쇼핑몰 크롤링 DB에 저장

- 데이터베이스 생성

cmd(명령프롬프트) / MySQL Workbench에서 "beauty_shop" 이라는 데이터베이스를 만들어봅시다.

- 크롤링 해오는 상품 이름 중에 이모지가 포함되어 있는 경우가 있어서 이모지를 제거하는 모듈을 설치해줍니다.
- MySQL의 기본 UTF-8(utf8)은 3바이트까지만 지원하는데 이모지는4 바이트의 문자

In [1]:
```
!pip install emoji
```

```
Requirement already satisfied: emoji in c:\programdata\anaconda3\lib\site-packages (2.15.0)
```

In [2]:
```
from urllib.request import urlopen
from bs4 import BeautifulSoup
import pandas as pd
import numpy as np
import pymysql
import emoji
```

모든 제품을 크롤링해서 DB에 저장

- 저장할 DB부터 테이블 생성해서 형식에 맞게 data 변환 후 DB로 insert

1. DB 연결 코드
2. 필요한 DB 및 table 생성
3. 크롤링
    - 데이터를 table 형식에 맞게 정제
4. insert (데이터 생성)
5. commit() 해서 db에 반영
6. db 닫기

- db 테이블을 읽어와서 df 에 저장

1. DB연결 및 필요 객체 생성

In [3]:
```
def conn(d_name) :
    import pymysql
    host_name = 'localhost'
    host_port = 3306
    username = 'root'
    password = '0000'
    database_name = d_name

    db = pymysql.connect(
    host=host_name, # MySQL Server Address
    port=host_port, # MySQL Server Port
    user=username, # MySQL username
    password=password, # password for MySQL username
    database=database_name, # Database name
    charset='utf8'
    )
    return db
```

In [4]:
```
db = conn('beauty_shop')
cursor = db.cursor()
```

In [5]:
```
sql = "SHOW DATABASES"
cursor.execute(sql)
result = cursor.fetchall()
result
```

Out[5]:
```
(('beauty_shop',),
 ('ecommerce',),
 ('employees',),
 ('information_schema',),
 ('mysql',),
 ('performance_schema',),
 ('sakila',),
 ('sqldb',),
 ('student_mgmt',),
 ('sys',),
 ('world',))
```

In [6]:
```
sql = "USE beauty_shop"
cursor.execute(sql)
```

Out[6]:
```
0
```

In [7]:
```
# 현재 세션이 어떤 데이터베이스를 사용하고 있는 중인지 출력하는 쿼리
sql = "SELECT DATABASE()"
cursor.execute(sql)
result = cursor.fetchone()
result
```

Out[7]:
```
('beauty_shop',)
```

In [8]:
```
# 테이블 생성
sql = '''
CREATE TABLE IF NOT EXISTS product (
PRODUCT_CODE int AUTO_INCREMENT NOT NULL,
TITLE VARCHAR(200) NOT NULL,
ORI_PRICE VARCHAR(100),
DISCOUNT_PRICE VARCHAR(100),
link VARCHAR(200),
PRIMARY KEY(PRODUCT_CODE)
);
'''

cursor.execute(sql)
db.commit()
```

In [9]:
```
sql = "SHOW TABLES"
cursor.execute(sql)
result = cursor.fetchall()
result
```

Out[9]:
```
(('product',),)
```

In [10]:
```
sql = "DESC product"
cursor.execute(sql)
result = cursor.fetchall()
result
```

Out[10]:
```
(('PRODUCT_CODE', 'int', 'NO', 'PRI', None, 'auto_increment'),
 ('TITLE', 'varchar(200)', 'NO', '', None, ''),
 ('ORI_PRICE', 'varchar(100)', 'YES', '', None, ''),
 ('DISCOUNT_PRICE', 'varchar(100)', 'YES', '', None, ''),
 ('link', 'varchar(200)', 'YES', '', None, ''))
```

2. 크롤링 코드 - insert 구문을 추가해서 변경

In [11]:
```
# 크롤링 문서 요청해서 응답객체 반환(첫번째 크롤링 문서 요청 후 응답)
url="http://jolse.com/category/toners-mists/1019/"
html = urlopen(url)
htmls = html.read()
bs_obj = BeautifulSoup(htmls,"html.parser")
```

In [12]:
```
# box안에 들어 있는 1개의 상품에서 정보를 추출해서 dict형태로  반환하는 함수
# 데이터 전처리 : 제품명에 ' 제거/ 가격 USD 제거 / 세일가격 없는 경우 처리
def get_product_info(box) :
    strong_tag = box.find("strong",{"class":"name"})
    # 품목 추출
    span =strong_tag.text.split(':',1)[1]

    # 세부 페이지 링크 추출
    a = strong_tag.find("a")
    sub_link = 'https://jolse.com' + a["href"]
    sub_link = emoji.replace_emoji(sub_link,'')
```

```
    # 가격 추출 코드
    price_ul = box.find("ul")

    old_price = price_ul.select("li")[0]
    old_price = old_price.select("span")[1].text
    old_price = old_price.split(" ")[1]

    dis_price = price_ul.select("li")[1]
    dis_price = dis_price.select("span")[1].text
    dis_price = dis_price.split(" ")[1]

    # 데이터 전처리
    title = span.replace("'","''") # ' 처리
    title = emoji.replace_emoji(title,'')

    # 세일 가격이 없는 경우
    if dis_price =='' :
        dis_price = '0.0'

    # 최종 data 추출 후 반환
    return{"prd_name":title,"price":old_price,"sale_price":dis_price,"sub_link":sub_link}
```

In [13]:

```
def save_data(prd_info) :
    # print(prd_info)

    # INSERT 구문
    sql = "INSERT INTO product (title, ori_price, discount_price,link) values('" \
        + prd_info["prd_name"] \
        +"'," \
        + prd_info['price']\
        +"," \
        + prd_info['sale_price']\
        +",'"\
        + prd_info['sub_link']\
        + "');"
    # 완성된 쿼리문 확인하려면 주석을 해제
    # print(sql)
    cursor.execute(sql)
```

In [14]:
```
# 전달된 url 페이지에 접근해서 해당페지의 전체 상품 데이터를 추출 한 후 각 상품마다 get_product_info()함수를 호출해서
# 각 상품에대한 추출 정보를 받아옴 - 들어온 각 상품 정보를 리스트에 저장 한 수 해당 반환
def get_page_products(url) :
    url=url
    html = urlopen(url)
    htmls = html.read()
    bs_obj = BeautifulSoup(htmls,"html.parser")

    # 한 페이지에 모든 상품이 들어있는 ul 태그 추출
    # ul class:prdList grid4
    ul=bs_obj.find("ul",{"class":"prdList grid5"})

    # 품목 1개를 담고 있는 div 태그 추출
    # div class:box
    prd_boxes = ul.findAll("div", {"class":"description"}) #1개 페이지의 전체 상품

    # 반환되는 품목 데이터를 db에 insert : 함수호출해서 진행
    for box in prd_boxes :
        prd = get_product_info(box)

        # 제품 상세정보를 잘 가져오는지 확인하려면 print의 주석을 해제
        # print(prd)

        # 위에서 정의한 데이터 저장하는 쿼리문을 실행하는 함수
        save_data(prd)
```

main 코드(프로그램 시작점)

In [15]:
```
from tqdm import tqdm # 상태바 표시

# 여러 페이지의 화장품 정보를 추출해서 df 에 저장 후 csv에 저장하는 코드
url = "http://jolse.com/category/toners-mists/1019/?page="

# last = int(bs_obj.find("a",{"class":"last"})['href'].split("=")[1])
# 마지막 페이지를 가져옴 (현재 사이트 상황으로 13)
# 지금은 빠른 실습을 위해 2페이지가 마지막이라고 가정하고 진행
last = 2

for i in tqdm(range(1,last+1)):
```

```
    urlfin =url + str(i)
    get_page_products(urlfin)
```

```
100%|██████████| 2/2 [00:02<00:00,  1.21s/it]
```

In [16]:
```
db.commit()
```

In [17]:
```
sql = "SELECT * FROM product"
cursor.execute(sql)
result = cursor.fetchall()
result[:5]
```

Out[17]:
```
((1,
  ' B : Lab Essence Toner Pad 10ea/25g',
  '5.00',
  '2.25',
  'https://jolse.com/product/b-lab-essence-toner-pad-10ea25g/89277/category/1019/display/1/'),
 (2,
  ' EQQUALBERRY Swimming Pool Toner 300ml',
  '22.00',
  '15.40',
  'https://jolse.com/product/eqqualberry-swimming-pool-toner-300ml/87123/category/1019/display/1/'),
 (3,
  ' celimax Jiwoogae Heartleaf BHA Peeling Pad 60pcs (22AD)',
  '17.50',
  '13.12',
  'https://jolse.com/product/celimax-jiwoogae-heartleaf-bha-peeling-pad-60pcs-22ad/55940/category/1019/display/1/'),
 (4,
  ' SKIN1004 Madagascar Centella Probio-Cica Essense Toner 210ml',
  '24.00',
  '14.49',
  'https://jolse.com/product/skin1004-madagascar-centella-probio-cica-essense-toner-210ml/68504/category/1019/display/1/'),
 (5,
  ' ROUND LAB 1025 Dokdo Toner 200ml',
  '17.00',
  '13.60',
  'https://jolse.com/product/round-lab-1025-dokdo-toner-200ml/18903/category/1019/display/1/'))
```

In [18]:
```
db.close()
```

db 테이블에 저장된 데이터 df로 가져오기

In [19]:
```
db = conn("beauty_shop")
```

In [20]:
```
sql = "SELECT * FROM product"
df = pd.read_sql(sql,db)
df
```

```
C:\Users\User\AppData\Local\Temp\ipykernel_14748\923367605.py:2: UserWarning: pandas only supports SQLAlchemy connectable (engi
ne/connection) or database string URI or sqlite3 DBAPI2 connection. Other DBAPI2 objects are not tested. Please consider using
SQLAlchemy.
  df = pd.read_sql(sql,db)
```

Out[20]:

	PRODUCT_CODE	TITLE	ORI_PRICE	DISCOUNT_PRICE	link
0	1	B : Lab Essence Toner Pad 10ea/25g	5.00	2.25	https://jolse.com/product/b-lab-essence-toner-...
1	2	EQQUALBERRY Swimming Pool Toner 300ml	22.00	15.40	https://jolse.com/product/eqqualberry-swimming...
2	3	celimax Jiwoogae Heartleaf BHA Peeling Pad 60...	17.50	13.12	https://jolse.com/product/celimax-jiwoogae-hea...
3	4	SKIN1004 Madagascar Centella Probio-Cica Esse...	24.00	14.49	https://jolse.com/product/skin1004-madagascar-...
4	5	ROUND LAB 1025 Dokdo Toner 200ml	17.00	13.60	https://jolse.com/product/round-lab-1025-dokdo...
...	...	...	...	...	...
75	76	mixsoon Centella Asiatica Toner 300ml	25.00	16.25	https://jolse.com/product/mixsoon-centella-asi...
76	77	APLB Glutathione Niacinamide Mist Essence 105ml	16.40	10.17	https://jolse.com/product/aplb-glutathione-nia...
77	78	WellDerma PDRN Exosome Hydrating Gel Mist 100ml	42.00	16.80	https://jolse.com/product/wellderma-pdrn-exoso...
78	79	One-day's you Handy Help Me Real Collagen Pad...	4.00	3.00	https://jolse.com/product/one-days-you-handy-h...
79	80	Isntree Hyaluronic Acid Toner Plus 200ml	22.20	17.76	https://jolse.com/product/isntree-hyaluronic-a...

80 rows × 5 columns

In [21]:
```
sql = "SELECT title,ori_price,discount_price FROM product"
df = pd.read_sql(sql,db)
df
```

```
C:\Users\User\AppData\Local\Temp\ipykernel_14748\1628218276.py:2: UserWarning: pandas only supports SQLAlchemy connectable (eng
ine/connection) or database string URI or sqlite3 DBAPI2 connection. Other DBAPI2 objects are not tested. Please consider using
SQLAlchemy.
  df = pd.read_sql(sql,db)
```

Out[21]:

	title	ori_price	discount_price
0	B : Lab Essence Toner Pad 10ea/25g	5.00	2.25
1	EQQUALBERRY Swimming Pool Toner 300ml	22.00	15.40
2	celimax Jiwoogae Heartleaf BHA Peeling Pad 60...	17.50	13.12
3	SKIN1004 Madagascar Centella Probio-Cica Esse...	24.00	14.49
4	ROUND LAB 1025 Dokdo Toner 200ml	17.00	13.60
...	...	...	...
75	mixsoon Centella Asiatica Toner 300ml	25.00	16.25
76	APLB Glutathione Niacinamide Mist Essence 105ml	16.40	10.17
77	WellDerma PDRN Exosome Hydrating Gel Mist 100ml	42.00	16.80
78	One-day's you Handy Help Me Real Collagen Pad...	4.00	3.00
79	Isntree Hyaluronic Acid Toner Plus 200ml	22.20	17.76

80 rows × 3 columns

In [22]:
```
db.close()
```

In [ ]: