데이터 사이언스를 위한

파이썬 프로그래밍

2020.06.25



산업시스템공학과

2017112526

김민지

목차

**주제 선정3**

세부주제 선정3

**데이터 선정3**

웹페이지 선정3

데이터 수집 시 이용 패키지3

**프로젝트 범위 설정3**

데이터 범위 및 한계3

주요 사용기술4

예상 데이터베이스 형태4

**입력 데이터 처리4**

칼럼별 데이터 수집 방법4

데이터베이스 구축 및 테이블 생성6

**출력 데이터 처리 및 최종결과물7**

출력 데이터 형식7

출력 데이터 항목8

**결론15**

1. 주제선정
2. 세부 주제 선정

주제1에서 제시한 내용은 매일매일 바뀌는 데이터를 가진 웹페이지를 크롤링하여, 데이터를 얻고 해당 데이터는 Maria DB에 저장하며, 데이터베이스에서 해당 자료를 받아 시계열 데이터를 형성하는 것을 최종 목표로 하는 것이다.

따라서, 본 프로젝트에서 매일 바뀌는 날씨데이터를 이용한 시계열 데이터를 형성하고, 사용자가 원하는 데이터를 제공하고자 한다.

1. 데이터 선정
2. 웹페이지 선정

[[1]](#footnote-1)웹페이지는 네이버 날씨 사이트로 선정하였다. 해당 페이지를 선정한 이유로는 네이버 웹 사이트는 대중적인 웹사이트로 접근이 용이하였고, 원하는 지역의 원하는 동까지 선정할 가능해 원하는 정보를 추출할 수 있기 때문이다.

1. 데이터 수집 시 이용 패키지

웹페이지 크롤링을 위한 BeautifulSoup와 Urlopen 그리고 데이터를 수집하고 Maria DB에 저장할 수 있도록, pymysql 패키지를 이용하였다.

1. 프로젝트 범위 설정
2. 데이터 범위 및 한계

데이터는 매일 오후 2시에 크롤링을 통해 수집한다.

날씨 정보는 현재 날씨(맑음, 흐림, 비), 현재 기온(ex. 24), 미세먼지 정도(나쁨/보통/좋음), 오존 정도(나쁨/보통/좋음)으로 설정하였으며, 지역은 ‘서울특별시 중구 장충동2가’, ‘충청북도 제천시 고암동’, ‘대전광역시 대덕구 송촌동’, ‘대구광역시 중구 동성로 1가’, ‘부산광역시 중구 남포동’으로 총 5곳을 선정하였다.

본 프로젝트의 한계는 정해진 시간에 크롤링을 하지 않으면 시간적인 일관성이 결여되어 의미 있는 데이터 값이 도출할 수 없다는 것이다.

1. 주요 사용기술

○ 데이터 수집 및 저장

- BeautifulSoup, Urlopen 패키지를 이용하여 웹사이트를 열고 웹사이트에 있는 자료를 크롤링하였다. 주로 Url을 연 후, find\_All() 및 find() 함수를 이용하여 원하는 정보가 있는 태그에 도달하였으며, 원하는 최종 데이터 수집을 위하여 .text, get\_attr() 함수를 이용하였다. 크롤링 한 데이터를 pymysql 패키지를 이용하여 Maria DB에 저장한다.

○ 데이터프레임 및 데이터시리즈 다루기

- Pandas 패키지를 이용하여 데이터를 다루었으며, 주로 상황에 맞게 index를 변경하고 원하는 정보를 추출하기 위하여 index 값을 이용하여 추출하는 loc[] 함수를 이용한다. 또한, 주로 날짜별 혹은 지역별로 groupby() 함수를 이용해 같은 group에 있는 것들끼리 평균, 최대, 최소 값을 계산하는 작업을 수행한다

○ 데이터 시각화

- Matplotlib.pyplot, Matplotlib 패키지를 이용하여, 그래프를 이용해 데이터를 시각화 한다. 주로, 시계열 그래프를 사용하였고 bar 형태의 그래프도 그린다. 지역 요소별로 그래프의 색깔, 점의 색깔을 다르게 설정하였다. 한글 Legend 및 X축과 Y축, Title을 사용하기 위해 rc() 함수를 이용한다.

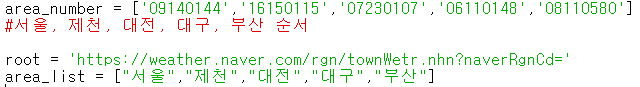
1. 예상 데이터 프레임 형태

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 지역 | 날짜 | 현재 날씨 | 현재 온도 | 미세먼지 정도 | 오존 정도 |
| 서울 | 20200603 | 맑음 | 28 | 보통 | 보통 |
| 제천 | 20200603 | 흐림 | 24 | 좋음 | 보통 |
| 대전 | 20200603 | 흐림 | 24 | 보통 | 보통 |
| 대구 | 20200603 | 맑음 | 29 | 나쁨 | 나쁨 |
| 부산 | 20200603 | 흐림 | 26 | 좋음 | 나쁨 |

칼럼은 지역, 날짜, 현재 날씨, 현재 온도, 미세먼지 정보, 오존 정도로 설정하며, 출력데이터의 특징에 따라 인덱스를 변경하며 데이터를 추출한다.

1. 입력 데이터 처리
2. 칼럼별 데이터 수집 방법

○ URL 접근, 지역









[그림 1. URL 접근을 위한 리스트 생성 및 지역 데이터 저장]

- 지역별로 URL을 보았을 때, 앞부분의 경우 동일하지만, 뒷부분에 지역코드가 지역 및 동네별로 다르다. 따라서 root 변수에 url의 동일한 부분을 저장하고, area\_number 리스트를 만들어 지역코드를 리스트 형태로 저장해주어, for문을 이용해 각 지역별 날씨 URL에 접근하여 데이터를 수집하였다.

- area\_number(지역 코드가 저장된 리스트)에 있는 순서에 기반으로 area\_list = ["서울","제천","대전","대구","부산"] 지역리스트를 제작하여 데이터를 넣어주었다.

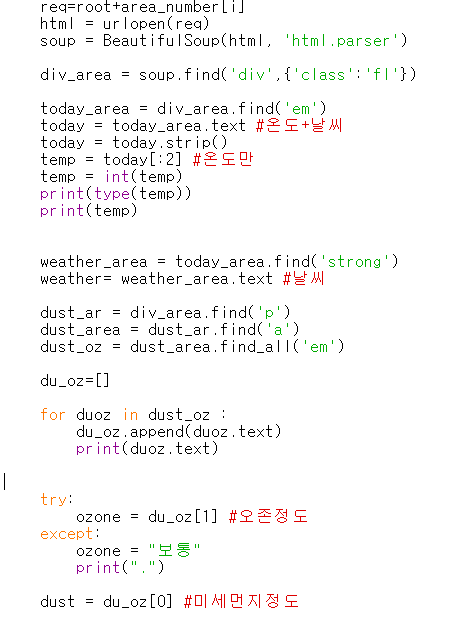
○ 날짜



[그림 2. 당일의 날짜 데이터를 얻기 위한 코드]

- datetime 패키지에 있는 today() 함수를 이용하여, 예를 들어 2020년 6월 20일의 데이터는 ‘20200620’ 이라고 데이터가 저장되도록 포맷을 만들어주었다.

○ 날씨, 온도, 미세먼지 정도, 오존 정도



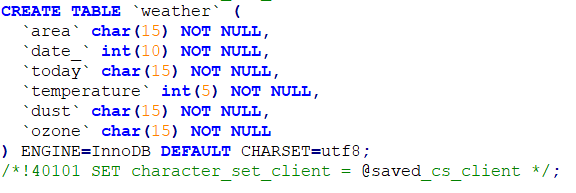
[그림 3. 크롤링 코드]

- BeautifulSoup 패키지를 이용하여 웹에 있는 데이터를 수집하였다. 주로 find\_All(), find와 같이 원하는 태그에 접근하여 태그의 정보를 가져오고, 안에 있는 텍스트를 저장하였다.

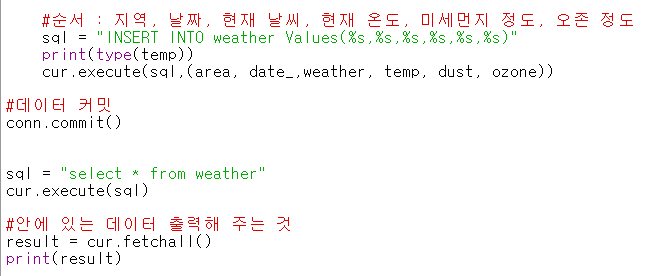
- 단, 오존정도는 안 나올 때가 존재한다. 정확한 이유가 있는데 안 나오는 것이 아니라 아이에 웹사이트에 안 올라와 있어 종종 오류가 뜬다. 그래서 그럴 때에는 try문을 이용하여, 만약 에러가 떴을 때 except로 이동하여, “보통”을 넣도록 하였다. 그리고 print로 “.”을 출력하고 직접 사이트에 들어가서 확인해본다. 하지만 확인을 하러 가도 웹사이트에 아이에 공시가 안 되어있었다.

1. 데이터베이스 구축 및 테이블 생성

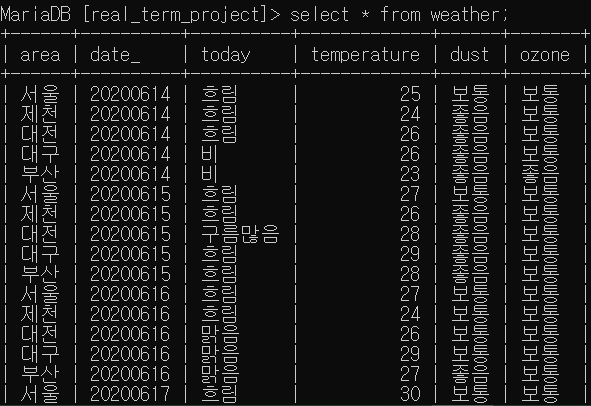
Real\_term\_project라는 데이터베이스를 생성하였고, weather이라는 테이블을 만들어 해당 데이터들을 저장하였다.



[그림 4. 데이터베이스에서 Table weather 정의]



[그림 5. python에서 Maria DB로 데이터 전송 코드]



[그림 6. Maria DB에 저장된 Table weather]

1. 출력 데이터 처리 및 최종 결과물
2. 출력 데이터 형식

출력 데이터는 메뉴형식으로 1번부터 8번까지 중 원하는 항목을 선택하는 형식이다. While(True)를 이용해 계속해서 메뉴를 보여주고 메뉴를 선택하게 한다. 1~7번까지는 데이터 출력 항목이고 8번은 프로그램 종료를 의미한다.

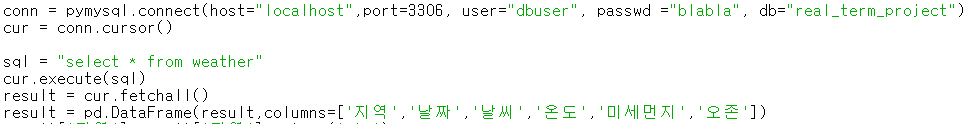


[그림 7. 메뉴형식 코드]

1. 출력 데이터 항목

○ 데이터베이스에서 테이블을 가져오기

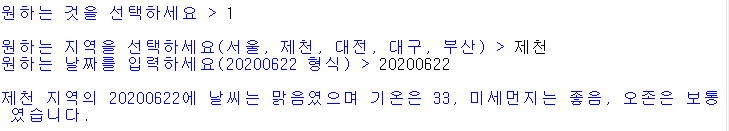
- 데이터베이스에 있는 데이터를 테이블로 가져오고 가져온 데이터들을 데이터 프레임 형태로 구성한다. 그리고 다음 데이터프레임을 result라고 정의한다.



[그림 8. 데이터베이스로 데이터 가져오기]

○ 1번 : 원하는 지역, 날짜를 입력하고 전체 날씨 정보 얻기

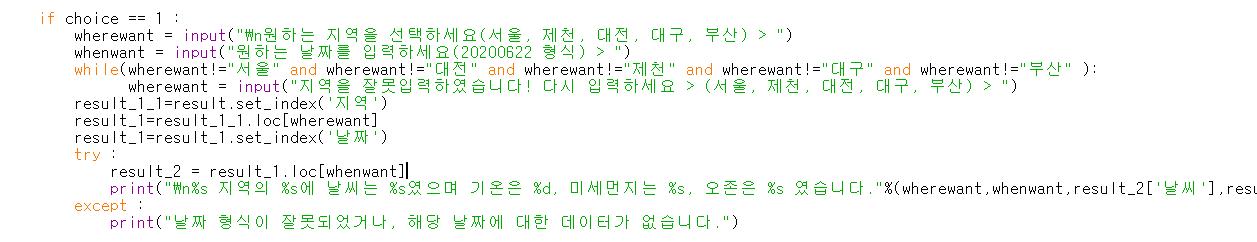
- 입력 데이터 : 지역, 날짜



[그림 9. 1번 항목 입력/출력]

- 사용자가 입력한 지역이 [서울, 제천, 대전, 대구, 부산]이 아닐 시에는 while문을 이용하여 다시 지역을 입력하도록 한다.

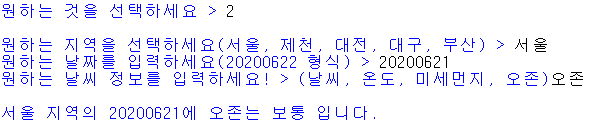
- 사용자가 원하는 지역을 입력 받아, wherewant에 저장시키고, 사용자가 원하는 날짜를 whenwant변수에 저장해 result의 index를 ‘지역’으로 변경해 새로운 데이터프레임을 만들어준다. index를 이용해 데이터를 찾는 loc 함수를 이용해 사용자가 원한 지역의 데이터를 추출하고, index를 ‘날짜’ 재설정 해 새로운 데이터프레임 result\_1로 바꿔준 후, 원하는 날짜데이터를 loc함수를 이용해 최종 추출해 정보를 제공한다.



[그림 10. 1번 항목 코드]

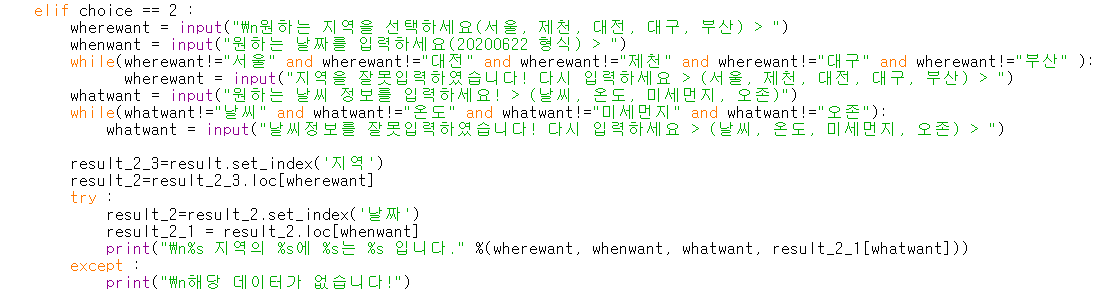
○ 2번 : 원하는 지역, 날짜를 입력하고 전체 날씨 정보 얻기

- 입력 데이터 : 지역, 날짜, 날씨정보



[그림 11. 2번 항목 입력/출력]

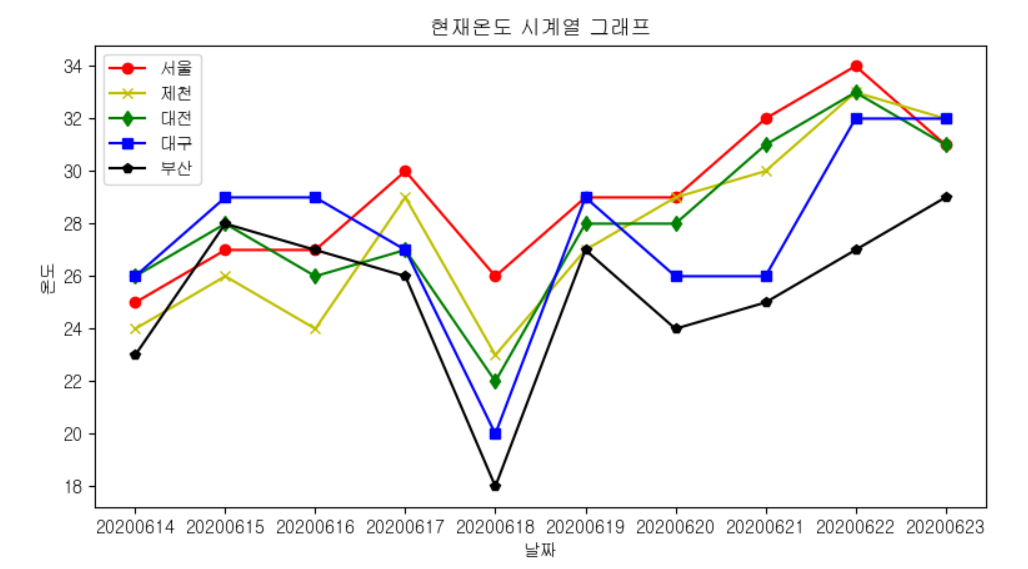
- 1번항목에서 날씨 정보를 추가한 항목이다. 인덱스를 지역, 날짜순으로 변경해주며 1번과 동일하게 해당 데이터들을 추출한 후, 원하는 날씨 정보를 whatwant변수에 입력받고, 해당 변수를 이용해 칼럼에 접근해 최종 데이터를 추출한다.



[그림 12. 2번 항목 코드]

○ 3번 : 전체 지역의 시계열 현재기온 변화 그래프 보기

- 입력데이터 : 없음

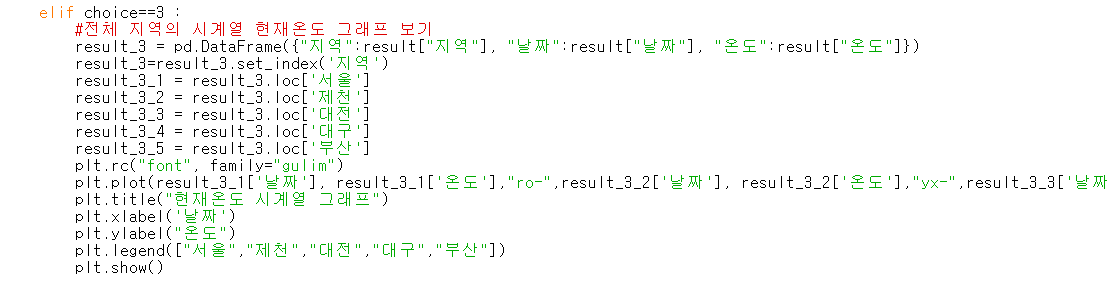


[그림 13. 3번 항목 시계열 그래프 출력]

- result 데이터프레임에서 ‘지역’, ’날짜’, ‘온도’ 칼럼만 존재하는 result\_3 데이터프레임을 만들어준다. Index는 ‘지역’으로 설정하여, loc 함수를 이용해 [서울, 제천, 대전, 대구, 부산] 지역의 데이터들을 각각 result\_3\_1, result\_3\_2, result\_3\_3, result\_3\_4, result\_3\_5에 저장해주고, X축에는 날짜가 되도록 그리고 Y축은 현재기온이 되도록 설정한다.

- X축 라벨 ,Y축 라벨, 타이틀에 한글을 사용하기 위해, rc() 함수를 이용하여 한글 데이터가 깨지지 않도록 설정하고, 범례는 지역명으로 지정한다.

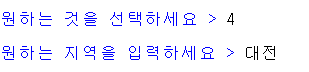
- 지역별로 색깔과 점 모양을 다르게 설정해준다.



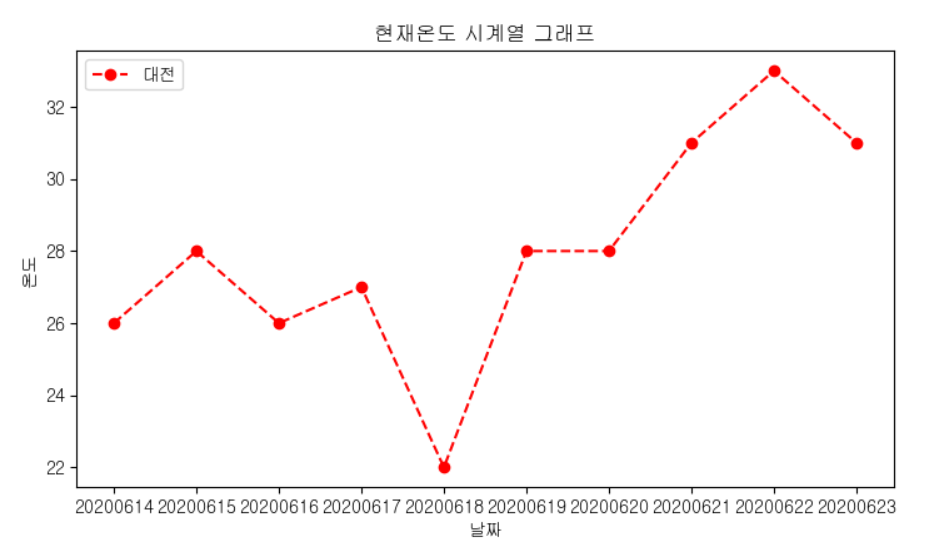
[그림 14. 3번 항목 코드]

○ 4번 : 원하는 지역의 시계열 현재기온 변화 그래프 보기

- 입력데이터 : 지역



[그림 15. 4번 항목 입력데이터]

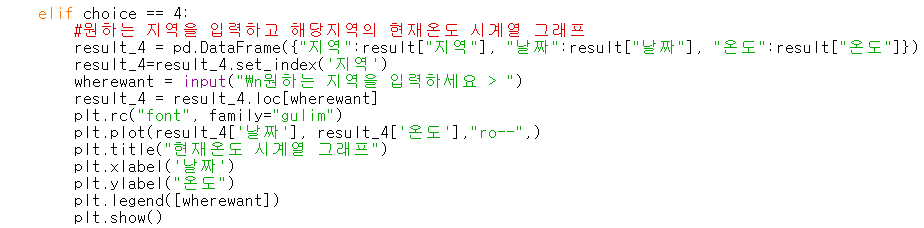


[그림 16. 4번 항목 출력데이터]

- result 데이터프레임에서 ‘지역’, ’날짜’, ‘온도’ 칼럼만 존재하는 result\_4 데이터프레임을 만들어준다. Index는 ‘지역’으로 설정한다. 사용자가 원하는 지역명을 wherewant 변수에 저장한 후, loc 함수를 이용하여 해당 지역데이터들을 추출해준다. X축에는 날짜가 되도록 그리고 Y축은 현재기온이 되도록 설정한다.

- X축 라벨 ,Y축 라벨, 타이틀에 한글을 사용하기 위해, rc() 함수를 이용하여 한글 데이터가 깨지지 않도록 설정하고, 범례는 지역명으로 지정한다.

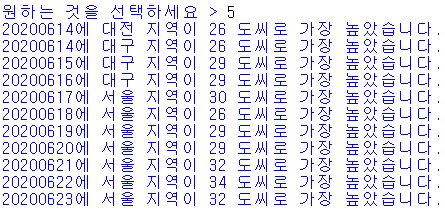
- 그래프는 빨간색이고 점 모양은 동그라미 모양 그리고 점선형태로 만든다.



[그림 17. 4번 항목 코드]

○ 5번 : 날짜 별, 현재 기온이 가장 높은 지역과 기온 정보 얻기

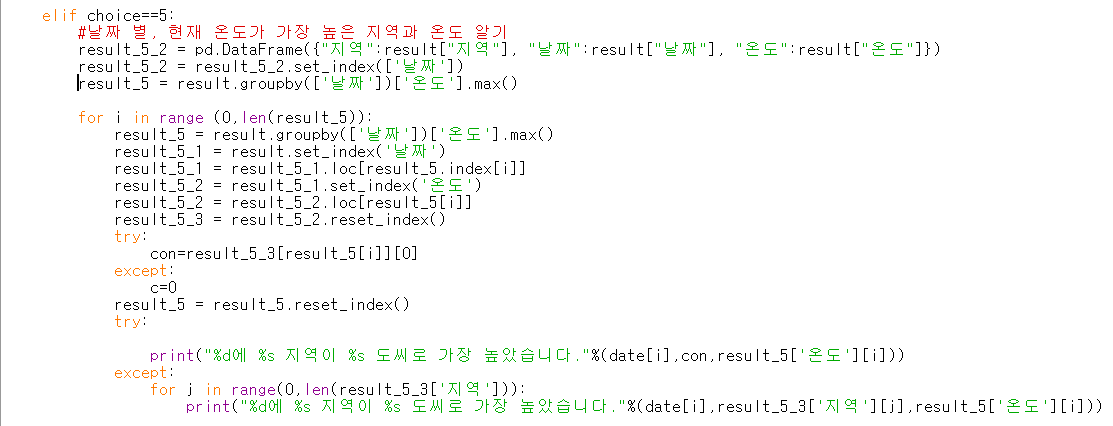
- 입력데이터 : 없음



[그림 18. 5번 항목 출력]

- result에서 ‘지역’, ’날짜’, ‘온도’ 칼럼만 포함하는 result\_5\_2 데이터 프레임을 만들어 주고 해당 데이터프레임의 index는 ‘날짜’로 지정한다. Result5라는 이름으로 result 데이터프레임에 ‘날짜’를 기준으로 groupby 해주고 ‘온도’ 칼럼의 max값을 저장한다. ‘날짜’, ‘온도’ 순서로 칼럼을 변경하며, 해당 날짜의 해당 최고온도에 부합한 것을 찾는다.

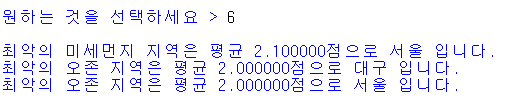
- 해당날짜에 최고 기온이 두 지역 이상인 것은 데이터프레임의 형태로, 한 곳인 것은 데이터시리즈로 나오는 것이 확인되어, try문과 except문을 이용해 출력을 다르게 구분해주었다.



[그림 19. 5번 항목 코드]

○ 6번 : 과거부터 현재까지 미세먼지/오존 정도가 가장 나쁜 지역 정보 얻기

- 입력데이터 : 없음



[그림 20. 6번 항목 출력데이터]

- 미세먼지/오존 정도는 나쁨/보통/좋음 3가지 형태로 나오는데 그것을 각각 1/2/3점으로 부여해준다.

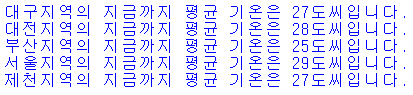
- result와 동일한 result\_6 데이터프레임의 ‘미세먼지’ 칼럼 값에 따라 점수를 부여해 result\_6에 ‘미세먼지 점수’ 칼럼을 만들어준다. ‘오존 점수’ 칼럼 또한 동일한 방법으로 만들어준다. result\_6\_1에서는 result\_6을 지역을 기준으로 ‘미세먼지 점수’ 칼럼의 평균값을 데이터프레임으로 저장해주고 result\_6\_2에는 result\_6을 지역을 기준으로 ‘오존 점수’ 칼럼의 평균값을 데이터프레임으로 저장해준다. result\_6\_1과 result\_6\_2의 인덱스를 reset해주고 for문을 이용해 최소 ‘미세먼지 점수’ 평균값과 ‘오존 점수’의 평균값을 min\_du, min\_oz에 저장해주고, result\_6\_1[‘미세먼지 점수’] 안에 데이터 중 min\_du에 해당된다면 해당 지역과 점수를 같이 출력해주고 오존도 같은 방식으로 진행한다.



[그림 21. 6번 항목 코드]

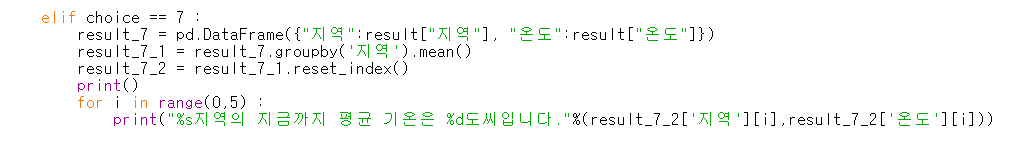
○ 7번 : 과거부터 현재까지 전체 지역의 평균 온도 구하기

- 입력데이터 : 없음



[그림 22. 7번 항목 출력]

- ‘지역’, ‘ 온도’ 칼럼만 모아둔 데이터프레임 result\_7을 만들고 ‘지역’칼럼을 기준으로 지역별 기온의 평균값을 구해 result\_7\_1을 만든 후, 인덱스를 제거하여 result\_7\_2에 저장한다. 그리고 차례로 result\_7\_2에 있는 값들을 출력한다.



[그림 23. 7번 코드]

1. 결론

본 프로젝트를 통해 웹 크롤링을 통해 정보를 수집하고, 수집한 정보를 Maria DB에 저장하는 작업과 저장된 데이터를 기반으로 Python 언어를 이용해 다양한 출력 결과를 도출 할 수 있었다.

1. 네이버 날씨. <https://weather.naver.com/> [↑](#footnote-ref-1)