



# TD Data Engineer Assignment

Thought Process -> Design -> Implementation

---

ไฟล์นำเสนอคำตอบของโจทย์แต่ละข้อโดยผ่าน 3 ขั้นตอนดังนี้

1.กระบวนการคิด - Thought Process From Requirement

2.การออกแบบ - Design From Thought Process

3.การดำเนินการ - Implementation From Design



# Content – สารบัญของเอกสารตัวนี้

## Question 1 – Data Pipeline Design

- 1.1 Thought Process From Requirement – การตีความต้องการจากโจทย์ที่ได้รับ
- 1.2 Design Solution From Thought Process – การออกแบบวิธีแก้ปัญหจากการตีความ
- 1.3 Implementation From Design – การดำเนินการจริงหลังการออกแบบ

## Question 2 – Text Sanitizer

- 2.1 Thought Process From Requirement – การตีความต้องการจากโจทย์ที่ได้รับ
- 2.2 Design Solution From Thought Process – การออกแบบวิธีแก้ปัญหจากการตีความ
- 2.3 Implementation From Design – การดำเนินการจริงหลังการออกแบบ

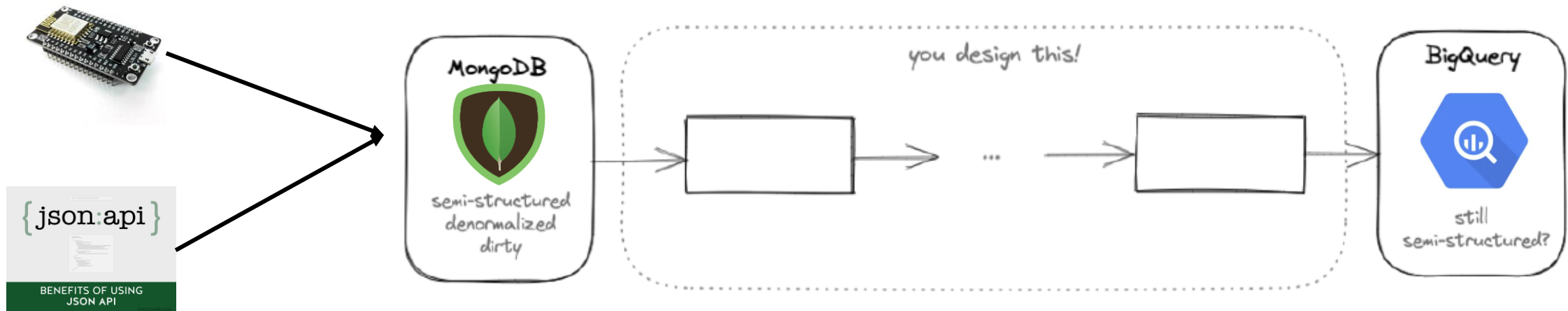
## Question 3 – SQL

- 3.1 Thought Process From Requirement – การตีความต้องการจากโจทย์ที่ได้รับ
- 3.2 Design Solution From Thought Process – การออกแบบวิธีแก้ปัญหจากการตีความ
- 3.3 Implementation From Design – การดำเนินการจริงหลังการออกแบบ

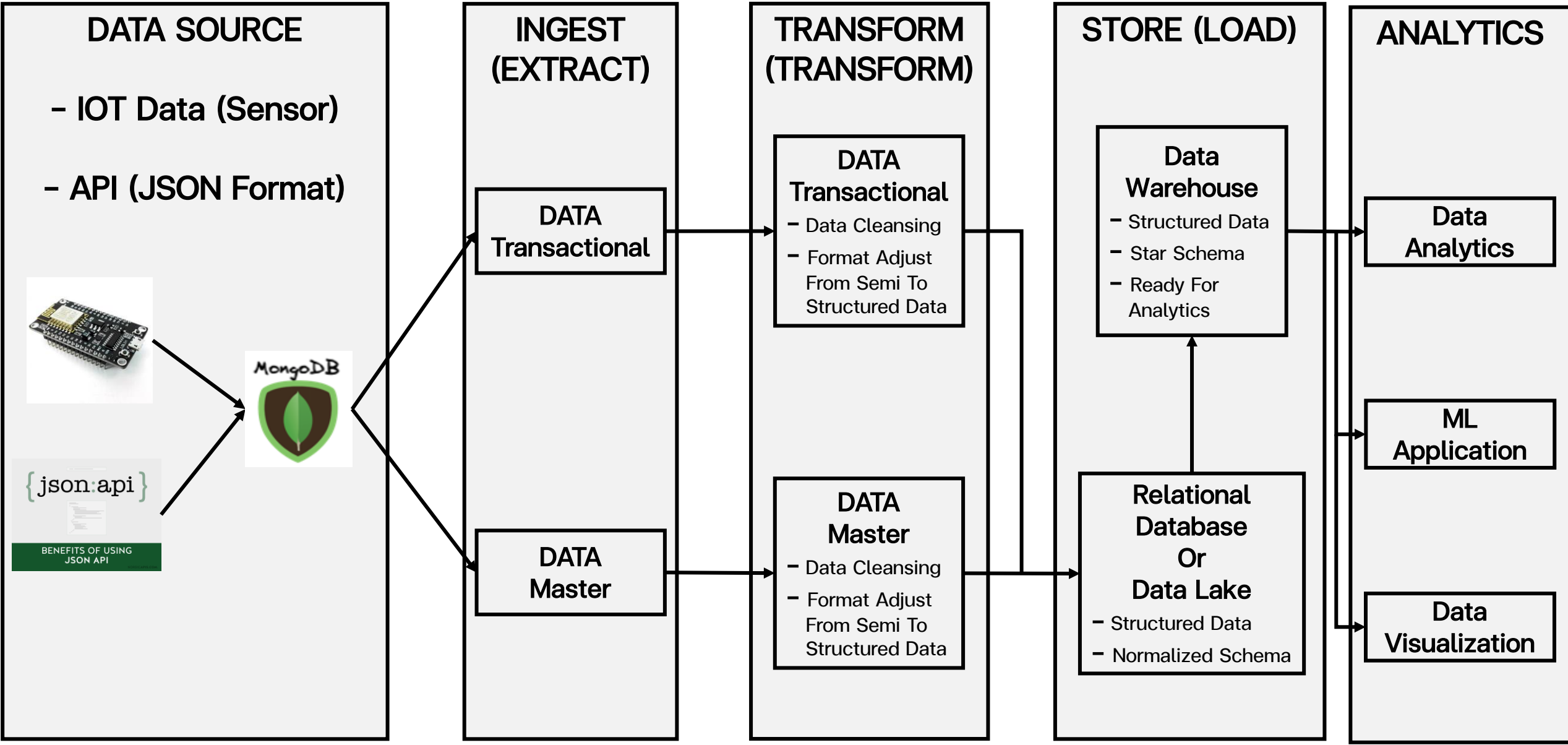
# Question 1 : Data Pipeline Design – Thought Process From Requirement

Assumption - จากโจทย์ที่ได้รับมานั้นทางผมจะขออนุญาตสร้างสถานการณ์สมมติขึ้นมาดังนี้

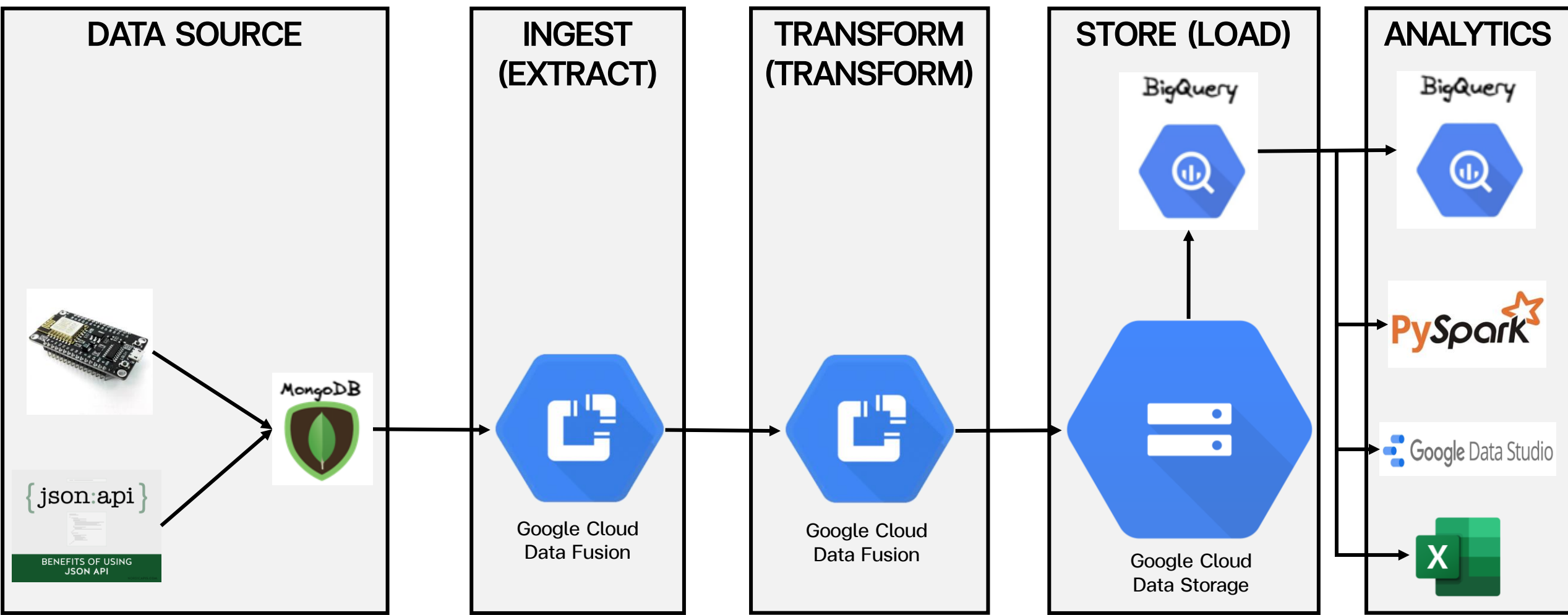
1. ณ บริษัทเอกชนแห่งหนึ่งมีแผนกำลังจะ Implement โครงการ Big Data เนื่องจากภายในองค์กรนั้นมีข้อมูลจากส่วนงานต่างๆ จำนวนมหาศาล / หลากหลายแหล่งที่มา / หลากหลายประเภทของข้อมูล / มีความถี่ในการส่งข้อมูลมีหลากหลายตั้งแต่แบบ Real - Time ไปจนถึง Schedule
2. ทางผู้บริหารมีความเห็นตรงกันว่า จะแบ่งโครงการ Big Data ออกเป็น Phase ย่อยๆ หลายๆ Phase โดยที่ Phase แรกของโครงการนั้นจะเริ่มจัดการกับข้อมูลบางประเภทที่มีความถี่ในการส่งข้อมูลเป็นแบบ Schedule ไปก่อนเพื่อนำร่องโครงการและเพื่อเป็นการประเมินความคุ้มค่าเพื่อตัดสินใจว่าจะดำเนินโครงการใน Phase ถัดๆ ไปต่อหรือไม่
3. ในส่วนของ Data Flow ที่ทางองค์กรนั้นจะตัดสินใจนำมาใช้ในโครงการ Phase แรกอย่างแน่นอน คือ จะยังข้อมูลจาก IoT / API (JSON File) เข้าสู่ Mongo DB เพื่อเป็นแหล่งต้นทางในการไปใช้ร่วมกับ Tools อื่นๆ ต่อไปจนสุดท้ายปลายทางไปออกที่ Google Big Query (Analytics Engine)
4. ด้วยเหตุนี้เองทางผู้บริหารจึงได้มอบหมายงานให้แก่ทีม Data Engineer และส่วนงานที่เกี่ยวข้องในการช่วยออกแบบ Data Architect ส่วนที่เหลือ โดยมี Architecture ต้นแบบมาให้ดังนี้



# Question 1 : Data Pipeline Design – Design Solution From Thought Process



# Question 1 : Data Pipeline Design – Implementation From Design



**Data Pipeline (Batch Job) & Data Governance**

# Question 2 : Text Sanitizer – Thought Process From Requirement

Assumption – จากโจทย์ที่ได้รับมานั้นทางผมจะขออนุญาตสร้างสถานการณ์สมมุติขึ้นมาดังนี้

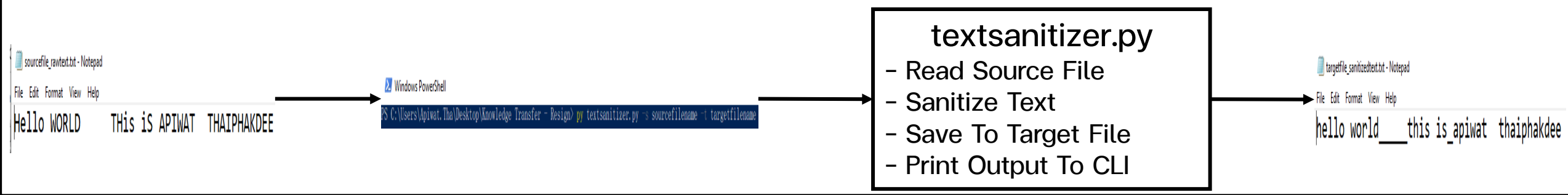
1.ทีม Data Scientist ได้มีการร้องขอให้ทีม Data Engineer ช่วยทำ ETL Pipeline สำหรับข้อมูลประเภท Text เพื่อนำไปใช้ทำ Model หรือวิเคราะห์ต่อโดยอยากให้

- Run เป็น Batch ผ่าน Command Line Interface โดยจะส่งค่า Arguments ต่างๆผ่าน CLI
- Extract ข้อมูล Text จากแหล่งที่มาต่างๆ เช่น จาก Text File หรือ Database
- Transform ข้อมูล Text ที่สกัดมาโดยให้ Sanitize Text ดังนี้ : 1.เปลี่ยนเป็นตัวพิมพ์เล็ก 2.แทนที่ Tab ด้วยคำว่า “\_” 3.นับความถี่ของตัวอักษรแต่ละตัวใน Text นั้นๆ
- Load ข้อมูล Text ที่ได้รับการ Transform แล้วนั้นเข้าไปอยู่ใน Text File ตัวใหม่อีกไฟล์หนึ่ง

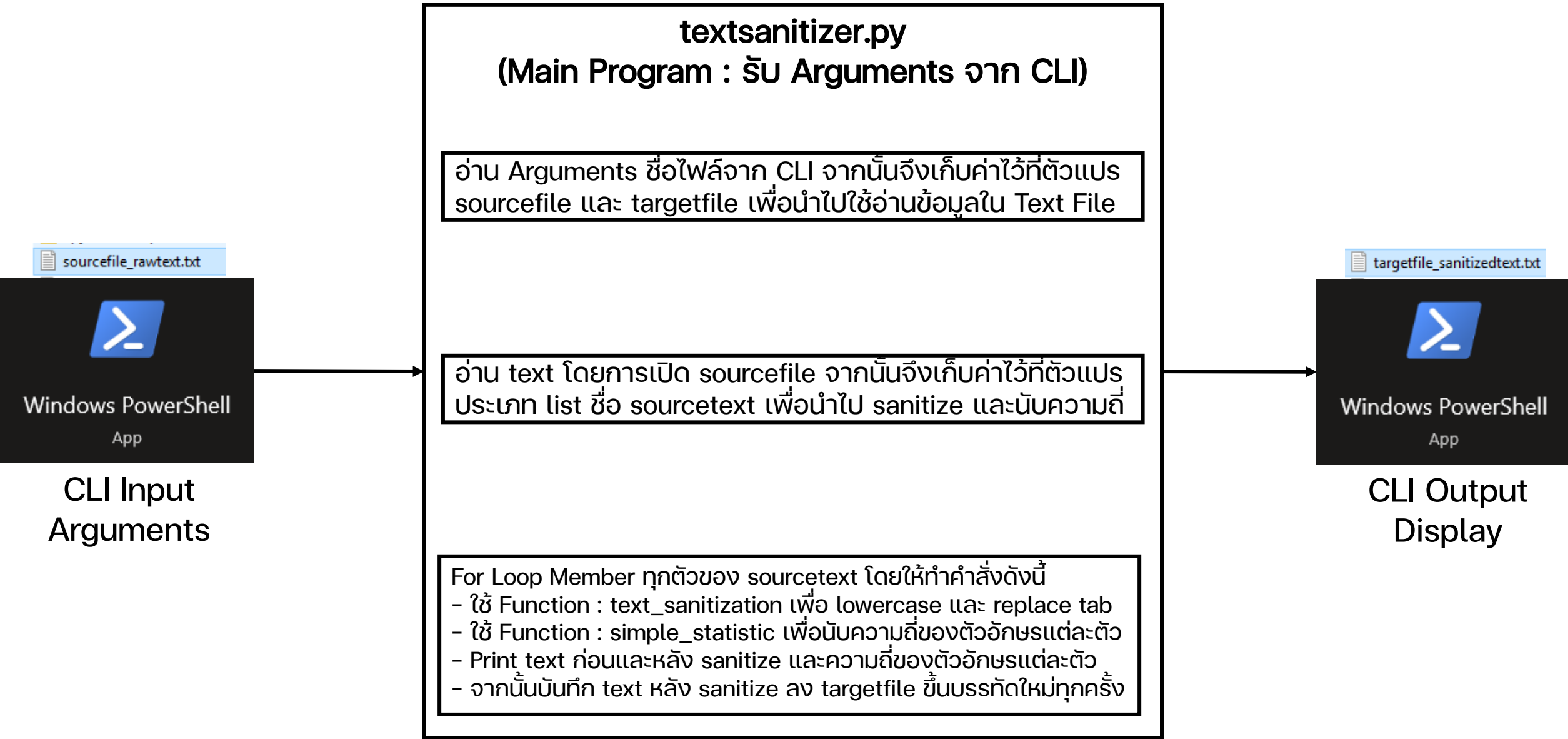
2.หลังจากทีม Data Engineer ได้รับโจทย์จากทีม Data Scientist แล้วจึงได้ทดลองทำโปรแกรมเพื่อนำไป Demo ให้ทีม Data Scientist ดูโดยสามารถทำได้ดังนี้

- รับ Arguments ซึ่งเป็นชื่อของ Source File ที่จะนำ Text มา Sanitize และ Target File ที่จะเอาไว้เก็บ Sanitized Text ผ่าน Command Line Interface
- ให้โปรแกรมอ่านข้อมูล Text ที่อยู่ใน Source File ทั้งหมดจากนั้นนำมาเก็บไว้ในตัวแปรเพื่อเตรียมทำการ Sanitize
- Sanitize Text ตามขั้นตอนดังต่อไปนี้ : 1.เปลี่ยนเป็นตัวพิมพ์เล็ก 2.แทนที่ Tab ด้วยคำว่า “\_” 3.นับความถี่ของตัวอักษรแต่ละตัวใน Text นั้นๆ
- แสดง Output ของ Text ที่ผ่านการ Sanitize และ Dictionary ความถี่ของตัวอักษรแต่ละตัวใน Text นั้นๆผ่าน CLI Console
- เก็บ Text ที่ผ่านการ Sanitize เข้าสู่ Target File โดยถ้าหากยังไม่มี Target File อยู่ก่อนแล้วนั้นก็ให้โปรแกรมสร้างไฟล์ขึ้นมาใหม่

## Application Flow



# Question 2 : Text Sanitizer – Design Solution From Thought Process

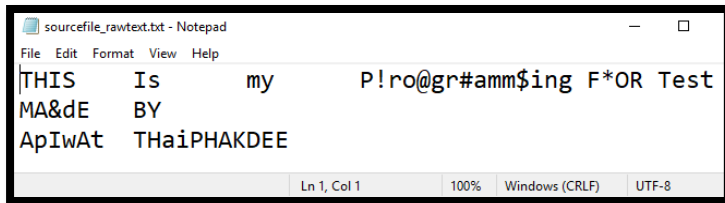





## Question 2 : Text Sanitizer – Implementation From Design

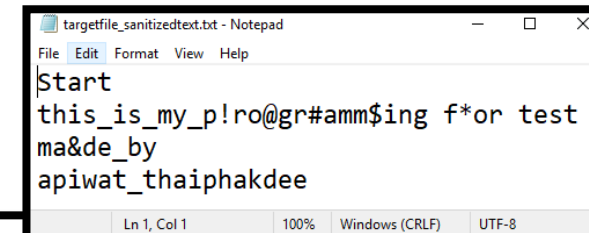
**Remark : ตรวจสอบเพิ่มเติมได้ที่ Folder : Question 2 - Text Sanitizer** โดยข้างใน Folder จะประกอบไปด้วย

- sourcefile / - targetfile / - textsanitizer.py / - textsanitizer.ipynp



 Select Windows PowerShell

```
PS C:\Users\Apiwat.Tha\Desktop\Knowledge Transfer - Resign\99.Note\๗-๑๒๒๖ TD\Question 2 - Text Sanitizer> python textsanitizer.py -s sourcefile_rawtext.txt -t targetfile_sanitizedtext.txt
```

[illegible]

Windows PowerShell

```

PS C:\Users\Apiwat.Thai\Desktop\Knowledge Transfer - Resign\99.Note\U-uasuu TD\Question 2 - Text Sanitizer> python textsanitizer.py -s sourcefile_rawtext.txt -t targetfile_sanitizedtext.txt
This is a text from source file = THIS Is my P!ro@gr#amm$ing F*OR Test
This is a text after sanitization = this is my_p!ro@gr#amm$ing f*or test
This is a number of occurrence for each alphabet not include space & tab = ['e': 1, 'p': 1, 'y': 1, 'a': 1, 'r': 1, 'h': 1, 'n': 1, 'g': 2, 'o': 2, 'm': 3, 'r': 3, 's': 3, 'i': 3, 't': 3]
Alphabet : e , Occurrence : 1
Alphabet : p , Occurrence : 1
Alphabet : y , Occurrence : 1
Alphabet : a , Occurrence : 1
Alphabet : f , Occurrence : 1
Alphabet : h , Occurrence : 1
Alphabet : n , Occurrence : 1
Alphabet : g , Occurrence : 2
Alphabet : o , Occurrence : 2
Alphabet : m , Occurrence : 3
Alphabet : r , Occurrence : 3
Alphabet : s , Occurrence : 3
Alphabet : i , Occurrence : 3
Alphabet : t , Occurrence : 3
This is a text from source file = MA&dE BY
This is a text after sanitization = ma&d_e by
This is a number of occurrence for each alphabet not include space & tab = ['m': 1, 'e': 1, 'b': 1, 'a': 1, 'd': 1, 'y': 1]
Alphabet : m , Occurrence : 1
Alphabet : e , Occurrence : 1
Alphabet : b , Occurrence : 1
Alphabet : a , Occurrence : 1
Alphabet : d , Occurrence : 1
Alphabet : y , Occurrence : 1
This is a text from source file = ApiWat THaiPHAKDEE
This is a text after sanitization = apiwat_thaiphakdee
This is a number of occurrence for each alphabet not include space & tab = ['w': 1, 'd': 1, 'k': 1, 'e': 2, 'p': 2, 'i': 2, 't': 2, 'h': 2, 'a': 4]
Alphabet : w , Occurrence : 1
Alphabet : d , Occurrence : 1
Alphabet : k , Occurrence : 1
Alphabet : e , Occurrence : 2
Alphabet : p , Occurrence : 2
Alphabet : i , Occurrence : 2
Alphabet : t , Occurrence : 2
Alphabet : h , Occurrence : 2
Alphabet : a , Occurrence : 4
PS C:\Users\Apiwat.Thai\Desktop\Knowledge Transfer - Resign\99.Note\U-uasuu TD\Question 2 - Text Sanitizer>

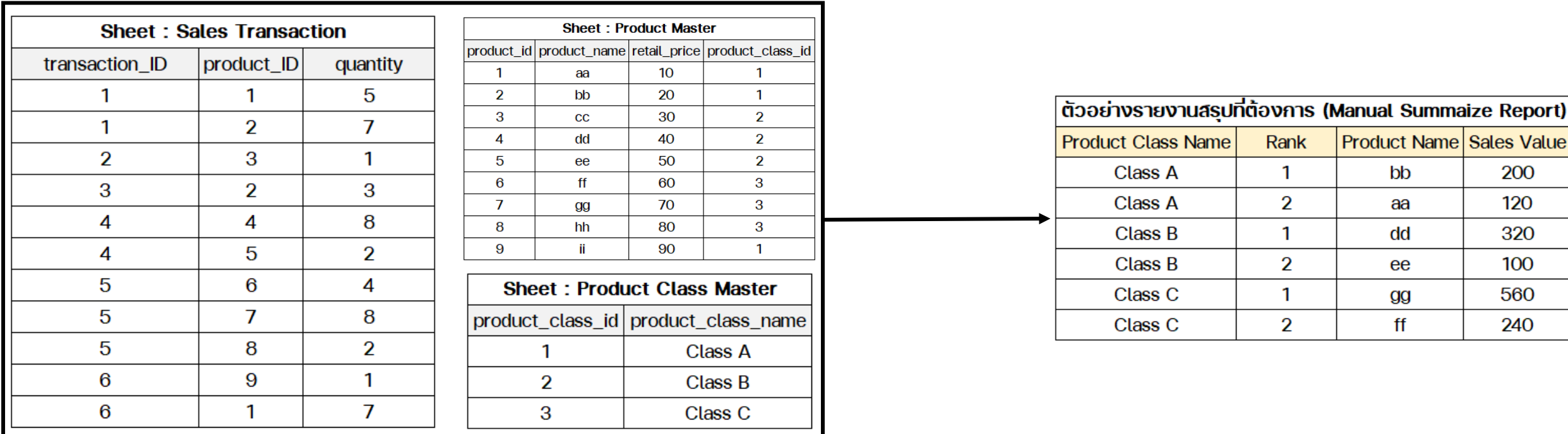
```



# Question 3 : SQL – Thought Process From Requirement

Assumption - จากโจทย์ที่ได้รับมานั้นทางผมจะขออนุญาตสร้างสถานการณ์สมมุติขึ้นมาดังนี้

- 1.ปกติแล้ว User จากแผนกฝ่ายขายและการตลาดส่วนงานวิเคราะห์ยอดขายจะต้องทำรายงานสรุปเกี่ยวกับยอดขายในทุกๆเดือน
- 2.ข้อมูลดิบที่ User มีนั้นจะอยู่ใน Excel File ซึ่งประกอบไปด้วย 3 Sheet ได้แก่ Sales Transaction / Product Master / Product Class Master
- 3.ในทุกๆเดือนทาง User จะต้องทำรายงานสรุป Product ที่มียอดขายมากที่สุด 2 อันดับแรกในแต่ละ Product Class โดยเรียงลำดับตาม Class และมีหมายเหตุว่าหากยอดขายที่เป็นมูลค่านั้นเท่ากันให้คิดว่าผลิตภัณฑ์ที่มีจำนวนขายน้อยกว่าเป็นฝ่ายที่อันดับดีกว่า
- 4.Pain Point ของ User คือ ต้อง Look Up ข้อมูลจากไฟล์ Master เข้ามาสู่ไฟล์ Transaction จากนั้นก็ต้องมาผูกสูตรคำนวณและทำสรุปจัดอันดับยอดขายที่มากที่สุดในแต่ละ Class ซึ่งเป็นการทำ Manual ด้วยมือทั้งหมด
- 5.ด้วยเหตุนี้เองทาง User จึงได้ร้องขอความช่วยเหลือให้แผนก Data ช่วยจัดทำรายงานอัตโนมัติตามรูปแบบที่ User MKT ต้องการโดยอัปเดตข้อมูลดิบที่ฐานข้อมูลของทีม Data ผ่าน App ต่างๆ



# Question 3 : SQL – Design Solution From Thought Process

## Design Solution – การออกแบบตั้งแต่ต้นทางไปยังรายงานปลายทางจะมีขั้นตอนดังนี้ (Based On PostgreSQL Database)

1.ออกแบบตารางบนฐานข้อมูลเพื่อให้ User กรอกข้อมูลโดยยึดจากโครงสร้างของไฟล์ Excel ต้นทางทั้ง 3 Sheet -> ซึ่งจะได้ Table ทั้งหมด 3 ตารางบน Database

- Table 1 : sales\_transaction ประกอบด้วย - transaction\_id / product\_id (FK) / quantity
- Table 2 : product\_master ประกอบด้วย - product\_id (PK) / product\_name / retail\_price / product\_class\_id (FK)
- Table 3 : product\_class\_master ประกอบด้วย - product\_class\_id (PK) / product\_class\_name

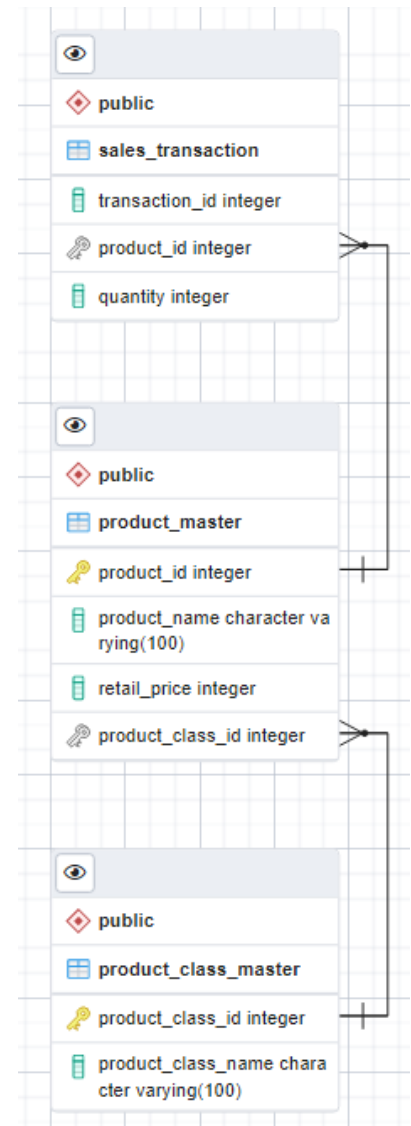
2.ออกแบบความสัมพันธ์ของตารางทั้ง 3 ตารางบนฐานข้อมูล (Schema Design) – ตามรูป ER Diagram

3.ทำการสร้าง Database และ Import ข้อมูลของแต่ละ Sheet ใน Excel ไฟล์เข้าสู่แต่ละ Table ใน Database

4.ทำการสร้าง View โดยให้มี Layout และเงื่อนไขตามที่โจทย์ต้องการ (Expected Output)

- View : sales\_summary\_report – ดัง SQL Syntax ต่อไปนี้

```
Query  Query History
1 CREATE OR REPLACE VIEW sales_summary_report AS
2 SELECT * FROM
3 (
4 SELECT product_class.product_class_name,
5        ROW_NUMBER() OVER(PARTITION BY product_class_name ORDER BY SUM(sales.quantity * product.retail_price) DESC,SUM(sales.quantity) DESC) AS rank,
6        product.product_name,
7        SUM(sales.quantity * product.retail_price) AS sales_value
8 FROM sales_transaction AS sales
9 LEFT JOIN product_master AS product ON sales.product_id = product.product_id
10 LEFT JOIN product_class_master AS product_class ON product.product_class_id = product_class.product_class_id
11 GROUP BY product_class.product_class_name,product.product_name
12 ORDER BY product_class.product_class_name ASC,rank ASC
13 )
14 AS sub_query
15 WHERE rank <= 2;
```



# Question 3 : SQL – Implementation From Design

Query Query History

1 `SELECT * FROM public.product_master`

Data output Messages Notifications

	product_id [PK] integer	product_name character varying (100)	retail_price integer	product_class_id integer
1	1	aa	10	1
2	2	bb	20	1
3	3	cc	30	2
4	4	dd	40	2
5	5	ee	50	2
6	6	ff	60	3
7	7	gg	70	3
8	8	hh	80	3
9	9	ii	90	1

Query Query History

1 `SELECT * FROM public.product_class_master`

Data output Messages Notifications

	product_class_id [PK] integer	product_class_name character varying (100)
1	1	Class A
2	2	Class B
3	3	Class C

Query Query History

1 `SELECT * FROM public.sales_transaction`

Data output Messages Notifications

	transaction_id integer	product_id integer	quantity integer
1	1	1	5
2	1	2	7
3	2	3	1
4	3	2	3
5	4	4	8
6	4	5	2
7	5	6	4
8	5	7	8
9	5	8	2
10	6	9	1
11	6	1	7

## Expected Output

Query Query History

1 `SELECT * FROM public.sales_summary_report`

Data output Messages Notifications

	product_class_name character varying (100)	rank bigint	product_name character varying (100)	sales_value bigint
1	Class A	1	bb	200
2	Class A	2	aa	120
3	Class B	1	dd	320
4	Class B	2	ee	100
5	Class C	1	gg	560
6	Class C	2	ff	240

**Remark :** สามารถตรวจสอบรายละเอียดเพิ่มเติมได้ที่ Folder : **Question 3 – SQL** โดยข้างใน Folder จะประกอบไปด้วย

- 0.Concept การตีความปัญหา / 1.รูป ER – Diagram / 2.Create DB Schema Script
- 3.ข้อมูลสำหรับ Import เข้า DB / 4.Create Expected Output View Script / 5.Database Back Up Script – สำหรับผลลัพธ์สุดท้าย (คนทุก Object จากข้อ 1-4)

*Thank  
you*

