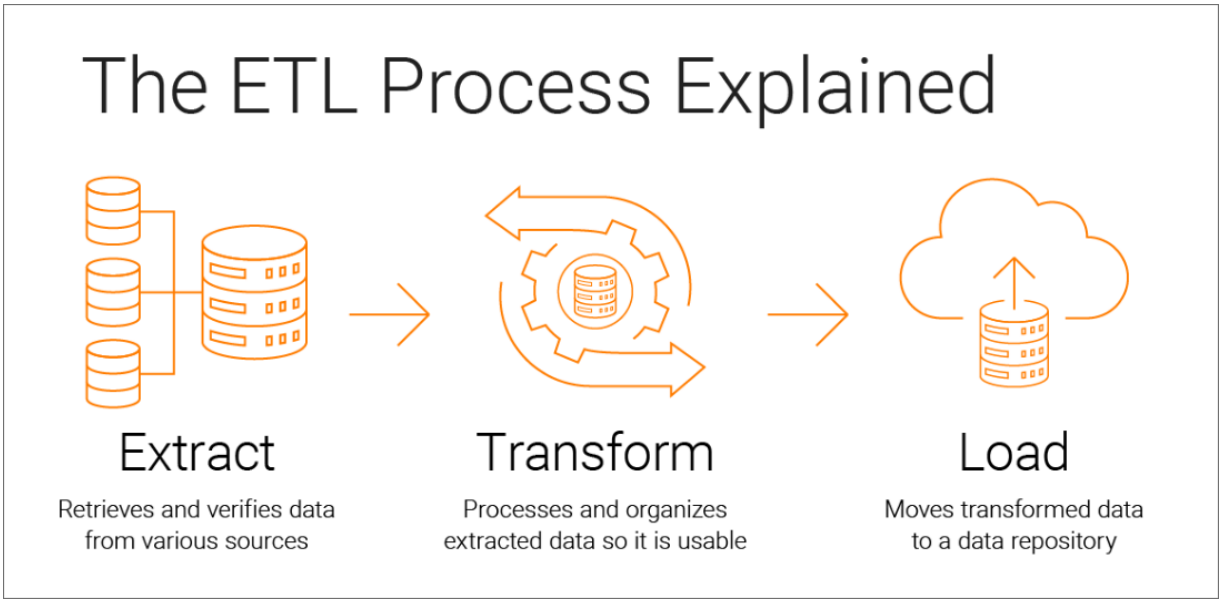
Methodology:

ETL
E- Extract
T- Transform
L – Load



ETL process

Web Scraping is first step.

Later on, we are going to transform the dataset into the clean dataset before we use.

Python Script could be considered.

After that, the output file will be load.

How reliability about this dataset?

Accuracy = 100% - Error Rate

Error Rate = |Observed Value - Actual Value|/Actual Value × 100

Extract

Use the keyword to put into the search bar.

Sdn Bhd
Enterprise
Agency
Pte. Ltd
Firm
Business
Factory
Manufacturing
Production
Engineering
Service
Hotel
Kedai
Trading
Shop
Kopitiam
Gerai
Pasar
Office
Export
Supplier
Technology
Chemical
Logistics
Storage
Perniagaan
Syarikat
Furniture
Industries
Plastics
Art
Graphics
Distributor
Energy
Palm
Roti

Event
Wedding
Makeup
Interior
GYM
Cosmetics
Automotive
Aircon
Contractor
Online
Sale
Sport
Wear
Lab
Pet
Investment
Islamic
Appliance
Travel
Club
Fashion
Printing
Dairy
Wholesale
Seafood
Soap
Telecommunication
funeral
Customer
Dessert
Kopi
Durian
Fruit
Vegetable
Meat
Farm
Nutrition
Pharmacy
Religion
ERP
SCADA
SENSOR
Automation
Treatment
Frozen
Bhd

Design
Architect
Law
Dance
Music
Ceramic
Textile
Rubber
Beverage
Steel
Iron
Hardware
Paint
Computer
Fix
Reflexsologi
Physiotherapy
Clinic
Petroleum
Branch
Café
Restaurant
Food
Store
Mall
Holding Bhd
Cooperation/ Co.
Syarikat2
SME
Marketing
Communication
Equipment
CNC
Commerce
Warehouse
Job
Car
Property
Grocery
Studio
Account
Health
Agriculture
Construction
Financial
Inc.

Group
Education
HVAC
Mechanical
Electronic
Electrical
Civil
Quantity
Insurance
Information Technology
Consultant
Management
Accessories

Add Instant Data Scraper Web Chrome Extension into browser:

Web Scraping Screenshot

The file will be stored into CSV file.

Transform

Since there are about 100++ files (CSV format), we are going to join all. By using Python script,

```
import pandas as pd
import glob
import os


files = os.path.join("C:\\Users\\USer\\Downloads\\
Company_DataSet_Scraping_From_Google_Map-main\\
Company_DataSet_Scraping_From_Google_Map-main\\", "1 (*).csv")

files = glob.glob(files)

print("Resultant CSV after joining all CSV files at a particular location...")

# joining files with concat and read_csv
df = pd.concat(map(pd.read_csv, files), ignore_index=True)
```

```
#print(df)
df.to_csv('out.csv')
```

Now we have completed dataset…

The problem still exists that, the column is not tally and some of Unicode data is unreadable and some of the rows are duplicated.

Problem:

1- Duplicated rows
2- Unicode Data display???
3- Not tally column

We are required to clean any data which is intersection.

So, I decide use SQL to solve the abovementioned problem. The reason I have come across that using Python Pandas Library is very hard to duel with this kind of data issue.

If I have imported CSV into SQL, therefore I can use SQL command…

DELETE FROM out where field3 in (SELECT field3 FROM out GROUP BY field3 HAVING COUNT(*)>1) ;

SQL Code

-- SELECT *, COUNT(*) FROM Testing5 where field3 == 'TKM Car Accessories';

-- SELECT field1, field3, COUNT(*) FROM Testing5 GROUP BY field3 HAVING COUNT(*)>1;

-- 1221Design

DELETE FROM Testing5 where field1 in (SELECT field1 FROM Testing5 GROUP BY field3 HAVING COUNT(*)>1) ;

Response

Execution finished without errors.
Result: query executed successfully. Took 54ms
At line 1:
-- SELECT *, COUNT(*) FROM Testing5 where field3 == 'TKM Car Accessories';
-- SELECT field1, field3, COUNT(*) FROM Testing5 GROUP BY field3 HAVING COUNT(*)>1;
-- 1221Design

DELETE FROM Testing5 where field1 in (SELECT field1 FROM Testing5 GROUP BY field3 HAVING COUNT(*)>1) ;

```
WITH cte AS (
   SELECT
      id,
          field3,
      ROW_NUMBER() OVER (
         PARTITION BY
               id,
                                  field3
         ORDER BY
               id,
                                  field3
      ) row_num
    FROM
      Testing5
)
DELETE FROM cte
WHERE row_num > 1;
```

SELECT DISTINCT field3 FROM Testing5;

--7-11

-- AEON

```
DELETE FROM Testing5 WHERE rowid NOT IN (SELECT min(rowid) FROM Testing5
```

```
GROUP BY field3);
SELECT *, COUNT(*) FROM Testing5 where field3 == 'Aeon Batu Pahat';
```

Reference:

https://dba.stackexchange.com/questions/116868/sqlite3-remove-duplicates

DELETE NULL DATA

```
delete from Testing5 where (field10='' OR field10 IS NULL) AND (field16='' OR field16 IS
NULL) AND (field17='' OR field17 IS NULL)  ;
```

Reference: https://qawithexperts.com/article/sql/import-csv-into-sql-server-with-query-or-without-query-using/265

Phrase 2

How to separate the dataset from Listed Company from the dataset?

Bursa Malaysia's Listed Company list
Source: https://www.bursamalaysia.com/trade/trading_resources/listing_directory/main_market

Then, the rest will become the SMEs.

Therefore, we need to scrape the data from Bursa Malaysia.

SQL

```
request = cursor.execute
('SELECT Testing5.field3 As batupahat_name,
 ListedCompany.field3
AS listed_name
 FROM Testing5 JOIN ListedCompany
ON Testing5.field3 LIKE ListedCompany.field3;')
```

I use join table to find out which company is sdn bhd or which is public limited company.

What is the improvement/ suggestion for this method?

Since we are big data, we can recruit the data scientist or specialist use the advanced or latest technology/ algorithm to find the hidden data among the dataset.