่าค่าเฉลี่ยปริมาณ PM2.5 ของประเทศไทยแนวโน้มลดลงในช่วงสถานการณ์โรค COVID-19 ระบาด (ปี 2020 - 2022) และมีแนวโน้มกลับมาเพิ่มขึ้นหลังจากสถานการณ์เข้าสู่สภาวะปกติ เรียงลำดับค่าเฉลี่ยปริมาณ PM2.5 จำนวนมากไปน้อยได้ดังนี้ : ภาคเหนือ ภาคตะวันออกเฉียงเหนือ ภาคกลาง = ภาคตะวันตก ภาคตะวันออก และภาคใต้ จังหวัดที่มีค่าเฉลี่ย PM2.5 สูงที่สุดคือจังหวัดเชียงรายโดยมีค่าอยู่ที่ 33 และจังหวัดที่มีค่าเฉลี่ย PM2.5 สูงที่สุดคือจังหวัดสตูล โดยมีค่าอยู่ที่ 12 ในครัวรัมต่อลูกบาศก์เมตร

หากพิจารณาภาคที่มีค่าเฉลี่ยของค่าฝุ่นสูงที่สุดคือภาคเหนือ ปัจจัยก่อให้เกิดฝุ่นที่โดดเด่นคือการเกิดไฟป่า ในปี 2023 เกิดไฟป่าในภาคเหนือมากที่สุด โดยมีพื้นที่รวมประมาณ 133 ตร.กม. ซึ่งมากกว่าอันดับสองคือภาคกลางถึง 1.8 เท่า ในด้านปัจจัยโรงงาน ภาคเหนือมีพื้นที่โรงงานคิดเป็น 10% ของพื้นที่โรงงานทั้งประเทศไทย โดยโรงงานหลักๆ เป็นโรงงานเกี่ยวกับผลิตภัณฑ์ทางการเกษตร โรงงานหินกรวดทราย และโรงงานอุตสาหกรรม ซึ่งเป็นปัจจัยส่งเสริมให้เกิดฝุ่นละอองได้ทั้งสิ้น

ภาคที่มีค่าเฉลี่ยของค่าฝุ่นเป็นอันดับสองคือภาคตะวันออกเฉียงเหนือ ปัจจัยก่อให้เกิดฝุ่นที่โดดเด่นคือโรงงานอุตสาหกรรม โดยมีพื้นที่โรงงานอุตสาหกรรมคิดเป็นสัดส่วน 55% ของพื้นที่โรงงานทั้งหมด โรงงานหลักในภาคตะวันออกเฉียงเหนือเป็นโรงงานเมล็ดพืช อาทิ ข้าว และผลิตภัณฑ์จากข้าว คิดเป็นสัดส่วน 80% ของพื้นที่โรงงานในตะวันออกเฉียงเหนือทั้งหมด ซึ่งสามารถก่อให้เกิดฝุ่นละอองได้จากการกระบวนการเผาไหม้เชื้อเพลิง กระบวนการผลิต และ

กระบวนการขนส่ง ในปี 2023 การเกิดไฟป่าในภาคตะวันออกเฉียงเหนือนั้นมีพื้นที่โดยรวมประมาณ 40.6 ตร.กม. คิดเป็นประมาณ 1 ใน 3 ของภาคเหนือ

ภาคที่มีค่าเฉลี่ยของค่าฝุ่นเป็นอันดับสามได้แก่ภาคกลางและภาคตะวันตก ซึ่งในภาคกลางนี้มีพื้นที่เกิดไฟป่าเป็นอันดับ 2 ประมาณ 72 ตร.กม. และมีพื้นที่โรงงานเป็นอันดับสองซึ่งมีสัดส่วน 18.5% ของพื้นที่โรงงานในประเทศไทยทั้งหมด โดยมีโรงงานพลาสติกและโรงงานโลหะเป็นจำนวนมากที่สุด ในส่วนภาคตะวันตกมีพื้นที่ไฟป่าน้อยกว่าภาคกลางถึง 2 เท่า และมีพื้นที่โรงงานคิดเป็นสัดส่วนเพียง 5% แต่มีค่าเฉลี่ยของค่าฝุ่นใกล้เคียงกับภาคกลาง อาจเกิดได้จากปัจจัยอื่นที่ไม่ได้เก็บข้อมูล เช่นการเผาใหม้ผลิตภัณฑ์ทางการเกษตรจากประเทศเพื่อนบ้าน (เมียนมาร์)

ภาคที่มีค่าเฉลี่ยของฝุ่นเป็นอันดับที่สี่และห้าได้แก่ภาคตะวันออกและภาคใต้ตามลำดับ ซึ่งเป็นภาคที่หันหลังต่อการเกิดไฟป่าและสัดส่วนโรงงานอุตสาหกรรมต่ำ จึงอาจส่งผลให้มีค่าเฉลี่ยของฝุ่น PM2.5 ต่ำไปด้วย