LAB ASSIGNMENT

Python Data Pipeline Engineering

สร้าง **ETL Pipeline** สำหรับข้อมูลยอดขายแบบ **Incremental** และ **Idempotent**

สถานการณ์จำลอง: Omnichannel Retail Data Warehouse

1. วัตถุประสงค์การเรียนรู้

- อธิบายและสร้างกระบวนการ Extract, Transform และ Load ได้อย่างเป็นระบบ
- ออกแบบ Star Schema และกำหนด Grain ของ Fact Table ได้ถูกต้อง
- จัดการ Missing Value, Invalid Type, Duplicate และ Referential Integrity
- พัฒนา Pipeline แบบ Idempotent และรองรับ Incremental Loading
- สร้าง Data Quality Report และจัดการ Error โดยไม่ทำให้ Pipeline ทั้งระบบหยุดทำงาน

2. สถานการณ์ปัญหา

บริษัทค้าปลีกมีข้อมูลคำสั่งซื้อจากหลายช่องทางส่งเข้ามาเป็นรอบ (batch) แต่ข้อมูลต้นทางมีรูปแบบไม่สม่ำเสมอ ฝ่ายวิเคราะห์ต้องการฐานข้อมูลแบบ Star Schema เพื่อสรุปยอดขายรายวัน จังหวัด หมวดสินค้า และช่องทางการขาย โดย Pipeline ต้องสามารถรันซ้ำและโหลดเฉพาะข้อมูลใหม่ได้อย่างปลอดภัย

3. Dataset และข้อผิดปกติที่ซ่อนอยู่

กลุ่มข้อมูล	หน้าที่	ตัวอย่างปัญหา
customers	ข้อมูลลูกค้า/Dimension	รหัสลูกค้าจากรายการขายอาจไม่พบในตารางหลัก
products	ข้อมูลสินค้า/Dimension	รหัสสินค้าไม่พบหรือสินค้า inactive
orders_batch_1-3	ข้อมูลธุรกรรม/Fact Source	ข้อมูลซ้ำ วันที่ผิด ค่าว่าง ตัวเลขเป็นข้อความ ค่าติดลบ หมวดหมู่สะกดไม่ตรง
data_dictionary	กฎคุณภาพข้อมูล	ชนิดข้อมูล ช่วงค่า และกฎอ้างอิงที่ต้องตรวจสอบ

4. ภารกิจ Lab

Task 1 - Pipeline Configuration และ Extract (15 คะแนน)

- สร้าง `@dataclass` ชื่อ PipelineConfig สำหรับ input path, output database, batch list และ error mode
- อ่านข้อมูลลูกค้า สินค้า และคำสั่งซื้อแต่ละ batch ด้วย Pandas หรือ Polars
- ใช้ try/except พร้อม logging ระบุชื่อ batch จำนวนแถว และเวลาเริ่ม-สิ้นสุด
- ห้ามแก้ข้อมูลในไฟล์ต้นฉบับด้วยตนเอง

Task 2 - Transform และ Data Quality (30 คะแนน)

- แปลงวันที่และตัวเลขอย่างปลอดภัย โดยใช้แนวคิด `errors="coerce"` หรือเทียบเท่า
- Normalize payment_method และ sales_channel; กำหนด mapping ที่อธิบายได้
- ตรวจ quantity > 0, unit_price > 0 และ discount_pct อยู่ระหว่าง 0-100
- ตรวจ customer_id และ product_id ว่ามีอยู่จริงใน Dimension
- Deduplicate ด้วย order_id โดยเก็บระเบียนที่ updated_at ล่าสุด
- สร้าง gross_amount และ net_amount
- แยกข้อมูลใช้ได้เป็น clean records และข้อมูลมีปัญหาเป็น quarantine พร้อม reason_code

Task 3 - Star Schema และ Load (25 คะแนน)

สร้าง SQLite Database อย่างน้อย 4 ตาราง โดยกำหนด Grain ของ fact_sales เป็น “หนึ่งรายการขายสินค้าที่ผ่านการตรวจสอบ ต่อ order_id”

ตารางปลายทาง	คอลัมน์ขั้นต่ำ
dim_customer	customer_key, customer_id, customer_name, province, segment
dim_product	product_key, product_id, product_name, category
dim_date	date_key, full_date, day, month, quarter, year
fact_sales	order_id, date_key, customer_key, product_key, quantity, unit_price, discount_pct, gross_amount, net_amount, payment_method, sales_channel

- กำหนด Primary Key/Unique Constraint เพื่อป้องกันข้อมูลซ้ำ
- ใช้ transaction และ upsert/insert-or-ignore อย่างเหมาะสม
- เขียน quarantine records แยกตารางหรือไฟล์ พร้อมเหตุผล

Task 4 - Idempotency และ Incremental Loading (20 คะแนน)

- รัน batch_1 หนึ่งครั้ง แล้วรันซ้ำ: จำนวน Fact ต้องไม่เพิ่ม
- โหลด batch_2 และ batch_3 ตามลำดับ โดยประมวลผลเฉพาะข้อมูลที่ยังไม่เคยสำเร็จหรือมี updated_at ใหม่กว่า
- สร้างตาราง pipeline_run_log หรือ watermark เพื่อบันทึก batch, started_at, ended_at, rows_read, rows_loaded, rows_rejected และ status
- แสดงหลักฐานผลการรันอย่างน้อย 4 รอบ: batch_1, batch_1 ซ้ำ, batch_2, batch_3

Task 5 - Orchestration และสรุปผล (10 คะแนน)

- สร้างฟังก์ชัน `run_pipeline(config)` ที่เรียก extract -> transform -> validate -> load
- หากหนึ่งแถวผิดให้ quarantine; หากไฟล์/batch ใช้งานไม่ได้ให้บันทึก failed และไม่ทำลายข้อมูลที่โหลดสำเร็จแล้ว
- สรุป KPI: rows read, valid, rejected, duplicated, loaded และยอดขายสุทธิรวม
- เขียน Reflection 5-8 บรรทัด: เหตุใด Availability จึงมักสำคัญกว่า Strictness ใน Production Pipeline

5. ผลลัพธ์ที่ต้องส่ง

ไฟล์	รายละเอียด
pipeline.py หรือ notebook.ipynb	Source code ที่รันได้ตั้งแต่ต้นจนจบ
retail_dw.db	SQLite Database หลังโหลดครบ 3 batch
quarantine.csv	รายการที่ไม่ผ่าน พร้อม reason_code และ source_batch
pipeline_run_log.csv หรือ table	ประวัติการรันและตัวชี้วัดแต่ละรอบ
README.md	วิธีติดตั้ง วิธีรัน โครงสร้าง Star Schema และคำตอบ Reflection

6. Acceptance Tests

1. Pipeline รันครบ 3 batch โดยไม่แก้ source data
2. order_id ใน fact_sales ไม่ซ้ำ
3. Foreign key ของ Fact เชื่อม Dimension ได้ทุกแถว
4. quantity, unit_price และ net_amount ของ Fact ไม่ติดลบ

5. การรัน batch เดิมซ้ำไม่เพิ่มจำนวนแถว Fact
6. ทุกแถวที่ถูกปฏิเสธมี reason_code
7. Run log แสดงจำนวน read = valid + rejected ก่อนการ deduplicate หรืออธิบายสูตรที่ใช้ไว้อย่างชัดเจน