Generation

Mid-Term Project

Generation

香 HONG KONG 港

Team Green

Jacky - Winson - Jeff

Data Analysis Tools

1. Market researchers more understand competition
2. Client's requirements and objectives for market
3. Identify the key data points (product price, promotion, update date etc..)
4. Determine the frequency of data collection and update schedule

Benetifs (Whats it bring?)

1. Comprehensive Market Insights
2. Real-time Data Collection
3. Competitive Advantage
4. Customization and Flexibility
5. Actionable Insights
6. Scalability and Future Growth

Brand

Wellcome 惠康

Wellcome is Hong Kong's the longest established supermarket chain. Wellcome's network of over 280 stores in Hong Kong.

PARKNSHOP 百佳

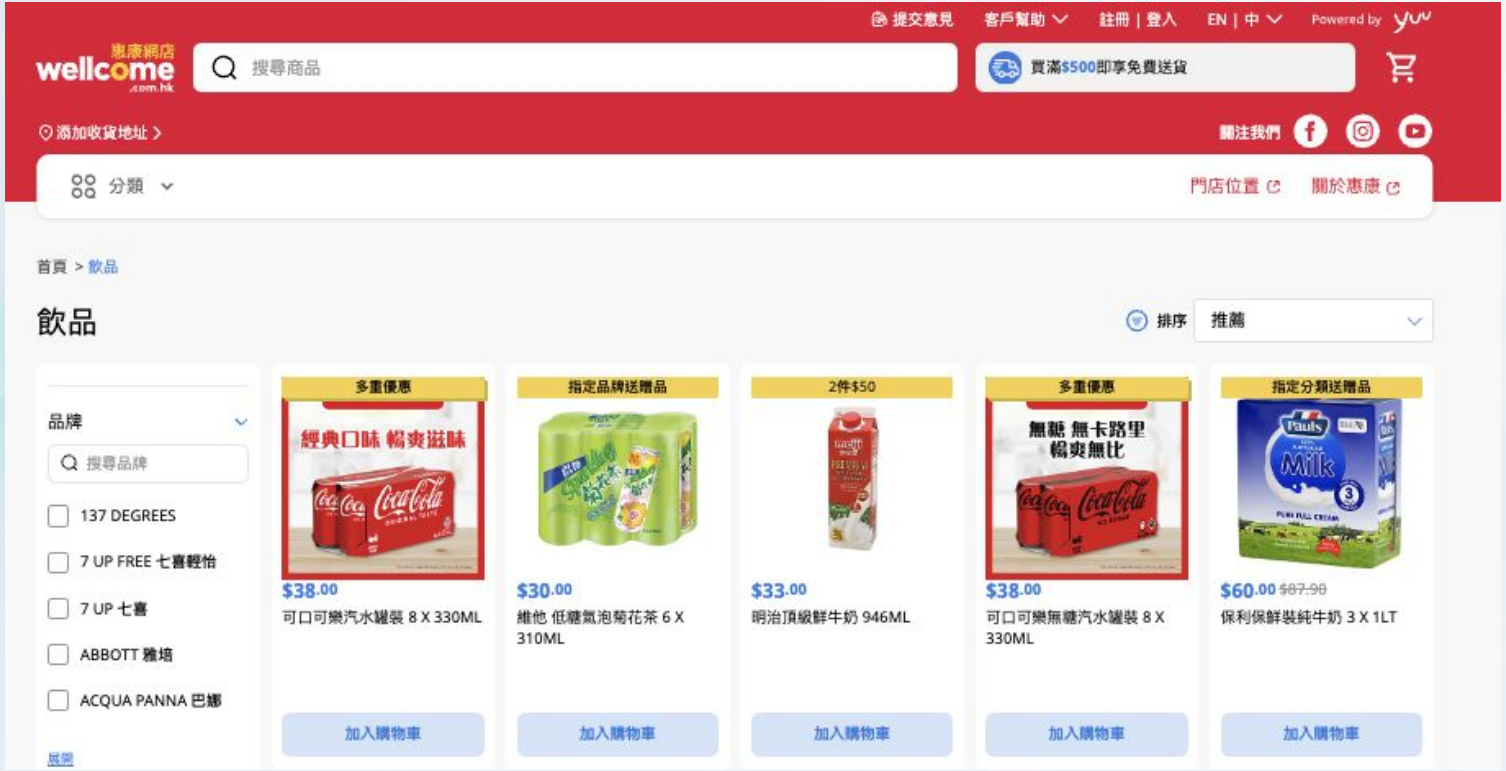
PARKnSHOP Supermarket is part of A.S. Watson Group, the world's largest international health and beauty retailer.

Ztore 士多

Ztore are an online supermarket with convenient delivery service, selling quality groceries and home essentials.

Scraping Tool

Wellcome 惠康
web crawler



```python
import pandas as pd
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.chrome.options import Options
import time
import datetime
```

Import the library which required in web crawler

```python
driver = webdriver.Chrome()
current_page= 1
url = "https://www.wellcome.com.hk/zh-hant/category/100002/"+ str(current_page) +".html"

try:
    driver.get(url)
except:
    print('url not found')
    exit

time.sleep(2)
```

Use selenium to get the Wellcome beverage website

Return error if the website is unavailable.

Create empty list and dictionary for further use.

```
remark=[]
name = []
selling_price=[]
original_price=[]
link=[]
selling_price_base=[]

df_dict={
    "name" : name,
    "selling_price" : selling_price,
    "original_price" : original_price,
    "remark" : remark,
    "link" : link
}
```

Find the last page element in the website, convert it to a integer and store in a variable.

```
total_page = driver.find_element(By.XPATH,'//a[@class="last cursor num-box"]')
total_page = int(total_page.text)
#print('total page : ', total_page)
```

```
while total_page >= current_page:
    url = "https://www.wellcome.com.hk/zh-hant/category/100002/"+ str(current_page) +".html"
    try:
        driver.get(url)

        products = driver.find_elements(By.XPATH,'//a[@class="a-link router-link ware-wrapper"] ')
        for product in products:
```

Use while loop to access the first url(Page 1). Use XPATH to find the html class element which contain the whole product independently. Use for loop to obtain the data from the path.

The web crawling process and method will show on the next page.

```
        current_page = current_page+1
    except:
        print('url not found')
        exit
```

At the end of the while loop, when all of the current page data is obtain, selenium will get in the next page

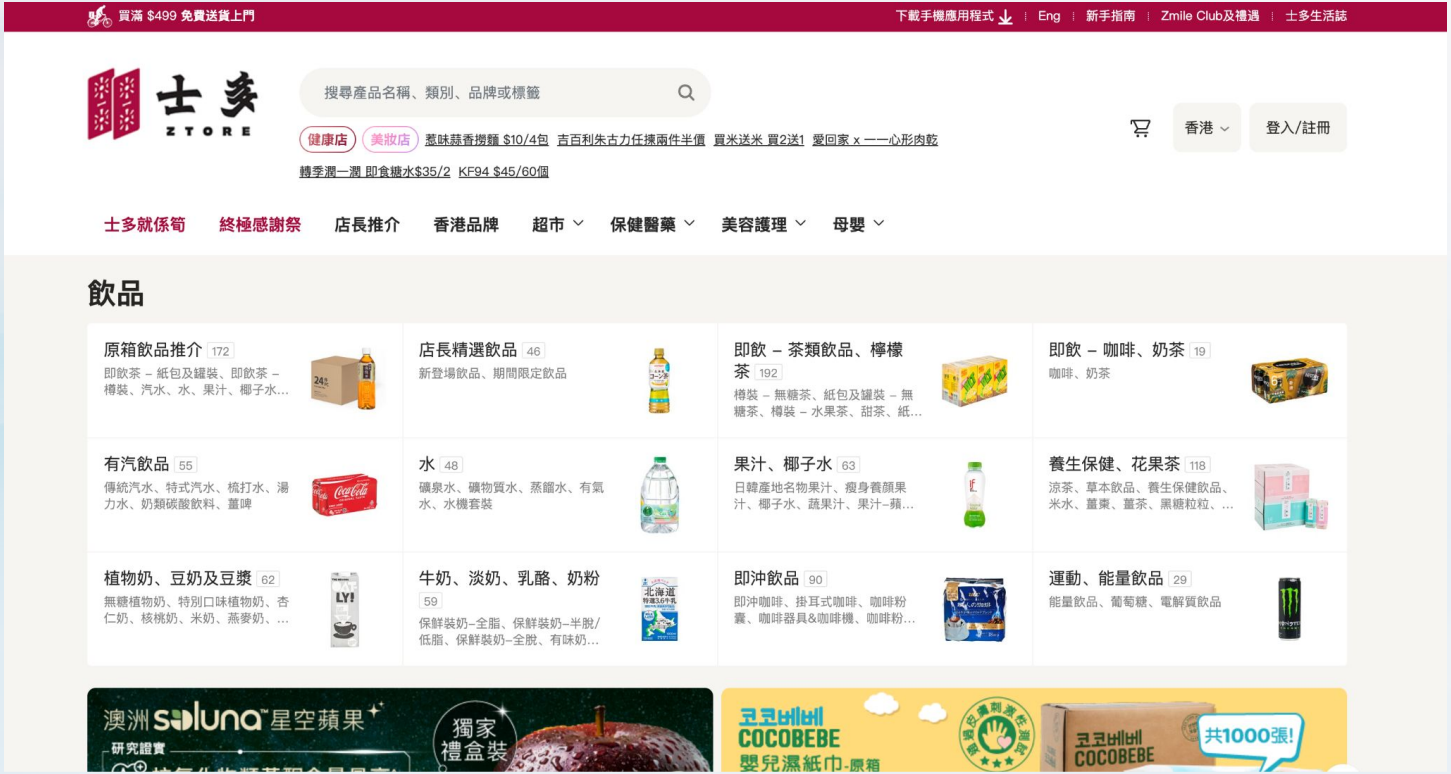
```
        try:
            remark.append(product.find_elements(By.XPATH, './/div[contains(@class, "pro tag")]')[0].text)
        except:
            remark.append(None)

        try:
            name.append(product.find_elements(By.XPATH, './/div[contains(@class, "name")]')[0].text)
        except:
            name.append(None)

        try:
            price_base = product.find_elements(By.XPATH, './/div[contains(@class, "price")]')[0].text.split("\n")[0]
            small_base = product.find_elements(By.XPATH, './/div[contains(@class, "price")]')[0].text.split("\n")[1]
            total=price_base+small_base
            selling_price.append(total)

        except Exception as error:
            selling_price.append(None)
        try:
            original_price.append(product.find_elements(By.XPATH, './/span[contains(@class, "line-price")]')[0].text)
        except:
            original_price.append(None)

    product_link =driver.find_elements(By.XPATH, './/a[contains(@class, "a-link router-link ware-wrapper")]')
    for product in product_link:
        link.append(product.get_attribute('href'))

    current_page = current_page+1
```

In for loop, use XPATH to find the different part of data in the website, store that data with append in correspond list. If the data is empty or not found, the correspond list would append None as value.

```
87    df = pd.DataFrame(df_dict)
88
89    if __name__ == '__main__':
90        current_time_str = datetime.datetime.now().strftime("%Y_%m_%d")
91        print(df)
92        df.to_csv(current_time_str + "_wellcome.csv", encoding="UTF-8")
```

In for loop, use XPATH to find the different part of data in the website, store that data with append in correspond list. If the data is empty or not found, the correspond list would append None as value.

Ztore 士多
web crawler



Import Libraries

```
1   import pandas as pd
2   from selenium import webdriver
3   from selenium.webdriver.common.by import By
4   from selenium.webdriver.chrome.options import Options
5   import time
6   import datetime
```

Def Crawler

```
8   def crawler(url):
9       driver = webdriver.Chrome()
10      driver.get(url)
11      time.sleep(3)
```

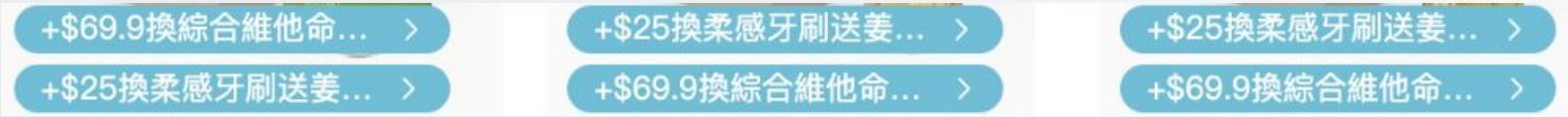
Click the show all button and scroll down the web

```python
    try:
        send_button = driver.find_element(By.XPATH, '//*[@id="BaseLayout"]/div/div[4]/span/div/span')
        send_button.click()
    except:
        pass

    time.sleep(3)

    while len(data) < number:
        data = driver.find_elements(By.XPATH, '//div[contains(@class, "jsx-2940394913 jsx-676457135 ProductItem  windowing-layout")]')
        driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
        time.sleep(3)
```

Get Item Data

```python
    for index, item in enumerate(data, start=1):
        try:
            # item.find_element(By.CLASS_NAME, 'brand-product-name')found something
            title_list.append(item.find_element(By.CLASS_NAME, 'name_bbd').text)
        except:
            # item.find_element(By.CLASS_NAME, 'brand-product-name') is not found
            title_list.append(None)
        try:
            packsize_list.append(item.find_element(By.CLASS_NAME, 'packsize').text)
        except:
            packsize_list.append(None)
        try:
            OP_list.append(item.find_element(By.CLASS_NAME, 'price-container').text.split('\n')[0].split('$')[1])
        except:
            OP_list.append(None)
        try:
            DP_list.append(item.find_element(By.CLASS_NAME, 'price-container').text.split('\n')[-2].split('$')[1])
        except:
            DP_list.append(None)
```

Get Promotion Data



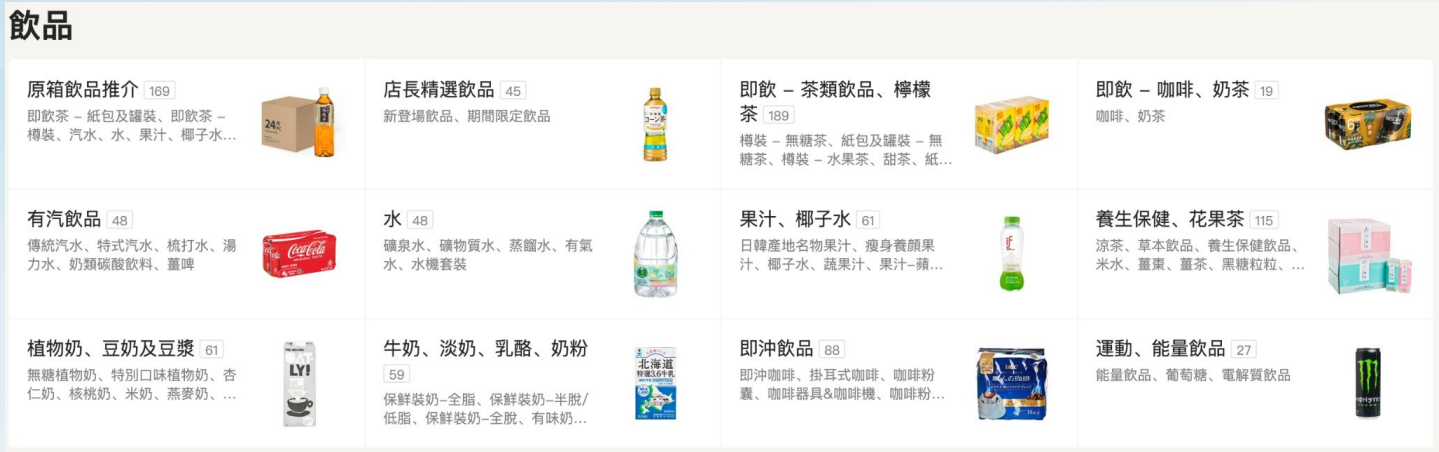

```
58          try:
59              promotions = item.find_elements(By.XPATH, f"//*[@id='BaseLayout']/div/div[4]/div[2]/div/div/div[{index}]/div[3]/div/div")
60              promotions = promotions[0].text.split("\n")
61
62              try:
63                  Promotion1_list.append(promotions[0])
64              except:
65                  Promotion1_list.append(None)
66
67              try:
68                  Promotion2_list.append(promotions[1])
69              except:
70                  Promotion2_list.append(None)
71          except:
72              Promotion1_list.append(None)
73              Promotion2_list.append(None)
74
75          try:
76              link_list.append(item.find_element(By.XPATH, f'//*[@id="BaseLayout"]/div/div[4]/div[2]/div/div/div[{index}]/a').get_attribute("href"))
77          except:
78              link_list.append(None)
```

Dataframe Design

```
81      df = pd.DataFrame(
82          {
83          'Product_Title' : title_list,
84          'Package_Size' : packsize_list,
85          'Original_Price' : OP_list,
86          'Discount_Price' : DP_list,
87          'Promotion_1' : Promotion1_list,
88          'Promotion_2' : Promotion2_list,
89          'Link' : link_list
90          }
91          )
92
93      df['Discount_Price'] = df['Discount_Price'].replace(',', '', regex=True).astype(float)
94      df['Original_Price'] = df['Original_Price'].replace(',', '', regex=True).astype(float)
95
96      driver.quit()
97
98      return df
```

Dictionary URL and Categories

```python
ztore_drinks_url = {
    "Beverage Case Deals" : "https://www.ztore.com/tc/category/all/beverage/case-offers",
    "Recommended Drinks" : "https://www.ztore.com/tc/category/all/beverage/recommended-drinks",
    "RTD - Tea & Lemon Tea" : "https://www.ztore.com/tc/category/all/beverage/rtd--tea-lemon-tea",
    "RTD - Coffee & Milk Tea" : "https://www.ztore.com/tc/category/all/beverage/rtd-coffee-milk-tea",
    "Carbonated Drinks" : "https://www.ztore.com/tc/category/all/beverage/carbonated-beverage",
    "Water" : "https://www.ztore.com/tc/category/all/beverage/water",
    "Juice and Coconut Water" : "https://www.ztore.com/tc/category/all/beverage/juice-and-energy-drink",
    "Healthy Drinks" : "https://www.ztore.com/tc/category/all/beverage/healthy-drinks",
    "Plant Based & Soy Milk" : "https://www.ztore.com/tc/category/all/beverage/soy-plant-based-milk",
    "Long Life Milk, Yogurt & Milk Powder" : "https://www.ztore.com/tc/category/all/beverage/long-life-milk-soy",
    "Hot Beverage" : "https://www.ztore.com/tc/category/all/beverage/coffee-tea",
    "Energy & Sports Drink" : "https://www.ztore.com/tc/category/all/beverage/energy-drink"
}
```



Print csv

```
117  # Create current date (datetime obj), and change it into string.
118  current_time_str = datetime.datetime.now().strftime("%Y_%m_%d")
119
120  for k, v in ztore_drinks_url.items():
121      df = crawler(v)
122      # Use assign to create a new column "date" with the same value (current_datetime)
123      df = df.assign(date = current_time_str)
124
125      # Create a new column "retailer" with retailer name "Ztore"
126      df = df.assign(retailer = "ztore")
127
128      # Create a new column "category" with the pns_drinks_url key
129      df = df.assign(category = k)
130
131      df.to_csv(k + "_" + current_time_str + "_ztore.csv", encoding="UTF-8")
```

PARKnSHOP 百佳
web crawler

There are 11 categories of beverages

PARKnSHOP 百佳 web crawler

Page layout

- Contains many products in the category
- When scrolldown, page will shows more products, until no more product

```python
import pandas as pd
from selenium import webdriver # use webdriver
from selenium.webdriver.common.by import By # different methods of locating data
from selenium.webdriver.chrome.options import Options # options for selenium driver
import time
import datetime
```

Import the libraries which required

```python
pns_drinks_url = {
    "Alcoholic Beverages"             : "https://www.pns.hk/zh-hk
    "Water"                           : "https://www.pns.hk/zh-hk
    "Carbonated Drink"                : "https://www.pns.hk/zh-hk
    "Instant Tea & Coffee & Milk Tea" : "https://www.pns.hk/zh-hk
    "Milk & Yogurt"                   : "https://www.pns.hk/zh-hk
    "Plant Based & Soy Milk"          : "https://www.pns.hk/zh-hk
    "Coffee, Hot Drink & Mix Powder"  : "https://www.pns.hk/zh-hk
    "Juice & Coconut Water"           : "https://www.pns.hk/zh-hk
    "Energy Drink"                    : "https://www.pns.hk/zh-hk
    "Herbal & Healthy Drink"          : "https://www.pns.hk/zh-hk
    "Case Offer"                      : "https://www.pns.hk/zh-hk
}
```

Create dictionary contains categories and URLs

```python
import pandas as pd
from selenium import webdriver # use webdriver
from selenium.webdriver.common.by import By # different methods of locating data
from selenium.webdriver.chrome.options import Options # options for selenium driver
import time
import datetime


def crawler(url):···


pns_drinks_url = {
    "Alcoholic Beverages"             : "https://www.pns.hk/zh-hk/%E9%A3%9F%E5%93%81%E5%8F%8A%E9%A3%B2%E5%93%81/%E9%85%92%E7%B2%BE%E9%A3
    "Water"                           : "https://www.pns.hk/zh-hk/%E9%A3%9F%E5%93%81%E5%8F%8A%E9%A3%B2%E5%93%81/%E6%B0%B4/c/04010100",
    "Carbonated Drink"                : "https://www.pns.hk/zh-hk/%E9%A3%9F%E5%93%81%E5%8F%8A%E9%A3%B2%E5%93%81/%E6%B1%BD%E6%B0%B4/c/040
    "Instant Tea & Coffee & Milk Tea" : "https://www.pns.hk/zh-hk/%E9%A3%9F%E5%93%81%E5%8F%8A%E9%A3%B2%E5%93%81/%E5%8D%B3%E9%A3%B2%E8%8C
    "Milk & Yogurt"                   : "https://www.pns.hk/zh-hk/%E9%A3%9F%E5%93%81%E5%8F%8A%E9%A3%B2%E5%93%81/%E5%A5%B6%E9%A1%9E%E3%80
    "Plant Based & Soy Milk"          : "https://www.pns.hk/zh-hk/%E9%A3%9F%E5%93%81%E5%8F%8A%E9%A3%B2%E5%93%81/%E6%A4%8D%E7%89%A9%E5%A5
    "Coffee, Hot Drink & Mix Powder"  : "https://www.pns.hk/zh-hk/%E9%A3%9F%E5%93%81%E5%8F%8A%E9%A3%B2%E5%93%81/%E5%92%96%E5%95%A1%E3%80
    "Juice & Coconut Water"           : "https://www.pns.hk/zh-hk/%E9%A3%9F%E5%93%81%E5%8F%8A%E9%A3%B2%E5%93%81/%E6%9E%9C%E6%B1%81%E3%80
    "Energy Drink"                    : "https://www.pns.hk/zh-hk/%E9%A3%9F%E5%93%81%E5%8F%8A%E9%A3%B2%E5%93%81/%E9%81%8B%E5%8B%95%E5%8F
    "Herbal & Healthy Drink"          : "https://www.pns.hk/zh-hk/%E9%A3%9F%E5%93%81%E5%8F%8A%E9%A3%B2%E5%93%81/%E8%8D%89%E6%9C%AC%E5%8F
    "Case Offer"                      : "https://www.pns.hk/zh-hk/%E9%A3%9F%E5%93%81%E5%8F%8A%E9%A3%B2%E5%93%81/%E5%8E%9F%E7%AE%B1%E9%A3
}

# create current date (datetime obj), and change it into string
current_time_str = datetime.datetime.now().strftime("%Y_%m_%d")

for k, v in pns_drinks_url.items():
    df = crawler(v)
    # use assign to create a new column "date" with the same value (current_datetime)
    df = df.assign(date = current_time_str)

    # create a new column "retailer" with retailer name "parknshop"
    df = df.assign(retailer = "parknshop")

    # create a new column "category" with the pns_drinks_url key
    df = df.assign(category = k)

    # save as csv file
    df.to_csv("pns_data/" + k + "_" + current_time_str + "_pns.csv", encoding="UTF-8")
```

Dictionary contains categories and URLs

Loop through all URLs

Use crawler function to get data
input ：url
return ：dataframe

Insert today’s date and category into the columns

Save as csv files named with category and date

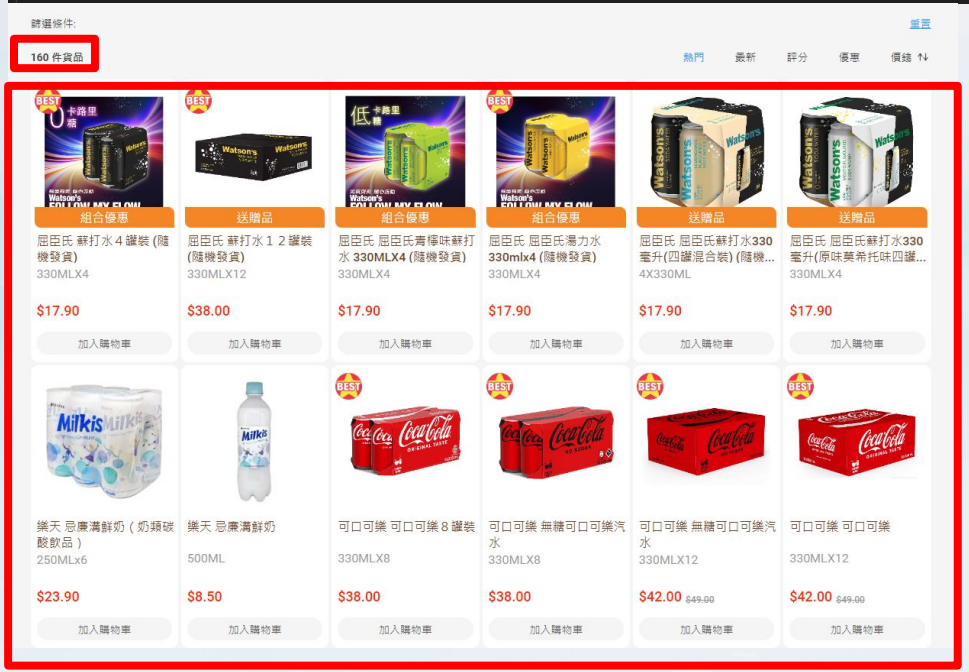
```python
def crawler(url):

    driver = webdriver.Chrome() # create instance of selenium driver

    try:
        driver.get(url)
    except:
        print("url not found")
        exit

    time.sleep(2)

    remark = []
    name = []
    capacity = []
    selling_price = []
    original_price = []
    more_info = []
    link = []


    result_dict = {
        "remark" : remark,
        "name" : name,
        "capacity" : capacity,
        "selling_price" : selling_price,
        "original_price" : original_price,
        "more_info" : more_info,
        "link" : link
    }
```

Define crawler function

Try to the web page
exit when url not found

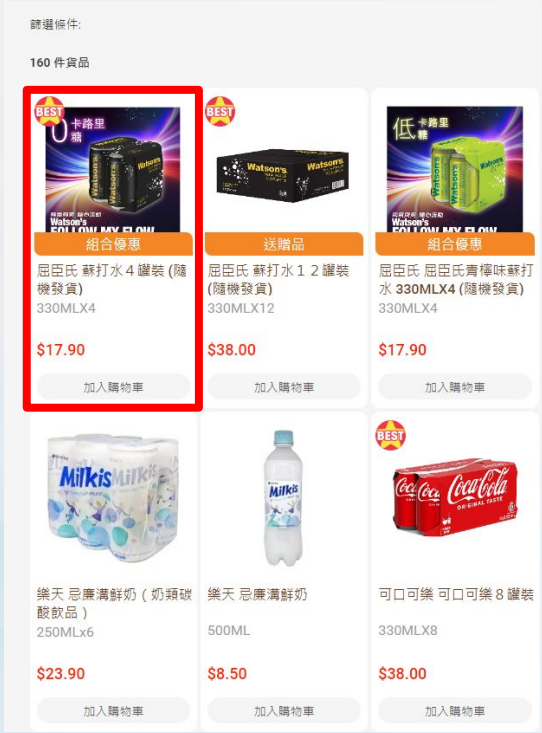
Create empty lists and dictionary
for the date we will get

```python
try:
    total_no_of_product = driver.find_element(By.XPATH, "/html/body/app-root/cx-storefront/main/cx-page-layout/cx-page-slot//pns-product-list/div[1]/div[1]")
    total_no_of_product = int(total_no_of_product.text.split(" ")[0])
    print(total_no_of_product)

    products = driver.find_elements(By.XPATH, "//html/body/app-root/cx-storefront/main/cx-page-layout/cx-page-slot//pns-product-list/div[2]/div/pns-product-tile")
    print(len(products))

    while len(products) < total_no_of_product:
        driver.execute_script("window.scrollTo(0, document.body.scrollHeight)")
        time.sleep(2)
        products = driver.find_elements(By.XPATH, "//html/body/app-root/cx-storefront/main/cx-page-layout/cx-page-slot//pns-product-list/div[2]/div/pns-product-tile")
        print(len(products))
```



Find the number of products scrolldown web page to show more products

```
for product in products:

    product_info = product.text.split("\n")
    if len(product_info) == 4:
        remark.append(None)
        name.append(product_info[0])
        capacity.append(product_info[1])
        selling_price.append(product_info[2])
        original_price.append(None)
        more_info.append(product_info[-1])

    elif len(product_info) == 5:
        if "$" in product_info[-2] and "$" in product_info[-3]:
            remark.append(None)
            name.append(product_info[0])
            capacity.append(product_info[1])
            selling_price.append(product_info[2])
            original_price.append(product_info[3])
        else:
            remark.append(product_info[0])
            name.append(product_info[1])
            capacity.append(product_info[2])
            selling_price.append(product_info[3])
            original_price.append(None)

        more_info.append(product_info[-1])

    elif len(product_info) == 6:
        remark.append(product_info[0])
        name.append(product_info[1])
        capacity.append(product_info[2])
        selling_price.append(product_info[3])
        original_price.append(product_info[4])
        more_info.append(product_info[-1])
```



Get data and append to the correspond lists

```
    product_link = driver.find_elements(By.XPATH, "/html/body/app-root/cx-storefront/main/cx-page-layout/cx-page-slot//pns-product-list/div[2]/div/pns-product-tile//div/div[1]/div[2]/a")
    for product in product_link:
        link.append(product.get_attribute('href'))

except Exception as error:
    print(f'Error - {error} = {url} ')
```

```
driver.quit()

df = pd.DataFrame(result_dict)

df['capacity'] = df['capacity'].str.upper()     # change capacity column to uppercase

df.index.name = 'sequence'      # the index column of df will be named 'sequence'

return df
```

```
# create current date (datetime obj), and change it into string
current_time_str = datetime.datetime.now().strftime("%Y_%m_%d")

for k, v in pns_drinks_url.items():
    df = crawler(v)
    # use assign to create a new column "date" with the same value (current_datetime)
    df = df.assign(date = current_time_str)

    # create a new column "retailer" with retailer name "parknshop"
    df = df.assign(retailer = "parknshop")

    # create a new column "category" with the pns_drinks_url key
    df = df.assign(category = k)

    # save as csv file
    df.to_csv("pns_data/" + k + "_" + current_time_str + "_pns.csv", encoding="UTF-8")
```

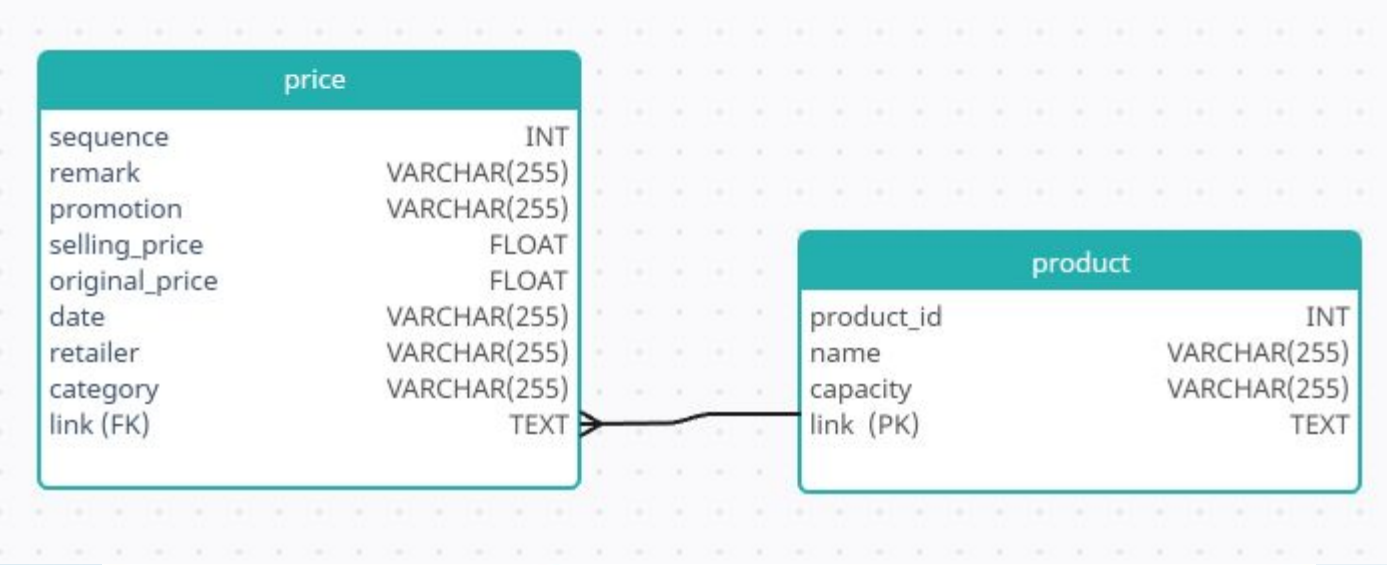
The function returns dataframe

Add current date, retailer name and category into the dataframe

Save as csv file

Database

Entity Relationship Diagram and Schemas



Create two tables
Decrease data redundancy

Create table and insert values

```
import psycopg2
import pandas as pd
import os
```

Import the libraries required

```
# Create a connection to the database
conn = psycopg2.connect(
    host = "...",
    dbname = "...",
    user = "...",
    password = "...",
    port = ...
)
```

Connect to the database such as PostgreSQL

Create table

```
cur.execute(
    """
    DROP TABLE IF EXISTS product CASCADE;
    CREATE TABLE product (
        product_id INT,
        name VARCHAR(255),
        capacity VARCHAR(255),
        link TEXT PRIMARY KEY
    );

    DROP TABLE IF EXISTS price CASCADE;
    CREATE TABLE price (
        sequence INT,
        remark VARCHAR(255),
        promotion VARCHAR(255),
        selling_price FLOAT,
        original_price FLOAT,
        date VARCHAR(255),
        retailer VARCHAR(255),
        category VARCHAR(255),
        link TEXT,
        FOREIGN KEY (link) REFERENCES product(link)
    );

    """)
```

Use SQL query in python to create tables in database

Insert values

```
dataFolder = "ztore_data/"
files = os.listdir(dataFolder)
```

Get files from folder

```
masterdf = pd.DataFrame()

for file in files:
    if file.endswith(".csv"):
        df = pd.read_csv(dataFolder + file)
        df.rename(columns={"Unnamed: 0":'sequence',
                           "Link" : "link",
                           "Product_Title" : "name",
                           "Package_Size" : "capacity",
                           "Original_Price" : "original_price",
                           "Discount_Price" : "selling_price",
                           "Promotion_1" : "remark",
                           "Promotion_2" : "promotion"
                          }, inplace=True)

        masterdf = pd.concat((masterdf, df), ignore_index = True)
```

Use Python logical conditions to read only csv files

Insert values

```
# Specify the columns you want to insert
columns_to_insert_product = [
        "product_id",
        "name" ,
        "capacity" ,
        "link"
    ]

columns_to_insert_price = [
        "sequence" ,
        "remark" ,
        "promotion" ,
        "selling_price" ,
        "original_price" ,
        "date" ,
        "retailer" ,
        "category" ,
        "link"
    ]
```

Insert values

```python
# Create a cursor object
cur = conn.cursor()

# Insert into the table

for index, row in masterdf.iterrows():
    # Construct the INSERT INTO query
    query = f"INSERT INTO product ({', '.join(columns_to_insert_product)}) VALUES ({', '.join(['%s'] * len(columns_to_insert_product))}) ON CONFLICT (link) DO NOTHING"

    # Prepare the values
    values = [row[column] for column in columns_to_insert_product]

    # Execute the INSERT INTO query
    cur.execute(query, values)


    query = f"INSERT INTO price ({', '.join(columns_to_insert_price)}) VALUES ({', '.join(['%s'] * len(columns_to_insert_price))})"
    values = [row[column] for column in columns_to_insert_price]
    cur.execute(query, values)
```

```python
# Commit the changes
conn.commit()

# Close the connection
conn.close()
```

Use SQL query in python to insert data into database

Price table schema

Query | Query History

```
1  select * from price;
```

Data Output | Messages | Notifications

	sequence integer	remark character varying (255)	promotion character varying (255)	selling_price double precision	original_price double precision	date character varying (255)	retailer character varying (255)	category character varying (255)	link text
1	0	多重優惠	[null]	38	NaN	2023_11_17	wellcome	[null]	https://www.wellcome
2	1	指定品牌送贈品	[null]	30	NaN	2023_11_17	wellcome	[null]	https://www.wellcome
3	2	2件$50	[null]	33	NaN	2023_11_17	wellcome	[null]	https://www.wellcome
4	3	多重優惠	[null]	38	NaN	2023_11_17	wellcome	[null]	https://www.wellcome
5	4	指定品牌送贈品	[null]	26	NaN	2023_11_17	wellcome	[null]	https://www.wellcome
6	5	NaN	[null]	13	NaN	2023_11_17	wellcome	[null]	https://www.wellcome
7	6	多重優惠	[null]	38	NaN	2023_11_17	wellcome	[null]	https://www.wellcome
8	7	多重優惠	[null]	18	NaN	2023_11_17	wellcome	[null]	https://www.wellcome
9	8	NaN	[null]	13	NaN	2023_11_17	wellcome	[null]	https://www.wellcome
10	9	指定分類送贈品	[null]	13	NaN	2023_11_17	wellcome	[null]	https://www.wellcome
11	10	NaN	[null]	12.9	NaN	2023_11_17	wellcome	[null]	https://www.wellcome
12	11	多重優惠	[null]	38	NaN	2023_11_17	wellcome	[null]	https://www.wellcome
13	12	NaN	[null]	13	NaN	2023_11_17	wellcome	[null]	https://www.wellcome
14	13	NaN	[null]	24	29	2023_11_17	wellcome	[null]	https://www.wellcome
15	14	指定品牌送贈品	[null]	26	NaN	2023_11_17	wellcome	[null]	https://www.wellcome
16	15	2件$41	[null]	23	NaN	2023_11_17	wellcome	[null]	https://www.wellcome
17	16	2件$32	[null]	20	NaN	2023_11_17	wellcome	[null]	https://www.wellcome

Product table schema

Query | Query History

```
1  select * from product;
```

Data Output | Messages | Notifications

	product_id integer	name character varying (255)	capacity character varying (255)	link [PK] text
1	[null]	可口可樂汽水罐裝 8 X 330ML	[null]	https://www.wellcome.com.hk/zh-hant/p/%E5%8F%AF%E5%
2	[null]	維他 低糖氣泡菊花茶 6 X 310ML	[null]	https://www.wellcome.com.hk/zh-hant/p/%E7%B6%AD%E4%
3	[null]	明治頂級鮮牛奶 946ML	[null]	https://www.wellcome.com.hk/zh-hant/p/%E6%98%8E%E6%
4	[null]	可口可樂無糖汽水罐裝 8 X 330ML	[null]	https://www.wellcome.com.hk/zh-hant/p/%E5%8F%AF%E5%
5	[null]	維他檸檬茶 9 X 250ML	[null]	https://www.wellcome.com.hk/zh-hant/p/%E7%B6%AD%E4%
6	[null]	維他冷泡無糖香片茶 6 X 250ML	[null]	https://www.wellcome.com.hk/zh-hant/p/%E7%B6%AD%E4%
7	[null]	可口可樂加系汽水罐裝 8 X 330ML	[null]	https://www.wellcome.com.hk/zh-hant/p/%E5%8F%AF%E5%
8	[null]	維他檸檬茶 6 X 250ML	[null]	https://www.wellcome.com.hk/zh-hant/p/%E7%B6%AD%E4%
9	[null]	維他冷泡無糖凍頂烏龍茶 6 X 250ML	[null]	https://www.wellcome.com.hk/zh-hant/p/%E7%B6%AD%E4%
10	[null]	北海道3.6牛乳迷你裝 200ML	[null]	https://www.wellcome.com.hk/zh-hant/p/%E5%8C%97%E6%
11	[null]	維他蘋果綠茶 6 X 250ML	[null]	https://www.wellcome.com.hk/zh-hant/p/%E7%B6%AD%E4%
12	[null]	玉泉罐裝梳打水 8 X 330ML	[null]	https://www.wellcome.com.hk/zh-hant/p/%E7%8E%89%E6%
13	[null]	維他冷泡無糖鐵觀音茶 6 X 250ML	[null]	https://www.wellcome.com.hk/zh-hant/p/%E7%B6%AD%E4%
14	[null]	Meadows全脂鮮牛奶 1LT	[null]	https://www.wellcome.com.hk/zh-hant/p/Meadows%E5%85
15	[null]	維他低糖檸檬茶 9 X 250ML	[null]	https://www.wellcome.com.hk/zh-hant/p/%E7%B6%AD%E4%
16	[null]	屈臣氏蒸餾水 4.5LT	[null]	https://www.wellcome.com.hk/zh-hant/p/%E5%B1%88%E8%
17	[null]	MEADOWS 梳打水 6 X 320ML	[null]	https://www.wellcome.com.hk/zh-hant/p/MEADOWS%20%E
18	[null]	維他高鈣朱古力奶 4 X 125ML	[null]	https://www.wellcome.com.hk/zh-hant/p/%E7%B6%AD%E4%
19	[null]	維他冷泡無糖鎮檸檬茶 6 X 250ML	[null]	https://www.wellcome.com.hk/zh-hant/p/%E7%B6%AD%E4%

product

sequence	INT
remark	VARCHAR(255)
promotion	VARCHAR(255)
selling_price	FLOAT
original_price	FLOAT
date	VARCHAR(255)
retailer	VARCHAR(255)
category	VARCHAR(255)
link (FK)	TEXT

Join two tables for analyzing data

Query | Query History

```
1  select * from product
2  join price
3  using (link);
```

Scratch Pad

Data Output | Messages | Notifications

	link text	product_id integer	name character varying (2	capacity character varying	sequence integer	remark character vary	promotion character varyin	selling_price double precision	original_price double precisior	date character var	retailer character varyi	category character varying (255)
1	https://www.pn...	327901	葡萄適 葡萄適 S...	750ML	22	$24.9 / 2件	[null]	14.9	NaN	2023_11_...	parknshop	Energy Drink
2	https://www.pn...	45321	好茶養生 桂花雪...	EACH	49	NaN	[null]	15	NaN	2023_11_...	parknshop	Coffee, Hot Drink & Mix ...
3	https://www.pn...	45323	好茶養生 寧神安...	EACH	50	NaN	[null]	12	NaN	2023_11_...	parknshop	Coffee, Hot Drink & Mix ...
4	https://www.pn...	45325	好茶養生 桂圓紅...	EACH	51	NaN	[null]	12	NaN	2023_11_...	parknshop	Coffee, Hot Drink & Mix ...
5	https://www.pn...	45326	好茶養生 清熱降...	EACH	52	NaN	[null]	12	NaN	2023_11_...	parknshop	Coffee, Hot Drink & Mix ...
6	https://www.pn...	45327	好茶養生 茉莉玫...	EACH	53	NaN	[null]	12	NaN	2023_11_...	parknshop	Coffee, Hot Drink & Mix ...
7	https://www.pn...	45329	好茶養生 皇牌老...	EACH	54	NaN	[null]	180	NaN	2023_11_...	parknshop	Coffee, Hot Drink & Mix ...
8	https://www.pn...	45330	好茶養生 黑糖薑...	EACH	55	NaN	[null]	180	NaN	2023_11_...	parknshop	Coffee, Hot Drink & Mix ...
9	https://www.pn...	45335	好茶養生 枸杞阿...	EACH	56	NaN	[null]	168	NaN	2023_11_...	parknshop	Coffee, Hot Drink & Mix ...
10	https://www.pn...	45337	好茶養生 黑穀十...	EACH	57	NaN	[null]	168	NaN	2023_11_...	parknshop	Coffee, Hot Drink & Mix ...
11	https://www.pn...	45352	好茶養生 仙楂烏...	EACH	58	NaN	[null]	50	NaN	2023_11_...	parknshop	Coffee, Hot Drink & Mix ...
12	https://www.pn...	45354	好茶養生 玫瑰四...	EACH	59	NaN	[null]	50	NaN	2023_11_...	parknshop	Coffee, Hot Drink & Mix ...
13	https://www.pn...	45362	好茶養生 蜜糖菊...	EACH	60	NaN	[null]	50	NaN	2023_11_...	parknshop	Coffee, Hot Drink & Mix ...
14	https://www.pn...	45381	好茶養生 蜂蜜檸...	EACH	61	NaN	[null]	80	NaN	2023_11_...	parknshop	Coffee, Hot Drink & Mix ...
15	https://www.pn...	48648	好茶養生 石斛花...	EACH	62	NaN	[null]	15	NaN	2023_11_...	parknshop	Coffee, Hot Drink & Mix ...

Analyzing data

Example : When the product release

```
import pandas as pd
import psycopg2
import matplotlib.pyplot as plt
from matplotlib.font_manager import FontProperties
```

```
# Establish a connection to the database
conn = psycopg2.connect(···

# SQL query
sql_query = """
    SELECT * FROM price
    JOIN product
    USING (link)
    where name like 'BOSS%';
"""

# Use pandas to execute the SQL query and store the result in a DataFrame
df = pd.read_sql_query(sql_query, conn)

# close the connection
conn.close()
```

Using matplotlib to create diagram

Analyzing data

example：to show when the product release

```python
# Specify a font that supports Chinese characters
font = FontProperties(fname=r"font.ttf")  # the path to your font file

df['date'] = pd.to_datetime(df['date'], format="%Y_%m_%d")  # Ensure the date column is in datetime format

# Sort the DataFrame based on date
df.sort_values('date', inplace=True)

# Create a new column 'available' where each product is 1 if available and 0 if not
df['available'] = df['name'].notna().astype(int)

start_date = '2023-11-01'
end_date = '2023-11-18'

# Create a date range that includes all dates you're interested in
all_dates = pd.date_range(start=start_date, end=end_date)

# Loop over each product
for product in df['name'].unique():
    # Filter data for the current product
    product_data = df[df['name'] == product]

    # Reindex the DataFrame to include all dates and fill missing availability with 0
    product_data.set_index('date', inplace=True)
    product_data = product_data.reindex(all_dates, fill_value=0)

    # Reset the index so 'date' is a column again
    product_data.reset_index(inplace=True)
    product_data.rename(columns={'index': 'date'}, inplace=True)

    # Plot the data
    plt.plot(product_data['date'], product_data['available'], label=product)
```

using matplotlib to create diagram

Analyzing data

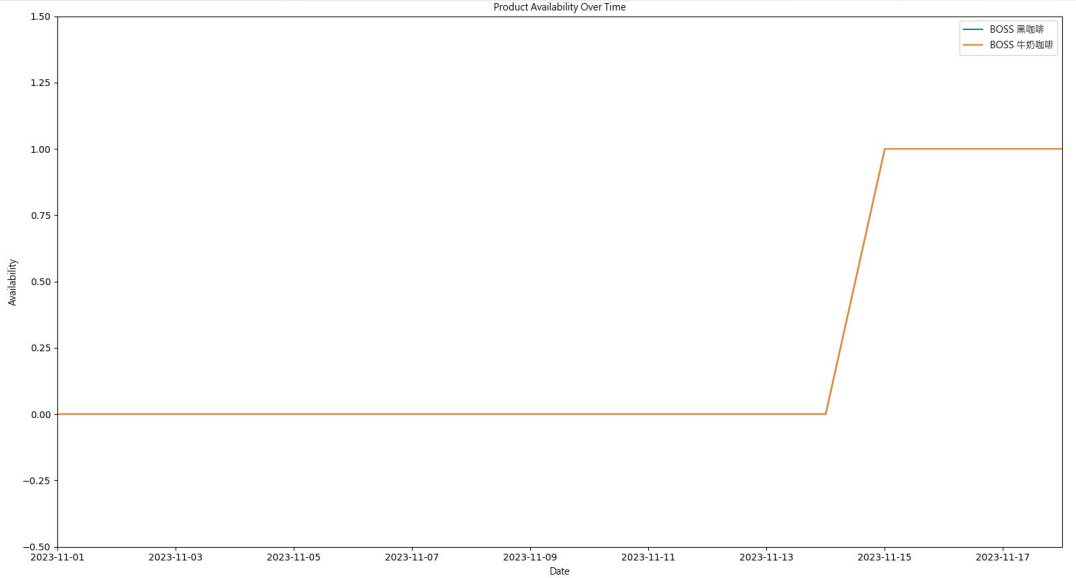
Example：When the product release

Using matplotlib to create diagram

```
# Set the range of the x-axis
plt.xlim(pd.to_datetime(start_date, format="%Y-%m-%d"), pd.to_datetime(end_date, format="%Y-%m-%d"))

# Set the range of the y-axis
plt.ylim(-0.5, 1.5)


plt.xlabel('Date', fontproperties=font)
plt.ylabel('Availability', fontproperties=font)
plt.title('Product Availability Over Time', fontproperties=font)
plt.legend(prop=font)
plt.show()
```

Analyzing data

Example ：When the product release output：



Result：
The brand BOSS 黑咖啡，牛奶咖啡 released on 16 Nov 2023

Analyzing data

Example ：When the product release

Query　Query History

```
1  select category, count(*) as no_of_product from price
2  where date = '2023_11_17'
3  and retailer = 'parknshop'
4  group by category;
```

Data Output　Messages　Graph Visualiser ×　Notifications

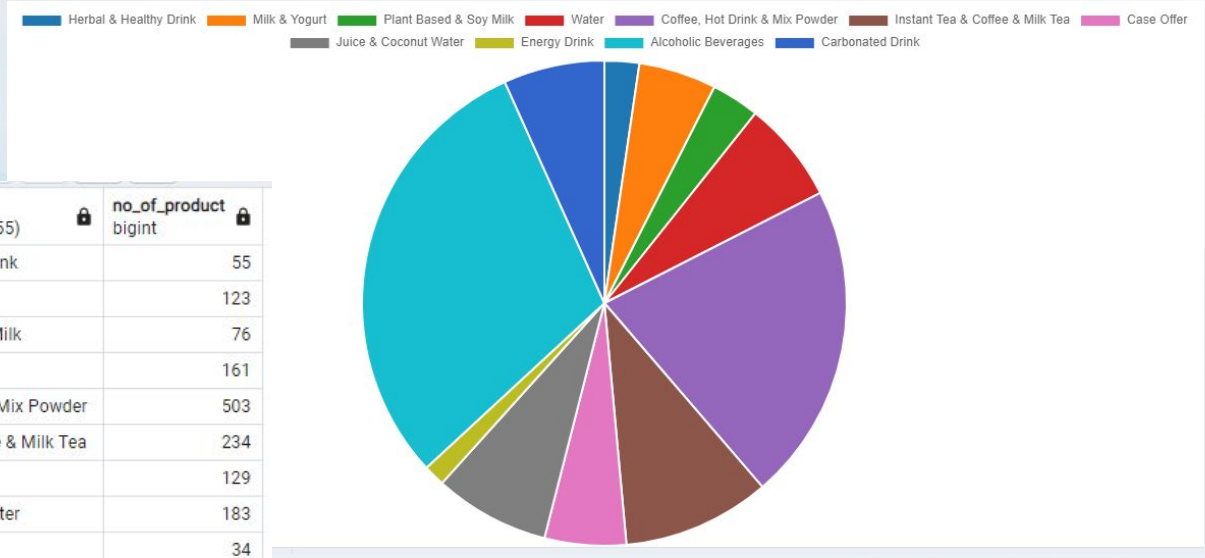
Graph Type　Pie Chart

Label　category

Value　no_of_product

	category character varying (255)	no_of_product bigint
1	Herbal & Healthy Drink	55
2	Milk & Yogurt	123
3	Plant Based & Soy Milk	76
4	Water	161
5	Coffee, Hot Drink & Mix Powder	503
6	Instant Tea & Coffee & Milk Tea	234
7	Case Offer	129
8	Juice & Coconut Water	183
9	Energy Drink	34
10	Alcoholic Beverages	715
11	Carbonated Drink	160

Using pgAdmin 4 Graph Viualiser



Analyzing data

average price of sugar-free drinks in last ten days

Query | Query History

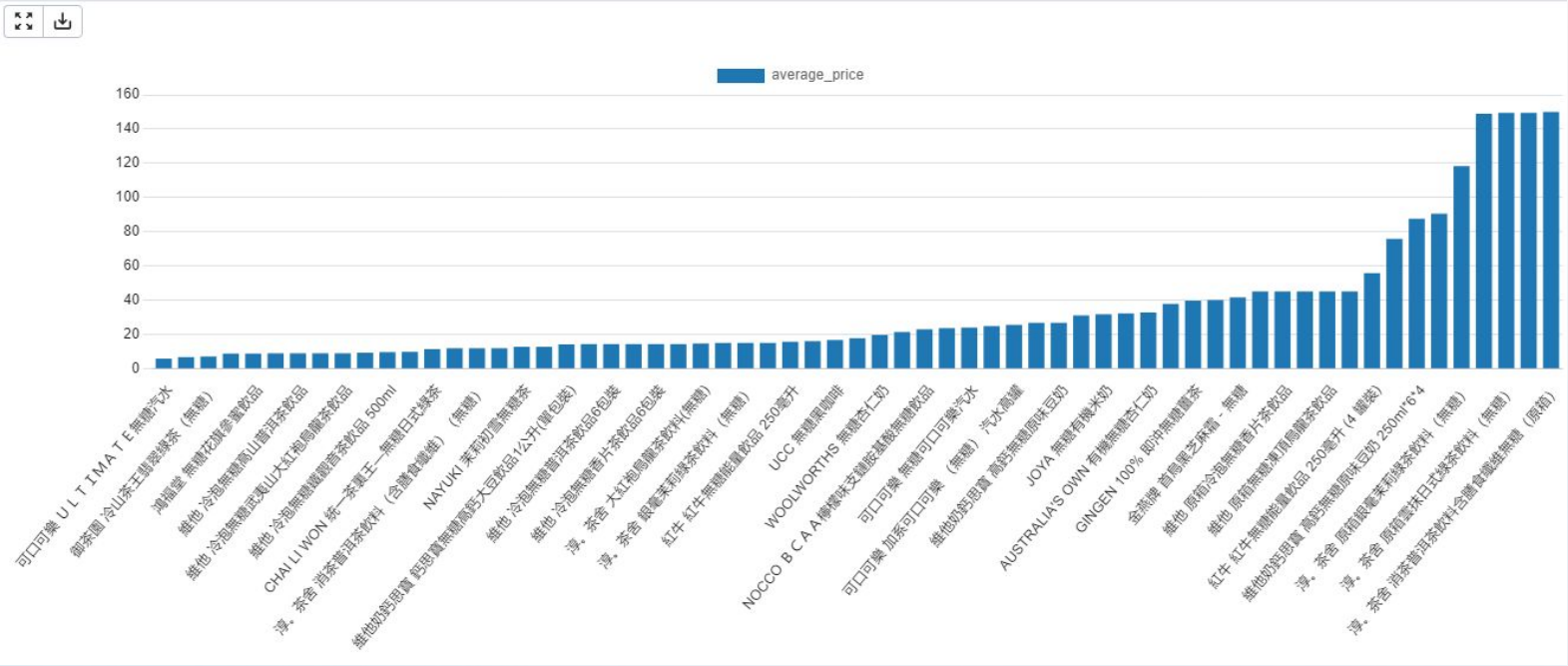
```
1  select name, avg(selling_price) as average_price from price
2  join product
3  using (link)
4  where name like '%無糖%'
5  and retailer = 'parknshop'
6  group by name
7  order by average_price;
```

	name character varying (255)	average_price double precision
1	可口可樂 ＵＬＴＩＭＡＴＥ無糖汽水	6
2	ITAMACHI 麥茶（無糖）	6.900000000000001
3	御茶園 冷山茶王翡翠綠茶（無糖）	7.316666666666667
4	鴻福堂 羅漢果(無糖)飲品	8.900000000000002
5	鴻福堂 無糖花旗參蜜飲品	8.900000000000002
6	維他 冷泡無糖觀音茶飲品 500ml	9.081818181818182
7	維他 冷泡無糖高山普洱茶飲品	9.15
8	維他 維他冷泡無糖香片茶飲品500ml	9.15
9	維他 冷泡無糖武夷山大紅袍烏龍茶飲品	9.15
10	淳。茶會 消煎茶飲料（含膳食纖維） 無糖	9.441666666666668
11	維他 冷泡無糖纖觀音茶飲品 500ml	9.9
12	可口可樂 可口可樂加系可口可樂（無糖）	9.916666666666666
13	CHAI LI WON 統一茶裏王－無糖日式綠茶	11.56666666666667
14	淳。茶會 裏抹TM日式綠茶飲料(無糖)	12.095833333333331
15	淳。茶會 消茶普洱茶飲料（含膳食纖維） （無糖）	12.095833333333331
16	淳。茶會 銀電茉莉綠茶飲料(無糖)	12.095833333333331
17	NAYUKI 茉莉初雪無糖茶	12.9
18	NAYUKI 祁門紅茶無糖茶	12.9
19	維他奶鈣思寶 鈣思寶無糖高鈣大豆飲品1公升(單包裝)	14.25
20	維他 冷泡無糖凍頂烏龍茶飲品	14.5
21	維他 冷泡無糖普洱茶飲品6包裝	14.5
22	維他 冷泡無糖錫蘭檸檬茶飲品	14.5
23	維他 冷泡無糖香片茶飲品6包裝	14.5
24	維他 冷泡無糖纖觀音飲品	14.5
25	淳。茶會 大紅袍烏龍茶飲料(無糖)	14.75
26	淳。茶會 淳。茶會 大紅袍烏龍茶-無糖	15.166666666666666
27	淳。茶會 銀電茉莉綠茶飲料（無糖）	15.166666666666666
28	道地 無糖烏龍茶1.5升裝	15.258333333333335
29	紅牛 紅牛無糖能量飲品 250毫升	15.900000000000004
30	百事 可樂無糖 4 罐裝	16.233333333333338
31	UCC 無糖黑咖啡	16.900000000000002
32	UCC 職人無糖咖啡	18

Total rows: 63 of 63 | Query complete 00:00:00.122

Analyzing data

average price of sugar-free drinks in last ten days



Challenge

Future Vision

- Enhanced Data Crawling
  - Refine the data crawler
  - Incorporate machine learning
  - Explore additional data sources
- Advanced Analytics
  - Develop advanced analytics
  - Implement predictive analytics
- User-Friendly Interface
  - Technical and less technical users
  - Drag-and-drop functionality, customizable dashboards
- Collaboration and Reporting:
  - Allow users share insights, reports, and findings
  - Automated report to PDF/PowerPoint/email.
- Integration with External Tools:
  - Tableau or Power BI more visualizations and storytelling.
  - Enable API share to other analytics tools.
- Continuous Feedback and Improvement:
  - Regularly collect feedback.
  - Conduct user testing sessions.

Thank You

Q&A Session