DataOps Foundation

From DevOps to Data Production

กระบวนการนำเข้าข้อมูลสินเชื่อสู่คลังข้อมูล

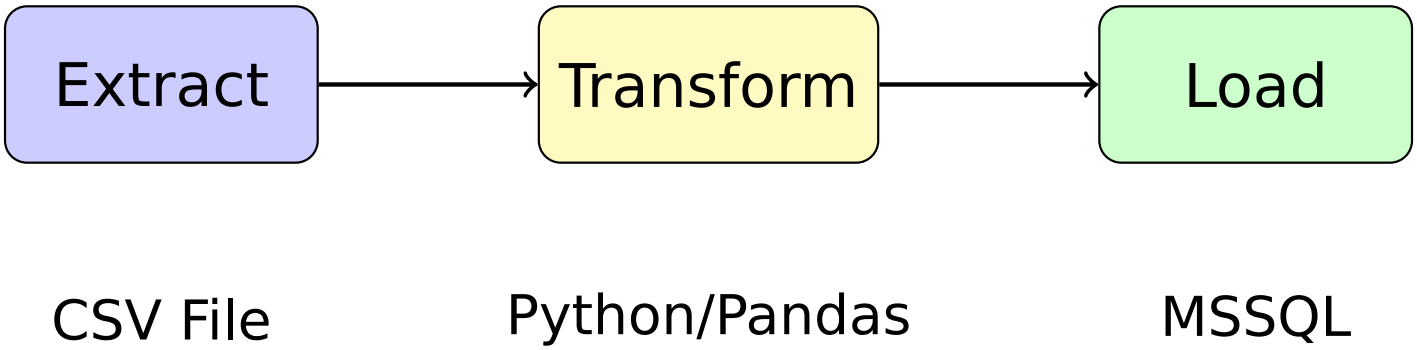
mindData-Technology

บริษัท มายด์ดาต้า เทคโนโลยี จำกัด

December 15, 2025

สารบัญ

ภาพรวมของ ETL Pipeline



- **Extract**: อ่านข้อมูลจาก CSV (LoanStats_web.csv)
- **Transform**: ทำความสะอาด แปลง และสร้าง Star Schema
- **Load**: นำเข้าสู่ MSSQL Data Warehouse

ขั้นตอนที่ 1: Extract - อ่านข้อมูล

กำหนด Data Type อัตโนมัติ

```
def guess_column_types(file_path):
    df = pd.read_csv(file_path)
    for column in df.columns:
        # Check datetime: YYYY-MM-DD HH:MM:SS
        # Check date: YYYY-MM-DD
        # Use pandas infer_dtype()
    return column_types
```

ฟังก์ชันนี้จะ:

- ตรวจจับรูปแบบวันที่และเวลาอัตโนมัติ
- กำหนด Data Type ที่เหมาะสมให้แต่ละคอลัมน์

ขั้นตอนที่ 2: Transform - จัดการ Missing Values

กลยุทธ์การจัดการค่าว่าง (2 ระดับ)

ระดับที่ 1: กรองคอลัมน์

- ลบคอลัมน์ที่มี NULL $>$ 30%

```
missing_percentage = df.isnull().mean() * 100
```

ระดับที่ 2: กรองแถว

- เลือกคอลัมน์ที่มี NULL $\leq$ 26 แถว
- ลบแถวที่เหลือที่ยังมีค่าว่าง

```
noNull_df = df_selected.dropna()
```

ขั้นตอนที่ 2: Transform - แปลงข้อมูล

การแปลงข้อมูลให้พร้อมใช้งาน

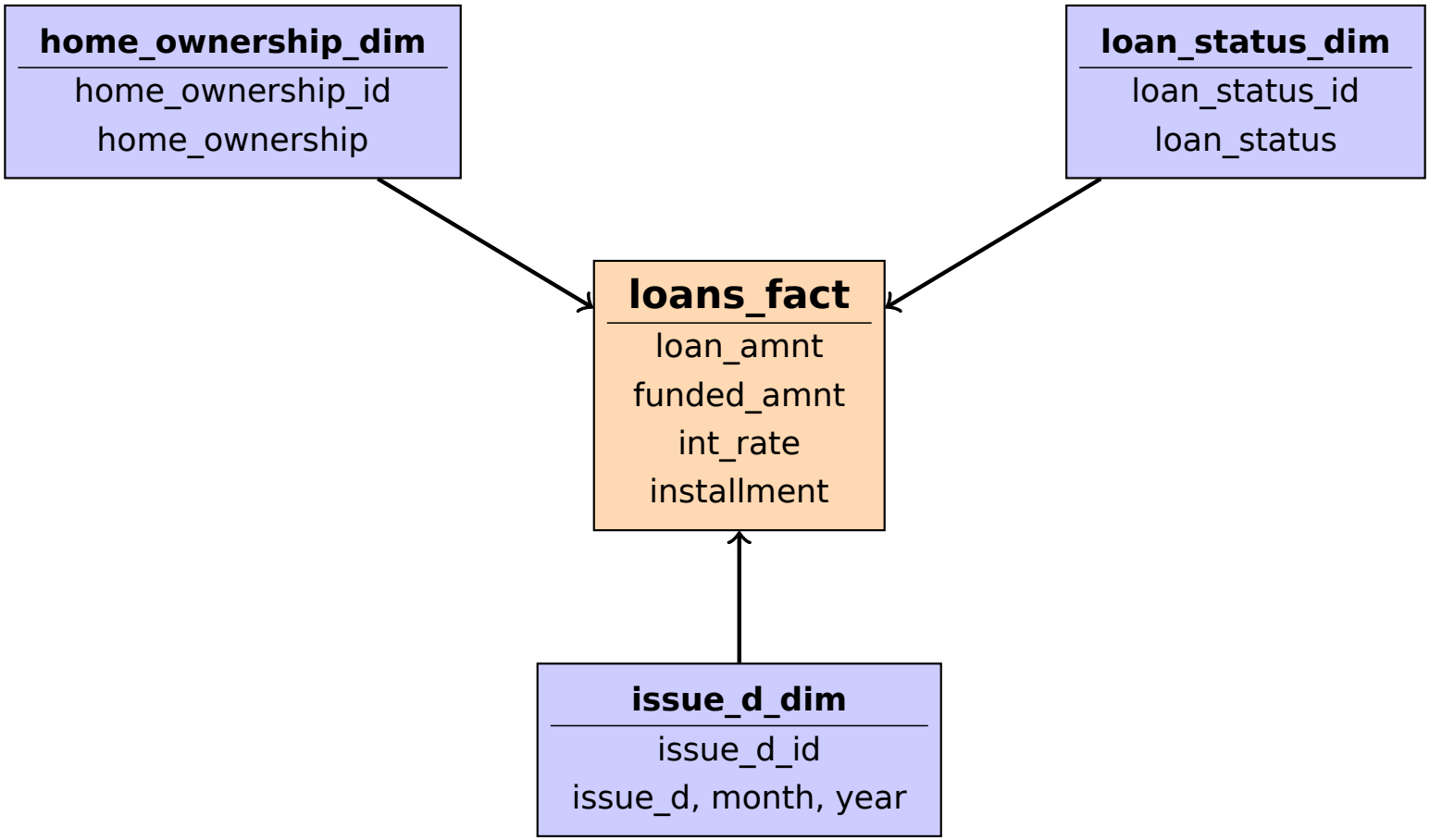
1. แปลงวันที่

```
df['issue_d'] = pd.to_datetime(
    df['issue_d'], format='%b-%Y')
# "Jan-2020" -> 2020-01-01
```

2. แปลงอัตราดอกเบี้ย

```
df['int_rate'] = df['int_rate']
    .str.rstrip('%').astype('float') / 100.0
# "15.5%" -> 0.155
```

ขั้นตอนที่ 2: Transform - Star Schema Design



ขั้นตอนที่ 2: Transform - สร้าง Dimension Tables

ขั้นตอนการสร้าง Dimension Table

Step 1: ดึงค่าที่ไม่ซ้ำกัน

```
home_ownership_dim = df[['home_ownership']]
    .drop_duplicates().reset_index(drop=True)
```

Step 2: สร้าง Primary Key

```
home_ownership_dim['home_ownership_id'] = \
    home_ownership_dim.index
```

ผลลัพธ์:

home_ownership_id	home_ownership
0	RENT
1	OWN
2	MORTGAGE

ขั้นตอนที่ 2: Transform - สร้าง Fact Table

การ Map Foreign Key

Step 1: สร้าง Dictionary สำหรับ Mapping

```
home_ownership_map = home_ownership_dim \
    .set_index('home_ownership')['home_ownership_id'] \
    .to_dict()
# {'RENT': 0, 'OWN': 1, 'MORTGAGE': 2}
```

Step 2: แทนที่ค่าด้วย Foreign Key

```
loans_fact['home_ownership_id'] = \
    loans_fact['home_ownership'].map(home_ownership_map)
```

ขั้นตอนที่ 3: Load - นำเข้าสู่ MSSQL

เชื่อมต่อและนำเข้าข้อมูล

```
from sqlalchemy import create_engine

engine = create_engine(
    f'mssql+pymssql://{user}:{pwd}@{server}/{db}')

# Load Dimension Tables
home_ownership_dim.to_sql('home_ownership_dim',
    con=engine, if_exists='replace', index=False)
loan_status_dim.to_sql('loan_status_dim',
    con=engine, if_exists='replace', index=False)
issue_d_dim.to_sql('issue_d_dim',
    con=engine, if_exists='replace', index=False)

# Load Fact Table
loans_fact.to_sql('loans_fact',
    con=engine, if_exists='replace', index=False)
```

การตรวจสอบความถูกต้อง

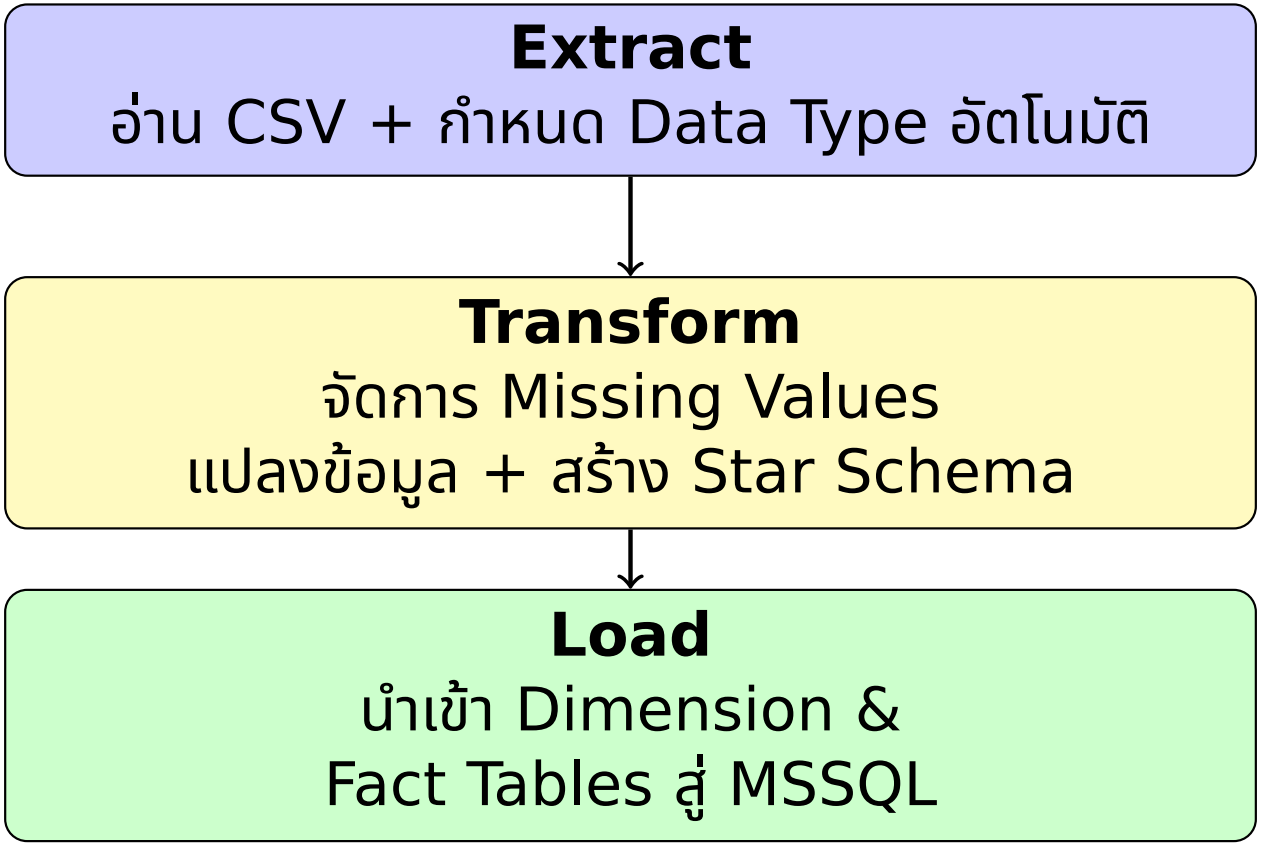
ทดสอบ Join ทุก Table

```
final_df = pd.merge(loans_fact,
    home_ownership_dim, on='home_ownership_id')
final_df = pd.merge(final_df,
    loan_status_dim, on='loan_status_id')
final_df = pd.merge(final_df,
    issue_d_dim, on='issue_d_id')
```

ตรวจสอบ:

- จำนวนแถวต้องเท่ากับข้อมูลต้นฉบับ
- ค่า Measure ต้องตรงกัน
- ไม่มีค่า NULL หลัง Join

สรุป


Extract
อ่าน CSV + กำหนด Data Type อัตโนมัติ
Transform
จัดการ Missing Values
แปลงข้อมูล + สร้าง Star Schema
Load
นำเข้า Dimension &
Fact Tables สู่ MSSQL

ขอบคุณครับ

Q&A