Introduction

Data Engineering

เบื้องหลังของ Dashboard สวยๆ หรือโมเดล Machine Learning ที่ทำงานอยู่ในผลิตภัณฑ์ต่างๆ
คือ ข้อมูล ซึ่งก็ไม่ใช่ข้อมูลดิบๆ นะ แต่เป็นข้อมูลที่เรารวบรวมมาจากหลายๆ แหล่ง
แล้วเอามาปัดฝุ่น ทำความสะอาด รวมไปถึงประมวลผลให้อยู่ในรูปแบบที่เราสามารถที่จะดึงคุณค่าออกมาจากมันได้

คำกล่าวที่ว่าข้อมูลก็เหมือนน้ำมัน น้ำมันก็ต้องผ่านกระบวนการต่างๆ เพื่อให้น้ำมันนั้นสามารถนำมาใช้งานได้
ข้อมูลก็เช่นกัน จะต้องผ่านกระบวนการต่างๆ เพื่อให้ข้อมูลนั้นมีคุณค่าได้

Data Sources

ในระบบแต่ละระบบจะมีการเก็บข้อมูลเพื่อนำเอาไปใช้งานต่อ ไม่ว่าจะเป็นระบบการซื้อสินค้าออนไลน์
หรือระบบดูหนังออนไลน์ก็ตาม ซึ่งข้อมูลในระบบเหล่านั้น เราสามารถได้มาจากหลายๆ
แหล่งด้วยกัน หนึ่งในนั้นคือ ข้อมูลจากการใช้งานของผู้ใช้ระบบ (User-Generated
Data) ได้แก่ การกรอกฟอร์ม การกดปุ่มบนหน้าเว็บ การเลื่อนหน้าแอพพลิเคชั่น เป็นต้น

แหล่งต่อไปคือข้อมูลที่ได้จากเครื่องจักร หรือเครื่องคอมพิวเตอร์ (System-Generated
Data) เช่น Logs ต่างๆ หรือพวก ข้อมูล Meta ที่เอาไว้บอกลักษณะของข้อมูลอีกที

Data Pipelines

Data Pipelines คือรากฐานของความสำเร็จในการทำ Data Analytics และ
Machine Learning การเคลื่อนย้ายข้อมูลจากหลายๆ แหล่งข้อมูลที่แตกต่างกัน
และนำมาประมวลผลเพื่อที่จะให้บริบทในการวิเคราะห์ข้อมูล คือความแตกต่างระหว่างการมีข้อมูลอยู่กับการสกัดเอาคุณค่

กล่าวได้ว่า Data Pipelines เป็นเซตของขั้นตอนการประมวลผลที่เคลื่อนย้าย
และเปลี่ยนรูปข้อมูลจากหลายๆ แหล่งข้อมูลที่ต่างกัน ไปที่จุดหมายปลายทาง ที่ๆ
คุณค่าสามารถถูกนำไปใช้ได้

Databases

Database คืออะไร? Database หรือ ฐานข้อมูล คือ ระบบคอมพิวเตอร์ที่เก็บรวบรวมชุดของข้อมูลแบบมีโครงสร้าง
ที่มีความสัมพันธ์ซึ่งกันและกัน

Data Engineer

Who is Data Engineer?

ในองค์กร Data Engineer จะเป็นคนคอยดูและการไหลของข้อมูลจากแหล่งหนึ่งไปอีกแหล่งหนึ่ง
หรือจากหลายๆ แหล่งมารวมกันในที่ๆ เดียว ซึ่งในการไหลของข้อมูลนั้นๆ จะต้องสัมพันธ์กับเป้าหมายทางธุรกิจ

และสามารถนำข้อมูลเหล่านั้นไปต่อยอดแก้ปัญหาทางธุรกิจได้ เรียกได้ว่า Data Engineer นั่นคือ คนที่ส่งมอบข้อมูลที่มีคุณค่าให้กับธุรกิจก็ว่าได้

คนที่เป็น Data Engineer มีหน้าที่ๆ จะต้องเข้าใจกระบวนการต่างๆ ทางธุรกิจ และสามารถหาแนวทางต่างๆ โดยการนำข้อมูลมาช่วยเหลือในการแก้ปัญหาทางธุรกิจนั้นๆ ได้ ถ้าไม่เข้าใจกระบวนการทางธุรกิจแล้ว เราก็มีโอกาสูงมากที่จะเดินทางผิดพลาด และก่อความเสียหายให้กับองค์กรได้

การทำความเข้าใจธุรกิจนั้น คนที่เป็น Data Engineer จำเป็นที่จะต้องถามคำถามอยู่ตลอดเวลา เพื่อให้เข้าใจ และเพื่อที่จะสามารถพัฒนาต่อยอดในงานได้

ในหลายๆ องค์กร ขาด Data Engineer ไป ส่วนใหญ่จะเป็น Data Scientist ที่มาทำหน้าที่แทน ซึ่งถ้าพูดกันตามตรงแล้ว Data Scientist ส่วนใหญ่ไม่ได้มีพื้นฐานทางด้านการเขียนโปรแกรม หรือ Software Engineering มาตรงนี้จะทำให้องค์กรเสียเปรียบ และอาจจะเสียโอกาสทางธุรกิจบางอย่างได้ เพราะว่าแทนที่จะมี Data Scientist คอยวิเคราะห์ข้อมูล หรือค้นหา Insights อะไรใหม่ๆ จากข้อมูล กลับต้องให้พวกเขามาทำในส่วนของ Engineer

Data Engineer นั้นควรที่จะต้องสามารถจัดการกับข้อมูลได้ทุกขนาด จากข้อมูลเล็กๆ ไปจนถึงขนาดใหญ่ แน่นอนว่าจะต้องเริ่ม ฝึกกันตั้งแต่ข้อมูลขนาดเล็กก่อน แล้วค่อยยกระดับตัวเองขึ้นไป แล้วก็ Data Engineer ก็ควรที่จะมีมุมมองเกี่ยวกับปัญหาด้านข้อมูลที่เจอ เพื่อที่จะเลือกใช้เครื่องมือที่เหมาะสมที่สุดในการ

What do Data Engineers Do?

ถ้าเราจะกล่าวเบื้องต้นว่างานของ Data Engineer คือ การนำเอาข้อมูลที่อยู่กระจัดกระจายในองค์กรมารวมกันให้กับทีมอื่
ได้ใช้ข้อมูลกัน ก็คงจะไม่ผิดเท่าไหร่ เนื่องจากว่าเป็นภาพที่หลายๆ คนเห็น อย่างไรก็ดี
Data Engineer ภายใต้ภาพนั้นประกอบไปด้วยงานหลักๆ ที่ซ่อนอยู่ คือ

- **การออกแบบและบำรุงรักษา Database, Data Warehouse และ Data Lake** ซึ่งจะต้องมีการตัดสินใจต่างๆ ในการเลือกหรือพัฒนาเทคโนโลยีให้ตอบโจทย์เป้าหมายทางธุรกิจ และยังต้องคงไว้ซึ่งความปลอดภัยต่างๆ
- **การสร้าง Data Pipelines** เพื่อที่จะควบคุมการไหลของข้อมูลภายในองค์กร จากที่หนึ่งไปอีกที่หนึ่งเพื่อนำข้อมูลไปสร้างคุณค่าให้กับทางธุรกิจ
- **การตรวจสอบติดตาม (Monitoring) ความเสถียรของ Data Pipelines** ซึ่งแน่นอนว่าจะเป็นการทำให้มั่นใจได้ว่าทุกคนในองค์กรสามารถนำข้อมูลไปใช้ได้อย่างไม่สะดุด
- **การติดตั้งหรือพัฒนาเครื่องมือในการจัดการข้อมูล** พวก Business Intelligence (BI) ต่างๆ หรือพวกเครื่องมือสร้าง Dashboard เช่น Tableau, Qlik หรือ Superset ให้กับฝ่ายดูแลธุรกิจในองค์กร
- **ดูแลส่วน Data Governance และ Data Quality ในองค์กร** เพื่อให้คนในองค์กรได้ใช้ข้อมูลกันอย่างถูกต้อง โปร่งใส สามารถตรวจสอบสิทธิ์ในการเข้าถึงข้อมูลได้ ปกป้องความเป็นส่วนตัวของลูกค้า

Data Pipelines

รูปแบบในการสร้าง Data Pipeline

มีรูปแบบ ETL และ ELT ซึ่งทั้ง 2 patterns นี้ใช้กันอย่างกว้างขวางมากที่สุด ความแตกต่างของทั้ง 2 แบบ อยู่ที่ลำดับของตัวอักษร 2 ตัวหลัง (Transform กับ Load)

- Extract (E) การดึงข้อมูลจากแหล่งข้อมูลต่างๆ
- Load (L) การโหลดข้อมูล ไม่ว่าจะเป็นข้อมูลดิบ หรือข้อมูลที่ถูกแปลงแล้ว ไปที่ปลายทาง เช่น Data Warehouse หรือ Data Lake หรือระบบอื่นๆ
- Transform (T) เป็นการแปลงข้อมูลดิบจากแหล่งข้อมูลต่างๆ หรือจากระบบต่างๆ ให้มาอยู่ในรูปแบบที่เราสามารถเอาไปวิเคราะห์ได้ หรือเอาไปทำ Visualization ได้

ขั้นตอน E กับ L อาจจะเรียกรวมๆ ได้เป็น Data Ingestion เพราะว่าดูแล้วมีอะไรที่คล้ายๆ กัน และมีความสามารถเหมือนๆ กันอยู่ แต่อย่างไรก็ดี ตอนที่เราออกแบบ Data Pipeline เราควรที่จะแยกออกจากกันดีกว่า เพราะว่าความซับซ้อนของ E กับ L ต่างกัน

ความท้าทายในการสร้าง Data Pipeline ที่ดี

ตอนที่เราสร้าง Data Pipeline เรามักจะเจอความท้าทายหลักๆ ประมาณ 5 อย่างนี้

1. Schema เปลี่ยนแปลงอยู่ตลอดเวลา
2. Machine Failure เป็นเรื่องปกติ
3. การ Scale เพื่อรองรับข้อมูลที่มีขนาดใหญ่ขึ้นเรื่อยๆ
4. Batch vs. Real-Time
5. การทำ Data Catalog และ Data Lineage

1. Schema เปลี่ยนแปลงอยู่ตลอด

แทบจะเป็นปัญหาอับดับ 1 เลยที่ทุกคนต้องเจอ ยิ่งในยุคปัจจุบันที่โลกของซอฟต์แวร์นั้นเปลี่ยนแปลงไปเร็วมาก ธุรกิจเราก็ปรับตัวตามไปด้วย และแน่นอนว่าจะส่งผลให้ Schema ของข้อมูลนั้นเปลี่ยนไป เราก็ต้องปรับ Data Pipeline ของเราตาม

วิธีการก็มีอยู่หลายวิธีขึ้นอยู่กับสถานการณ์ เช่น ถ้าข้อมูลเราไม่เยอะเท่าไหร่ แล้วการวิเคราะห์ข้อมูลก็ไม่จำเป็นต้องเป็น Real-Time เราอาจจะ Drop Table ทิ้ง สร้างใหม่ แล้วโหลดข้อมูลตามไป แต่ถ้าข้อมูลเราเยอะมากขึ้น การทำแบบนี้ก็อาจจะเสียเวลาไปเป็นวันๆ เราก็ต้องหาวิธีอื่นมาแก้ปัญหาให้เหมาะสมกับสถาน

ระบบ Monitoring กับ Logging ดีๆ จะช่วยให้เรารู้ตัวได้ไว และการทำ Schema Version Management ก็สามารถช่วยได้เช่นกัน

2. Machine Failure เป็นเรื่องปกติ

งาน Data Pipeline ไม่ได้มีแค่ส่วนโค้ดที่เราต้องดูแล ยังมีเรื่องของ Infrastructure ที่เราใช้อีกด้วย ระบบหรือตัวเครื่องเซิฟเวอร์ มักจะมีปัญหาประมาณว่า Disk เต็ม ทำให้เขียนไฟล์ไม่ได้บ้าง หรือเครื่องค้างต้องรีสตาร์ท รวมไปถึง การที่ระบบ Network หรือ DNS ล่ม แล้วยังต้องมาคอยอัพเกรด Patch อีก (โดยเฉพาะ Security Patch) ซึ่งสิ่งเหล่านี้เกิดขึ้นเป็นเรื่องปกติ เราหาวิธีรับมือไว้เลยแต่เนิ่นๆ เลย

3. การ Scale เพื่อรองรับข้อมูลที่มีขนาดใหญ่ขึ้นเรื่อยๆ

ช่วงแรกๆ ตอนที่มีข้อมูลน้อยๆ Data Pipeline อาจจะใช้เวลาไม่ถึง 10 นาทีก็ทำงานเสร็จแล้ว พอทำเสร็จ เราก็อาจจะไม่ได้เข้ามาดูแลสักเท่าไหร่ แต่นั่งอาจจะเป็นหลุมพลาง (Pit-fall) ของหลายๆ คน เนื่องจากข้อมูลจะไหลเข้ามาเรื่อยๆ อยู่ตลอดเวลาอยู่แล้ว Data Pipeline ของเราก็จะใช้เวลานานขึ้นเรื่อยๆ เช่นกัน แล้วแต่ละองค์กรก็คงไม่ได้มีแค่ Data Pipeline เดียวแน่ ดังนั้นให้นึกถึงการ Scale เอาไว้ด้วยเลย ตั้งแต่เนิ่นๆ ยิ่งองค์กรไหนมีข้อมูลเยอะอยู่แล้ว เรื่องการ Scale เป็นเรื่องที่สำคัญมาก ตรงนี้จะรวมไปถึงการเลือก Technology ที่เหมาะสมมาใช้ด้วยเช่นกัน

อยากเน้นย้ำเพิ่มเติมว่าให้คิดไว้ตั้งแต่ช่วงแรกๆ เพราะถ้าเรายิ่งปล่อยทิ้งไว้มันก็ยิ่งเหมือน Technical Debt ที่สะสม ไปเรื่อยๆ จนวันหนึ่งเราจะไม่สามารถแก้มันได้อีกแล้ว เพราะค่าใช้จ่ายที่จะลงแรงไปปรับปรุงให้ดีขึ้นมันสูงเกินไป

4. Batch vs. Real-Time

การเลือกว่าจะเป็นการประมวลผลแบบ Batch หรือ Real-Time ให้ลองทำความเข้าใจธุรกิจและ Use Case ต่างๆ ก่อน ตรงนี้จะขึ้นอยู่กับบริบทของแต่ละองค์กร และแต่ละงาน ถึงแม้ว่าตลาดส่วนใหญ่จะเป็นเรื่องการประมวลผลแบบ Batch แต่ก็อยากให้ระลึกไว้เสมอไว้ว่างานประเภท Real-Time ก็สามารถสร้างคุณค่าให้กับองค์กรได้เช่นกัน และการสร้าง Data Pipeline แบบ Batch กับแบบ Real-Time มีการพัฒนาและการดูแลที่แตกต่างกัน ในฐานะที่พวกเราเป็น Data Engineer ควรที่จะศึกษาและทำความเข้าใจไว้ทั้ง 2 แบบ

5. การทำ Data Catalog และ Data Lineage

หัวข้อนี้มีความเกี่ยวข้องกับการทำ Data Lake ด้วย ซึ่งหลายคนมักจะมองข้าม เอาไว้ทำทีหลังก็ได้ แล้วสุดท้ายเราก็ลืม หรือไม่ก็เกิดอาการ Curse of Knowledge ของคนทำข้อมูล ที่ว่ามองแว๊บเดียวก็รู้ว่าอะไรคืออะไร อย่าไปตกหลุมพลางเข้าล่ะ ระลึกไว้เสมอเลยว่าเราไม่ได้ทำงานคนเดียว

ซึ่งถ้าเราอยากให้ทุกคนในองค์กรสามารถใช้ข้อมูลได้อย่างมีประสิทธิภาพ และเอาไปช่วยตัดสินใจในงานของพวกเขาได้ การมี Data Catalog (เก็บ Metadata ไว้เพื่อให้ค้นหาข้อมูลได้สะดวกและรวดเร็ว)

และ Data Lineage (รู้ที่มาที่ไปของข้อมูลว่ามาจากไหน ได้มาได้อย่างไร โดน Transform มาแบบไหน) ถือว่าเป็นเรื่องที่ขาดไม่ได้เลย

Data Validation in Pipelines

การ Validate ข้อมูล เราควรที่จะ Validate ตั้งแต่เนิ่นๆ ช่วงต้นๆ ของ Data Pipeline ได้เลยยิ่งดี และควรที่จะ Validate ในหลายๆ จุด

Data Engineering Practices

ออกแบบให้งานสามารถทำซ้ำ (Reproducible) ได้

ความท้าทายอย่างหนึ่งในการสร้าง Data Pipelines คือการออกแบบให้งานแต่ละงานนั้นสามารถทำซ้ำ หรือ Reproducible ได้ ซึ่งนั่นก็หมายความว่าเราสามารถที่จะรัน Pipeline ของเราใหม่ไม่ว่าจะกี่ครั้ง หรือไม่ว่าจะเวลาใดก็ตาม เราสามารถที่จะคาดหวังผลลัพธ์สุดท้ายได้เสมอ

เขียนฟังก์ชั่นการทำงานให้มีสมบัติ Idempotent

Idempotent เป็นสมบัติอย่างหนึ่งของฟังก์ชั่นทางคณิตศาสตร์ เราจะได้ผลลัพธ์เป็นค่าเดิมเสมอ ไม่ว่าเราจะดำเนินการกี่ครั้งแล้วก็ตาม ตรงนี้เป็นส่วนสำคัญในการสร้าง Data Pipelines เลยทีเดียว

ผลลัพธ์ของงานควรจะเป็น Deterministic

เราสามารถกล่าวได้ว่า งานของเราสามารถทำซ้ำได้ ถ้างานเหล่านั้นเป็น Deterministic แปลว่างานนั้นๆ จะคืนค่าผลลัพธ์เดิมเสมอ ถ้าเราใส่อินพุตค่าเดิม

ในการพัฒนา Data Pipeline ให้เก็บข้อมูลไว้ที่ Shared Storage เสมอ

ตอนที่เราจัดการข้อมูลต่างๆ ใน Data Pipeline มักจะมีการอ่านและเขียนข้อมูลจากงานต่างๆ ใน Data Pipeline อยู่เป็นประจำ และงานแต่ละงานก็มักจะแชร์ข้อมูลระหว่างกัน การที่เราเก็บไฟล์ไว้ที่ Local File System ก็เป็นวิธีหนึ่งที่สามารถทำได้ ในกรณีที่เราจัดการระบบภายในเครื่อง 1 เครื่อง

ซึ่งในทางปฏิบัติแล้วเรามักจะมีเครื่องอยู่หลายเครื่อง ระบบเป็นแบบ Distributed และมีตัว Worker หลายตัวมาทำงานร่วมกันอยู่ ทำให้ถ้าเราสั่งให้งานๆ หนึ่งเขียนไฟล์ลง Local File System ไว้ ก็จะมีความเป็นไปได้สูงว่างานอื่นๆ ที่รันอยู่คนละเครื่องจะไม่สามารถเข้าถึงไฟล์นั้นได้

วิธีที่ง่ายที่สุดในการแก้ปัญหานี้คือให้เราใช้ Shared Storage แทน ซึ่ง Worker แต่ละตัว งานแต่ละงานจะสามารถเข้าถึงไฟล์ที่ต่างคนต่างเขียนลงมาได้

Tools

เครื่องมือต่างๆ ในโลก Data Engineering

Airbyte

https://airbyte.io/

Airbyte เป็นเครื่องมือโอเพ่นซอร์สที่ช่วยทำให้ข้อมูลเชื่อมต่อ และสอดคล้องกันระหว่างแอพพลิเคชั่น API และฐานข้อมูล กับระบบปลายทางที่เก็บข้อมูลอื่นๆ อาทิ เช่น Data Warehouse หรือ Data Lake

Airflow

https://airflow.apache.org/

Airflow หรือ Apache Airflow คือโอเพ่นซอร์สแพลตฟอร์มที่เราสามารถที่จะตรวจสอบ ทำตารางเวลา และเฝ้าสังเกต กระแสงาน หรือ Workflow ขั้นตอนการประมวลผลได้

การที่เราสามารถเขียนโค้ดเพื่อกำหนด Workflow ได้ จะทำให้เราดูแลรักษา Workflow ได้ง่ายขึ้น สามารถเก็บเป็นเวอร์ชั่น สามารถทดสอบ และสามารถทำให้การทำงานร่วมกันง่ายขึ้นได้

- Airflow 2.0 มีอะไรใหม่บ้าง มาดูกัน
- เขียน Document ใน Airflow โดยใช้ doc_md

Amundsen

https://www.amundsen.io/

dbt

https://www.getdbt.com/

- dbt คืออะไรนะ?
- เริ่มต้นกับ dbt
- การจัดการโมเดลใน dbt และการทดสอบ

Debezium

https://debezium.io/

Druid

https://druid.apache.org/

Kafka

https://kafka.apache.org/

Marquez

https://marquezproject.github.io/marquez/

Pulsar

https://pulsar.apache.org/

Superset

https://superset.apache.org/