출산율 관련 데이터 분석 및 시각화 보고서

- 2020444101
- 민찬기

- GItHub
- https://github.com/MCK-OOTS/DataVisualization

주제 선정 이유

- 최근 전례가 없는 출산율을 보이고 있는 대한민국의 출산율에 대한 원인과 관련 지표에 대한 궁금증을 해결하고 선택

- 흔히 말하는 여성의 사회진출, 청년들의 늦은 사회진출 같은 지표가 출산율과 연관이 있을까? 하는 궁금증을 해결하고자 이 주제를 선정

개발 환경 및 라이브러리, API

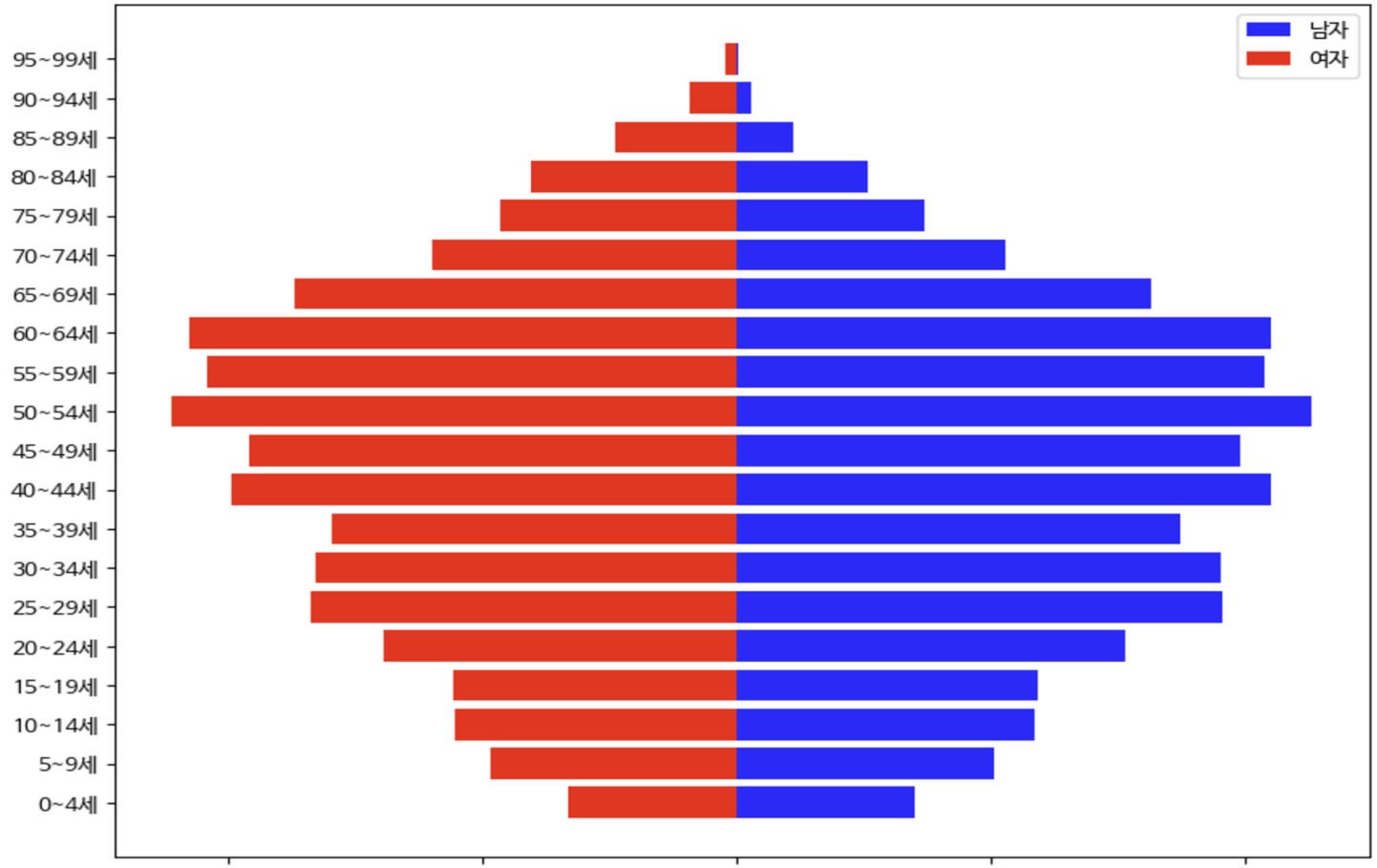
Google Colaboratory

pandas

matplotlib

seaborn

대한민국의 저출산이 심각한 문제가 되는 이유



1. 저출산 자체의 문제
2. 젊은 층의 인구 감소
3. 고령화 사회
4. 노동력 부족

저출산이 지속될 경우 젊은 층이 주 경제활동 인구가 되는 시기가 되면 2~3배 되는 윗 세대를 부양 해야하는 부담을 짊어지게 된다. Ex) 국민연금

데이터 수집(Kosis 국가통계포털의 API)

- 요청 URL : https://kosis.kr/openapi/statisticsList.do
- 요청 파라미터

- apiKey : 발급 ApiKey
- itmId : 데이터 분류 코드
- objL1 : 데이터 분류 코드
- objL2
- ……..
- objL8
- Format : 응답형식(JSON)
- jsonVD : JSON응답형식 (Y)
- PreSe : 기간별 데이터 요청(Y, M)
- newEstPrdCnt : 최신 데이터 연도
- orgid : 데이터 제공 조직 코드
- tblid : 테이블ID

```
# API URL (KOSIS 국가 통계 포털)
url = "https://kosis.kr/openapi/Param/statisticsParameterData.do"

# 요청 파라미터
params = {
    "apiKey": "OWVmNjBiOTgzZjhiNzYyNjYwNjU0YWUxYTNkYTAzMWM=",  # API 키
    "itmId": "T01+T02+",  # 데이터 분류 코드
    "objL1": "00+",  # 데이터 분류 코드
    "objL2": "005+010+015+020+025+030+035+040+045+050+055+060+065+070+075+080+085+090+095+100+",  # 데이터 분류 코드
    "format": "json",  # 응답 형식
    "jsonVD": "Y",  # JSON 형식
    "prdSe": "Y",  # 기간별 데이터 요청
    "newEstPrdCnt": "1",  # 최신 3년 데이터
    "orgId": "101",  # 데이터 제공 조직 ID (통계청)
    "tblId": "DT_1IN1503",  # 테이블 ID (인구 구조 관련 테이블)
}
```

데이터 수집(OECD Data Exploer API)

- API 데이터 쿼리 구문 :
- {호스트 URL}/{에이전시 식별자},{데이터세트 식별자},{데이터세트 버전}/{데이터선택}?{기타 선택 매개변수}

```
# API URL 설정
url = (
    "https://sdmx.oecd.org/public/rest/data/" #HOST URL
    "OECD.ECO.MPD.DSD_AN_HOUSE_PRICES@DF_HOUSE_PRICES," #Agency Identifier
    "1.0/OECD+USA+GBR+CHE+TUR+SWE+ESP+SVN+SVK+PRT+POL+NZL+NOR+NLD+MEX+LVA+LTU+LUX+KOR+JPN+HUN+ISR+ITA+IRL+ISL+GRC+DEU+FRA+EST+FIN+I
    "startPeriod=2023&endPeriod=2023&dimensionAtObservation=AllDimensions&format=genericdata"#Other Option
)
```

데이터 수집(OECD .xlsx)

- OECD API에서 제공하지 않는 데이터는 OECD dataset의 .xlsx파일을 GoogleDrive에 저장 후 불러내어 사용

SF1.2 Children in families	PDF XLSx
SF1.3 Further information on living arrangements of children	PDF XLSx
SF1.4 Population by age of children and youth dependency ratio	PDF XLSx
SF1.5 Living conditions of children	PDF XLSx
Fertility indicators	
SF2.1 Fertility rates	PDF XLSx @OECD_Social
SF2.2 Ideal and actual number of children	(under development)
SF2.3 Age of mothers at childbirth and age-specific fertility	PDF XLSx
SF2.4 Share of births outside of marriage	PDF XLSx
SF2.5 Childlessness	PDF XLSx
Marital and partnership status	
SF3.1 Marriage and divorce rate	PDF XLSx
SF3.2 Family dissolution and children	PDF XLSx
SF3.3 Cohabitation rate and prevalence of other forms of partnership	PDF XLSx

Kosis 데이터 수집(대한민국 인구 수)

1. KOSIS 국가 통계 포털의 API를 통해 데이터 수집
2. itmld : 총인구_남자(명), 총인구_여자(명)
3. objL1 : 전국
4. objL2 : 원하는 데이터의 나이대(0~99세)
5. newEstPrdCnt : 최근 1년의 데이터
6. orgid : 통계청의 제공 데이터
7. tblid : 데이터 가져올 테이블ID

```
# API URL (KOSIS 국가 통계 포털)
url = "https://kosis.kr/openapi/Param/statisticsParameterData.do"

# 요청 파라미터
params = {
    "apiKey": "OWVmNjBiOTgzZjhiNzYyNjYwNjU0YWUxYTNkYTAzMWM=",  # API 키
    "itmId": "T01+T02+",  # 데이터 분류 코드
    "objL1": "00+",  # 데이터 분류 코드
    "objL2": "005+010+015+020+025+030+035+040+045+050+055+060+065+070+075+080+085+090+095+100+",  # 데이터 분류 코드
    "format": "json",  # 응답 형식
    "jsonVD": "Y",  # JSON 형식
    "prdSe": "Y",  # 기간별 데이터 요청
    "newEstPrdCnt": "1",  # 최신 3년 데이터
    "orgId": "101",  # 데이터 제공 조직 ID (통계청)
    "tblId": "DT_1IN1503",  # 테이블 ID (인구 구조 관련 데이블)
}


# API 요청 보내기
response = requests.get(url, params=params)

# JSON 데이터 파싱 및 데이터 확인
population_data = response.json()
#print(population_data)
```

OECD 데이터 수집(주택 가격 지수)

- OECD 데이터 요청 파라미터에 맞춰서 작성

```
# API URL 설정
url = (
    "https://sdmx.oecd.org/public/rest/data/" #HOST URL
    "OECD.ECO.MPD,DSD_AN_HOUSE_PRICES@DF_HOUSE_PRICES," #Agency Identifier
    "1.0/OECD+USA+GBR+CHE+TUR+SWE+ESP+SVN+SVK+PRT+POL+NZL+NOR+NLD+MEX+LVA+LTU+LUX+KOR+JPN+HUN+ISR+ITA+IRL+ISL+GRC+DEU+FRA+EST+FIN+DNK+CZE+CRI+CHL+COL+CAN+BEL+AUT+AUS.A.HPI_YOH.?"#Data S
    "startPeriod=2023&endPeriod=2023&dimensionAtObservation=AllDimensions&format=genericdata"#Other Option
)

# 데이터 요청
response = requests.get(url)



# XML 데이터 파싱
root = ET.fromstring(response.text)
```

데이터 수집 및 전처리(OECD평균 초혼 연령)

- API로 제공하지 않는 데이터 중 xlsx파일로 제공하는 데이터는 구글 드라이브에 저장한 후에 사용

- 파일을 오픈시 국가를 인덱스로 잡고 DF에 저장

	Male mean age at first marriage			Female mean age at first marriage		
	1990	2000	2019	1990	2000	2019
Spain	27.8	30.2	36.1	25.6	28.1	33.9
Sweden	30.3	33	36.7	27.7	30.4	34.1
Norway	29	30.9	35.7	26.4	28.4	33.1
Italy	28.9	30.9	35.5	25.9	27.8	32.7
France	..	30.7	35.2	..	28.4	33.1
Denmark	30.5	32.5	35.1	27.8	29.9	32.8
Iceland	29.4	33.3	34.4	26.9	30.6	32.4
Portugal	26.6	27.4	33.2	24.6	25.2	31.5
Luxembou	27.7	30.3	34.8	25.6	27.4	32.1
United Kin	27.2	30.5	33.7	25.2	28.2	31.8
Austria	27.7	30	34.7	25.2	27.4	32
Belgium	26.5	29.1	33.5	24.4	26.9	31.2
Ireland	28.7	..	33.8	26.6	..	31.9
Finland	28.4	30.5	34.2	26.3	28.3	31.9
Netherlanc	28.5	30.7	34.4	26.1	28	31.9
Chile	..	27.7	32.7	..	25.6	31.5
OECD-26 a	27.7	29.8	33.5	25.2	27.3	31.2
Germany	28.2	30.5	34	25.5	27.7	31.2
Slovenia	26.9	29.9	33.9	23.9	27	31.2
EU-26 avei	-	-	33.2			30.7
Switzerlanc	29.5	30.8	33.1	27	28.2	30.7

```
#OECD에서 제공하는 .xlsx 파일 이용
#GoogleDrive에 저장 후 사용
#Google Drive 마운트
from google.colab import drive
drive.mount('/content/drive',force_remount=True)

#파일 경로
file_path = '/content/drive/My Drive/3-2BigData/'

#Excel 파일 읽기
try:
    df_OECD_avg_age = pd.read_excel(file_path+'초혼나이.xlsx', engine='openpyxl', header=1, index_col=0) #index_col = 국가 이름을 인덱스로 지정
    df_OECD_avg_age = df_OECD_avg_age.loc[:, [2019, '2019.1']] #.xlsx의 파일을 불러오면서 header=1 옵션으로 인해 같은 연도 구분을 위해 여성의 데이터 쪽의 연도는 '.1'이 붙으면서 문자 취급
    df_OECD_avg_age.columns = ['OECD 남성평균 초혼연령', 'OECD 여성평균 초혼연령'] #컬럼 이름 변경
    df_OECD_avg_age['전체 평균 초혼연령'] = df_OECD_avg_age[['OECD 남성평균 초혼연령', 'OECD 여성평균 초혼연령']].mean(axis=1)#남녀의 연령 데이터를 바탕으로 평균 값 구해서 컬럼추가
    df_OECD_avg_age = df_OECD_avg_age.sort_values(by='전체 평균 초혼연령', ascending=True)#내림차순 정렬

except Exception as e:
    print("파일을 읽는 중 오류 발생:", e)
```

대한민국 출산율 데이터 전처리

df_fertility – 비교 국가들의 출산율 데이터

1. DF로 저장
2. 원하는 데이터 컬럼만 추출 후 저장
3. 특정국가의 데이터만 따로 추출해서 저장

- 대표적인 출산율 개선국가(독일), 높은 출산율국가(프랑스)
- 대표적인 저출산국가(일본,이탈리아)

4. 컬럼 이름 변경
5. 저출산 국가와 높은 출산율 국가를 다시 분리

Df_all_fertility – 출산율 하위 10개국가의 출산율 데이터

1. DF로 저장
2. 원하는 컬럼 데이터만 추출 후 저장
3. 가장 최근 연도인 2021년도 데이터만 저장
4. 컬럼명 변경
5. 기본적으로 API값이 수치 데이터가 아닌 문자 데이터이기 때문에 flot형으로 형변환
6. 오름차순으로 정렬 후 head()를 통해 10개국가의 데이터만 저장

```
#전처리
df_fertility = pd.DataFrame(fertility_data)# list > DataFrame
df_fertility= df_fertility[['PRD_DE', 'C1_NM', 'DT']]#원하는 데이터 키 값만 저장
#대표적인 저출산국가(일본,이탈리아)와 높은 출산율(프랑스), 출산율 개선국가(독일)의 출산율 변화 비교
df_fertility= df_fertility[df_fertility['C1_NM'].isin(['대한민국', '일본','프랑스','독일','이탈리아'])]#원하는 국가의 데이터만 저장
df_fertility.columns = ['연도', '국가', '출산율(명)']# 데이터 컬럼 이름 변경
#저출산국가와 높은 출산율 국가 데이터를 다시 분리
df_high_fertility = df_fertility[df_fertility['국가'].isin(['프랑스', '독일', '대한민국'])]
df_low_fertility = df_fertility[df_fertility['국가'].isin(['일본','이탈리아','대한민국'])]



#전 세계 데이터
df_all_fertility = pd.DataFrame(fertility_data)# list > DataFrame
df_all_fertility = df_all_fertility [['PRD_DE', 'C1_NM', 'DT']]#원하는 데이터 키 값만 저장
df_all_fertility = df_all_fertility[df_all_fertility['PRD_DE'] == '2021']  # 2021년도 데이터만 저장
df_all_fertility.columns = ['연도', '국가', '출산율(명)']# 데이터 컬럼 이름 변경

#하위 10개국 데이터만 저장
df_all_fertility['출산율(명)'] = df_all_fertility['출산율(명)'].astype(float)  #하위 50개 국가 필터링을 위해 데이터를 float로 변경
df_all_fertility = df_all_fertility.sort_values(by="출산율(명)", ascending=True).head(10) #오름차순 정렬 후 10개의 국가 데이터만 저장

print(df_fertility)
print(df_all_fertility)
```

대한민국 청년(19~24세)실업률 데이터 전처리

1. DF로 바로 저장시 스칼라 값 에러로 인해 리스트로 데이터 가공 후 DF로 저장
2. 해당 데이터가 29년도 까지 받아지므로 데이터 기간을 선택해서 데이터 가공

```
# 단일 값(스칼라 값) 에러로인해 리스트로 저장 후 DF로 저장
data_list = []
for item in unemploy_data:
  data_list.append({
  "연도": item.get("PRD_DE"),
  "국가": item.get("C2_NM"),
  "청년 실업률(%)": item.get("DT")
})
df_unemploy = pd.DataFrame(data_list)

#2000~ 2021까지의 데이터만 저장 (2029년도 까지 데이터 제공)
df_unemploy = df_unemploy[df_unemploy['연도'].isin([str(year) for year in range(2000, 2022)])]

# 컬럼이름 변경
df_unemploy.columns = ['연도', '국가', '청년 실업률(%)']
print(df_unemploy)
```

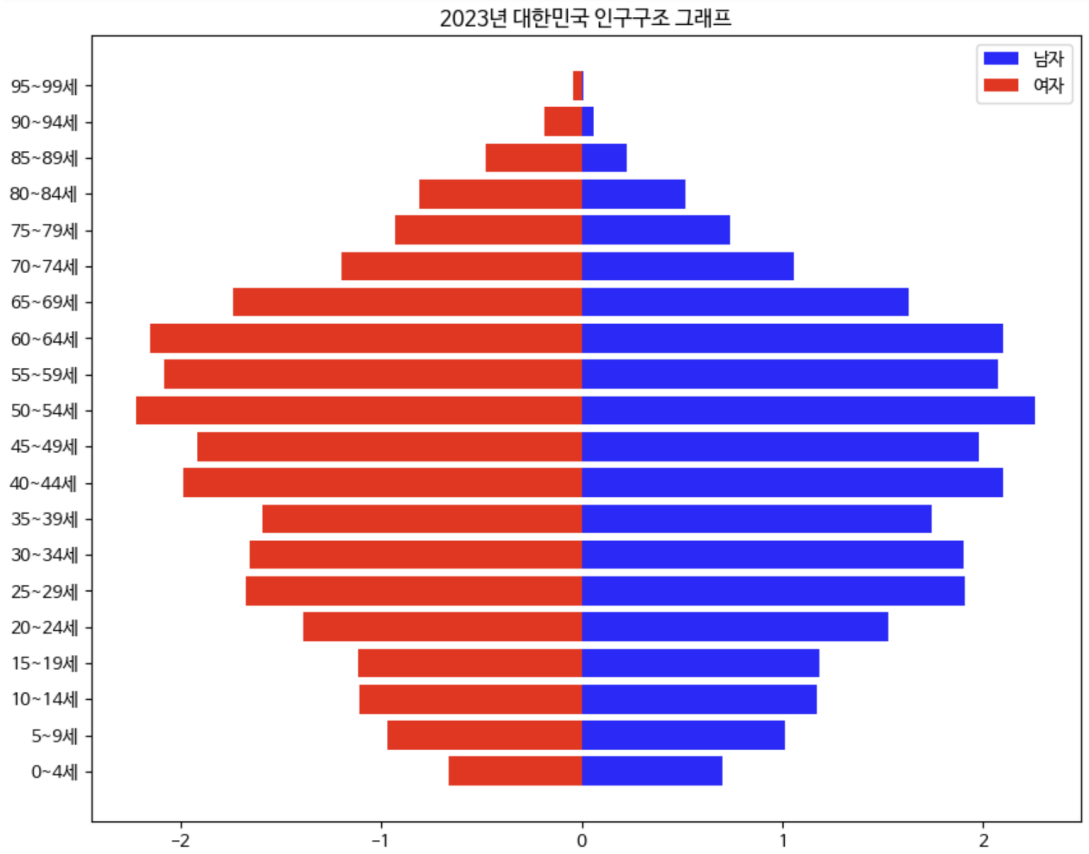
OECD 주택 가격 지수 데이터 전처리

- 개발자가이드에 CSV, XML, JSON을 제공한다고 되어 있지만 XMl 밖애 사용되지 않는다.

1. XML에서 관찰자(Obs)를 노드 기준으로탐색
2. 각 관찰자 노드에서 관측값의 세부정보 추출
   (ObsKey: REF_AREA(국가),TIME_PREIOD(기간))
   (ObsValue) : 실제 관측된 데이터 값(주택가격지수)
3. 데이터를 기준으로 정렬하기 위해 수치 데이터로 형변환
4. 시각화했을 때 높은게 위에서부터 나오도록 다시 정렬

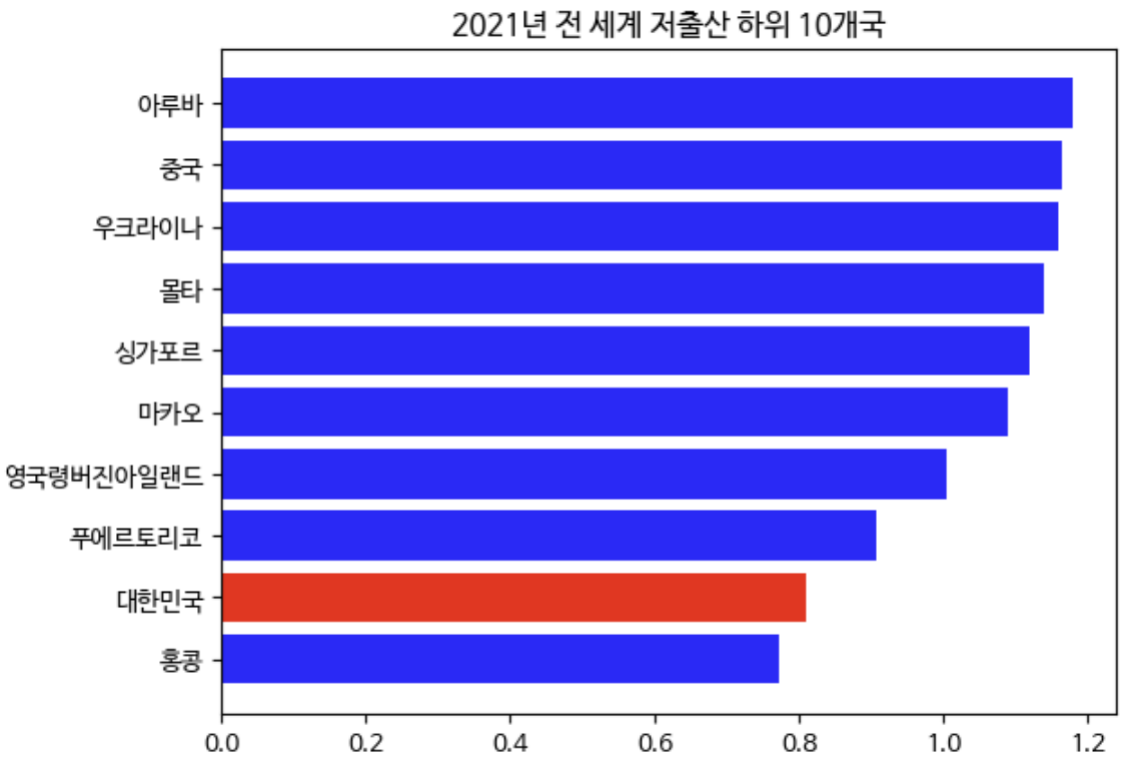
```python
#데이터 저장 리스트
countries = [] #나라
years = [] #연도
values = [] #집

# 각 'Obs' 항목에서 데이터 추출
for obs in root.findall('.//generic:Obs', namespaces={'generic': 'http://www.sdmx.org/resources/sdmxml/schemas/v2_1/data/generic'}):#관측값 obs를 namespaces에서 찾음
# 기간, 국가, 값 추출
  for obs_key in obs.findall('.//generic:ObsKey', namespaces={'generic': 'http://www.sdmx.org/resources/sdmxml/schemas/v2_1/data/generic'}): #obs를 바탕으로  국가와 기간 값 추출
    for value_elem in obs_key.findall('.//generic:Value', namespaces={'generic': 'http://www.sdmx.org/resources/sdmxml/schemas/v2_1/data/generic'}):
      if value_elem.attrib.get('id') == 'REF_AREA': #국가
        country = value_elem.attrib.get('value')
      if value_elem.attrib.get('id') == 'TIME_PERIOD': #기간
        year = value_elem.attrib.get('value')

  # 값 추출
  obs_value = obs.find('.//generic:ObsValue', namespaces={'generic': 'http://www.sdmx.org/resources/sdmxml/schemas/v2_1/data/generic'})
  if obs_value is not None:
    value = obs_value.attrib.get('value')#주택가격

  # 데이터 리스트에 추가
  if country and year and value:
    countries.append(country)
    years.append(year)
    values.append(value)

# DataFrame으로 변환
df_hous_price = pd.DataFrame({
  'Country': countries,
  'Year': years,
  'Value': values
})

df_hous_price['Value'] = df_hous_price['Value'].astype(float) #데이터 정렬을 위해 수치 데이터로 형변환
df_hous_price = df_hous_price.sort_values(by="Value", ascending=True)

# 데이터 출력
print(df_hous_price.head())
```

대한민국 인구 수 데이터 시각화



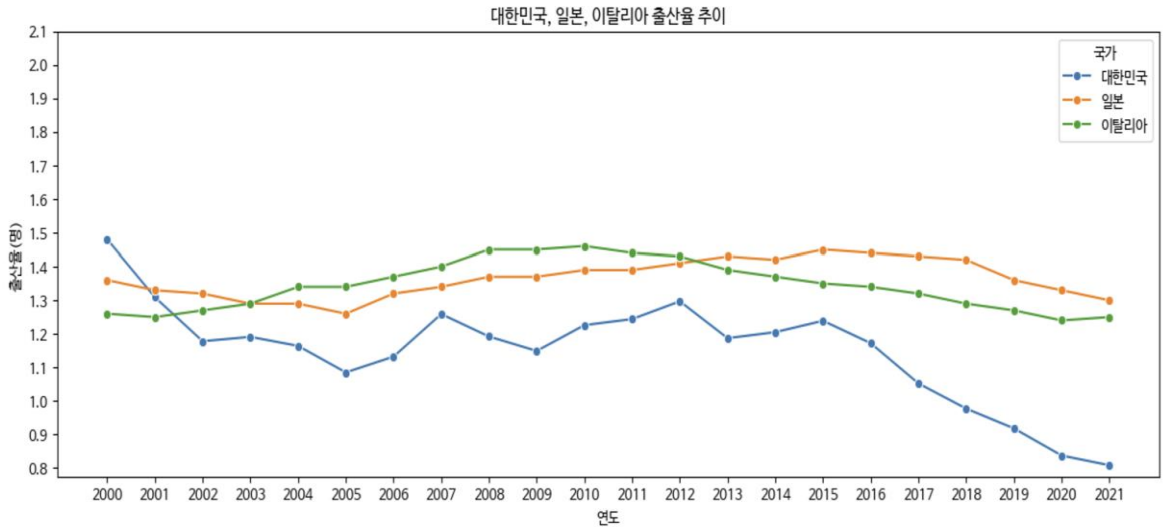
- 40 ~ 64세 가 인구의 다수를 차지하는 인구구조
- 0~23세의 인구가 절대적으로 부족하다.

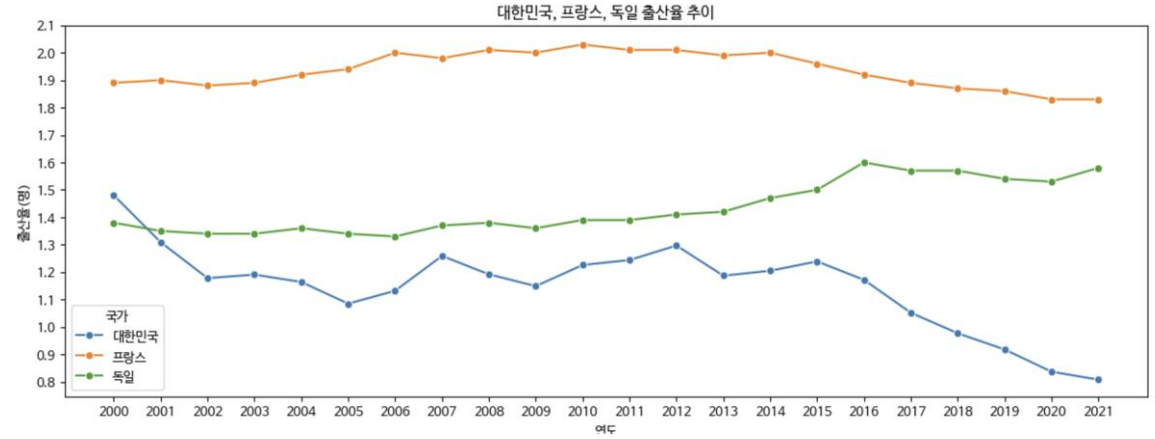
대한민국 출산율 데이터 시각화



- 2021년 기준 최하위권인 대한민국 출산율

대한민국 출산율 데이터 시각화



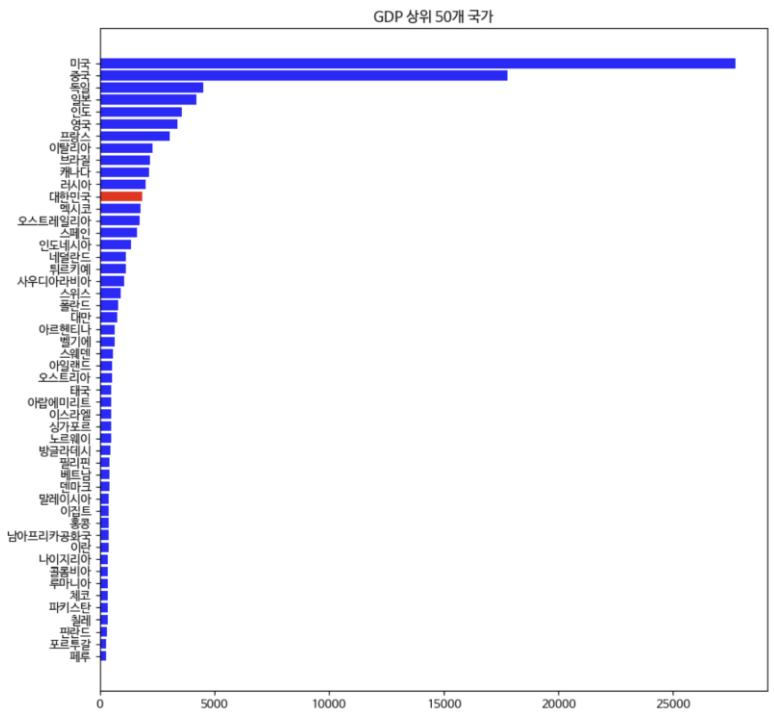


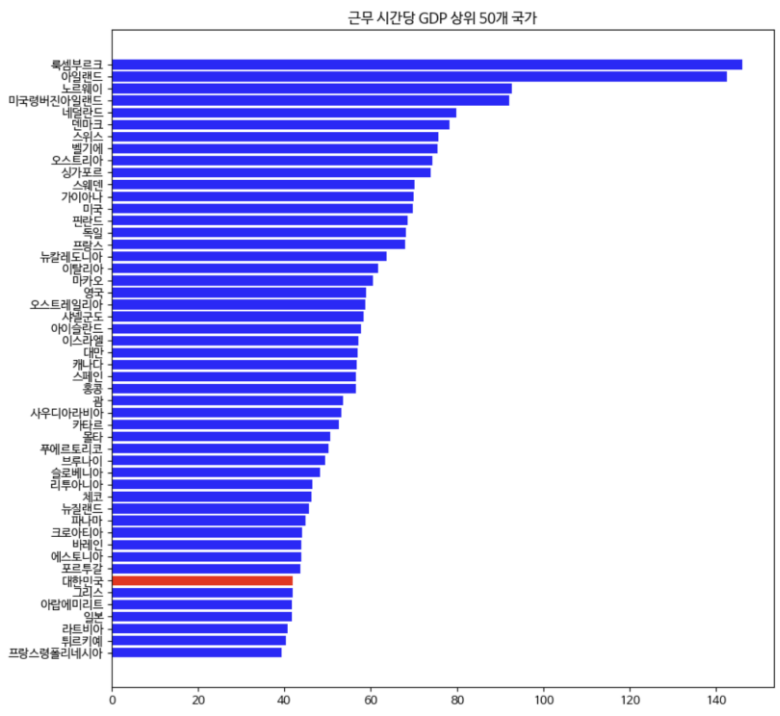
- 일본 : 대한민국과 비슷한 대표적 저출산 국가이자 비슷한 경제 구조 및 문화를 가진 국가
- 이탈리아 : 유럽의 대표적인 저출산 국가

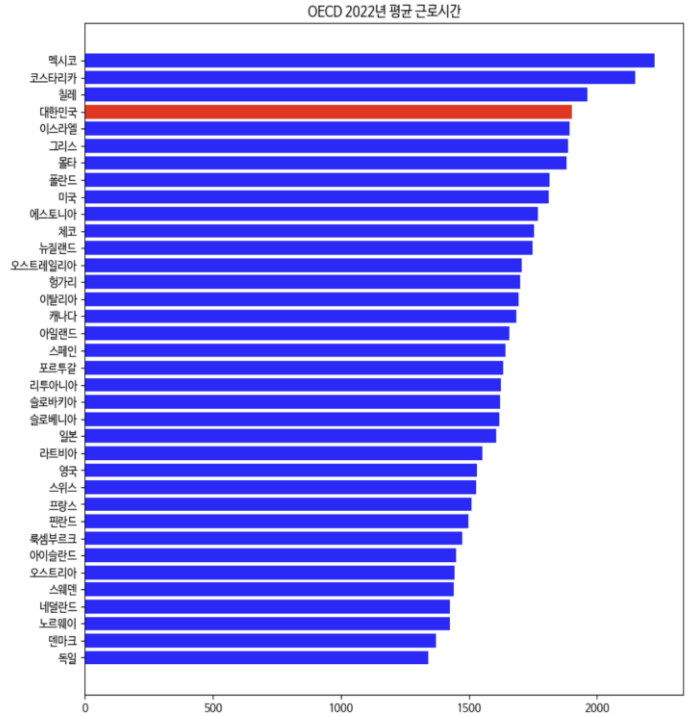
- 독일 : 저출산 극복 국가
- 프랑스 : 대표적으로 높은 출산율을 보이는 국가

- 2000년대의 각 국가별 출산율 추이

GDP 및 근무시간에 관한 시각화

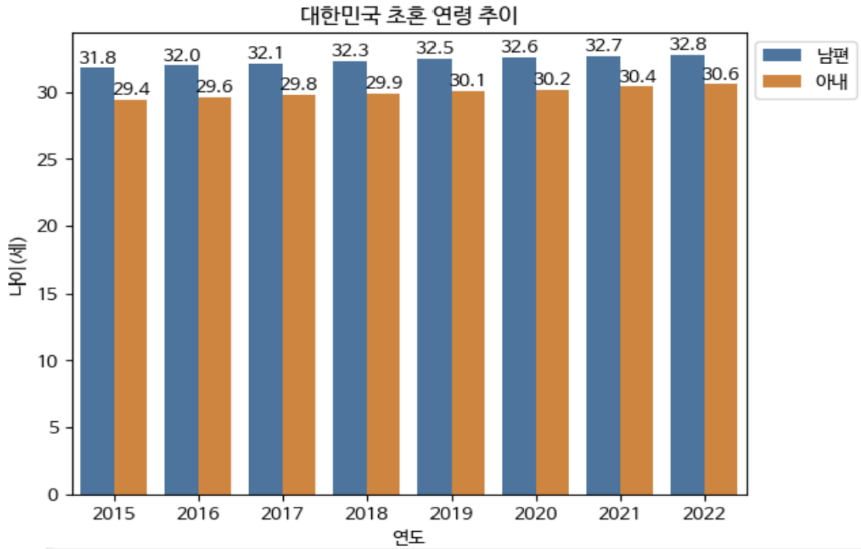


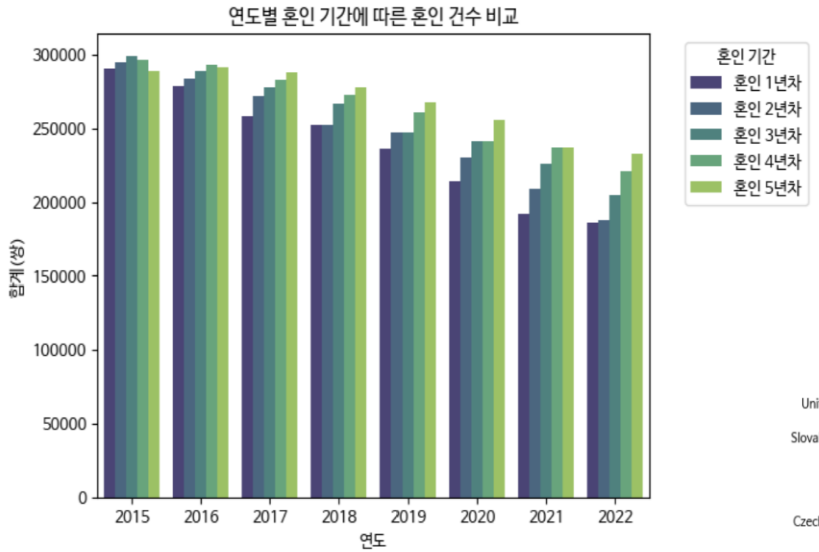


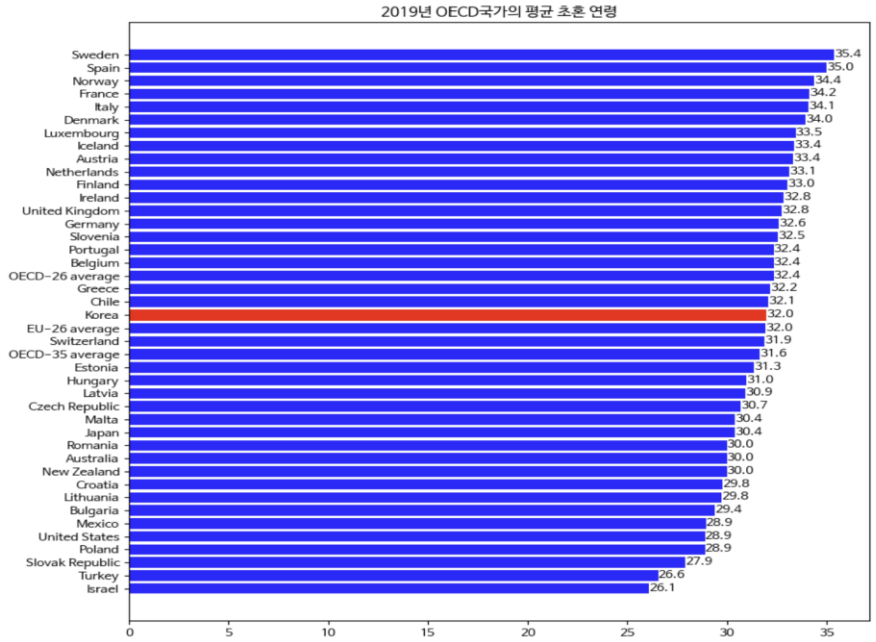


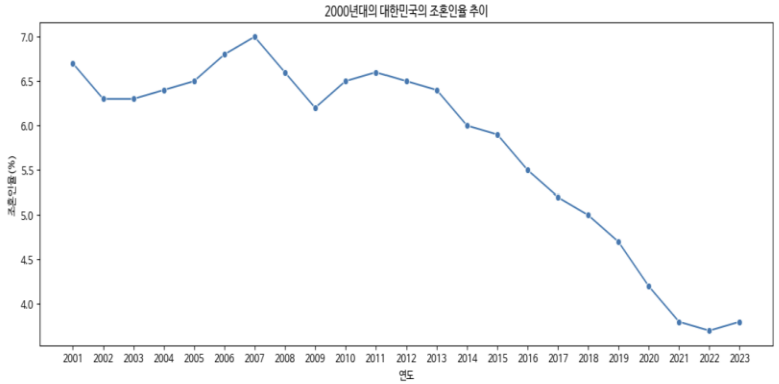
- GDP를 보면 결코 작지않은 경제규모를 갖고 있는 대한민국이지만 근로시간과 관련된 데이터를 보면 높은 GDP수치가 좋은 의미만 갖지는 아니라는 사실을 의미한다.
- 대한민국의 경우 높은 GDP대비 근무시간당 GDP는 하위권 수준, 근로시간은 상단에 위치한다.

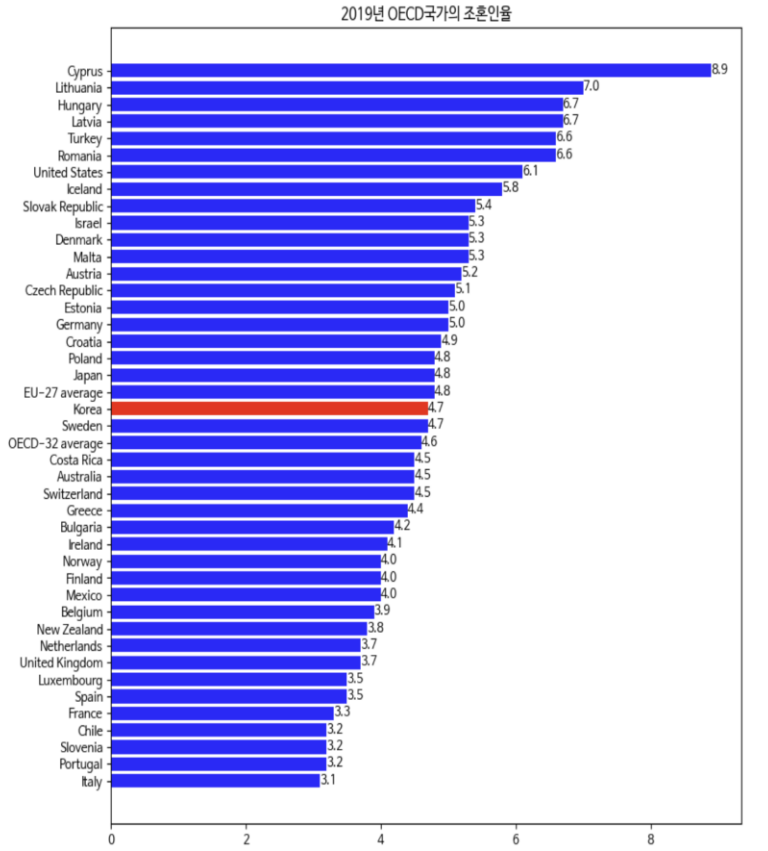
결혼에 관한 데이터 시각화



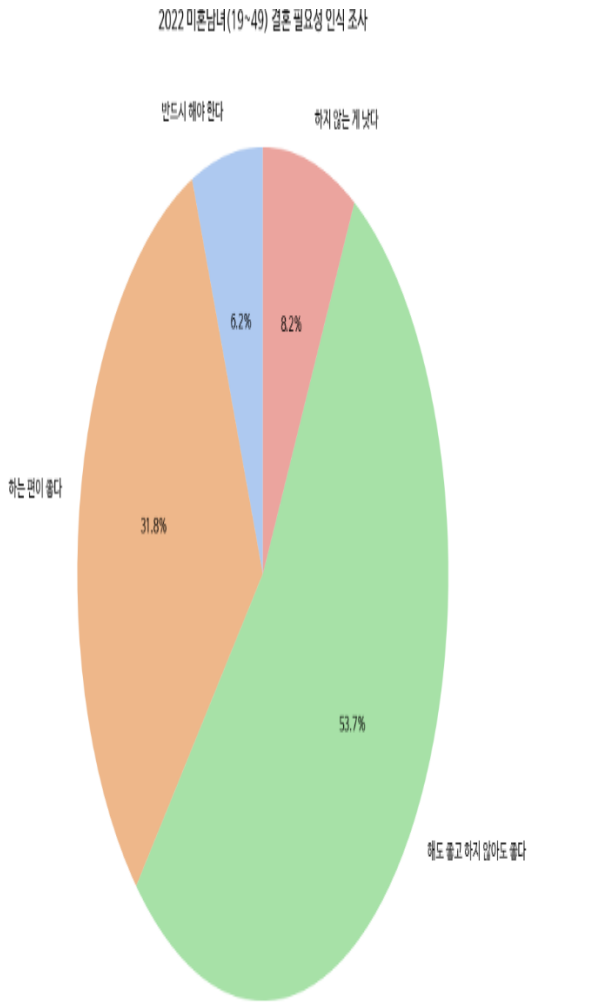


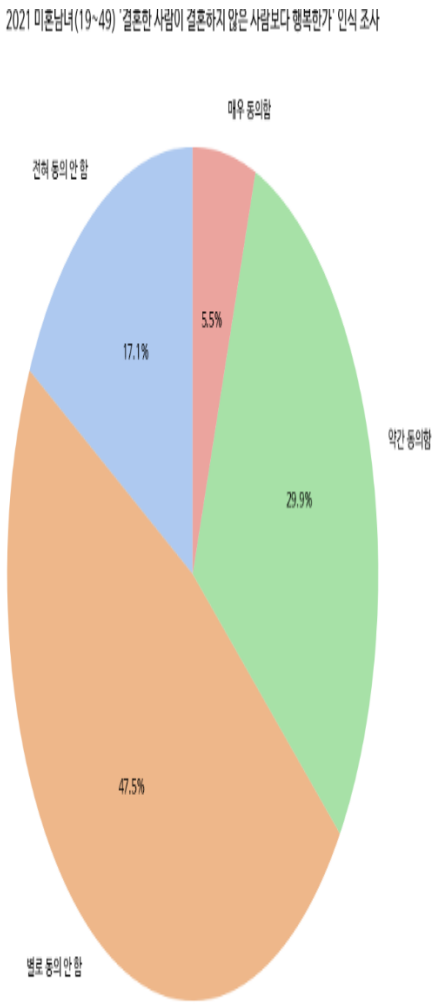






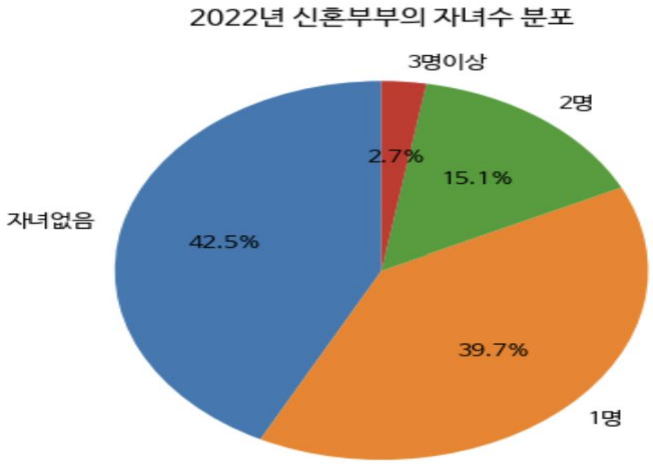
결혼에 관한 데이터 시각화

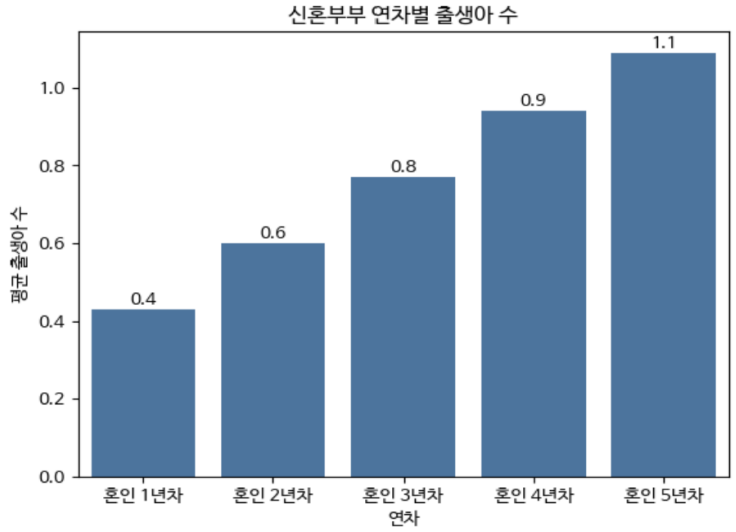


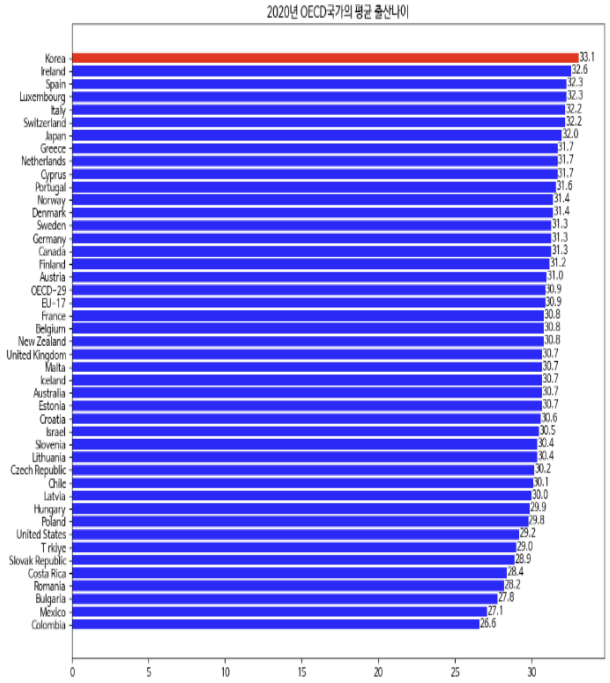


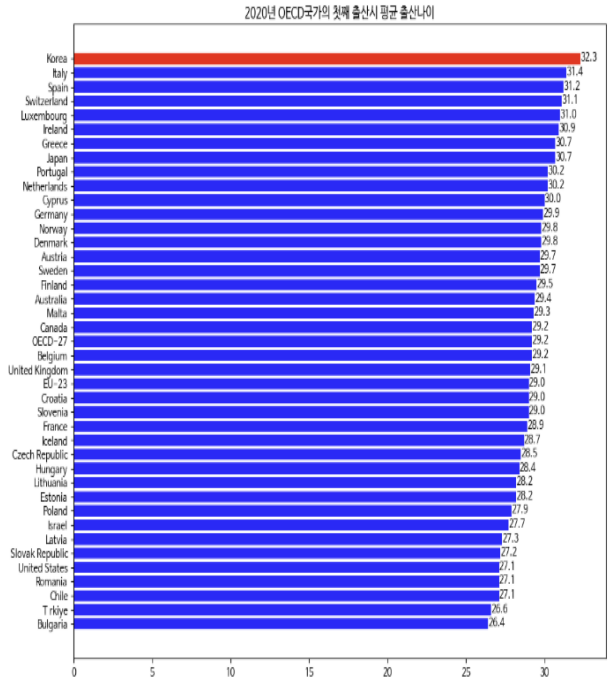
- 지속적으로 증가하는 평균 결혼 연령
- 지속적으로 내려가는 혼인 건수
- 결혼에 대해 부적정이거나 긍정적이지 않다는 의견이 다수를 차지한다.

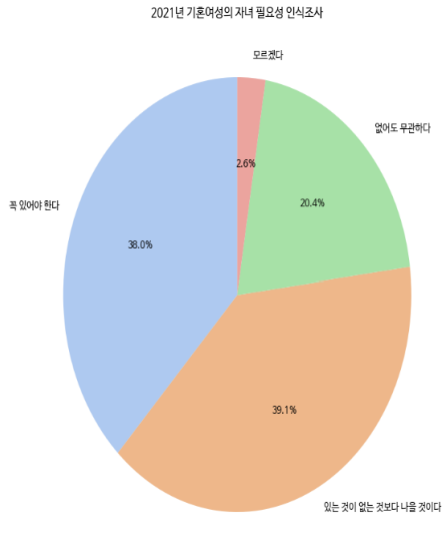
출생아 관한 데이터 시각화

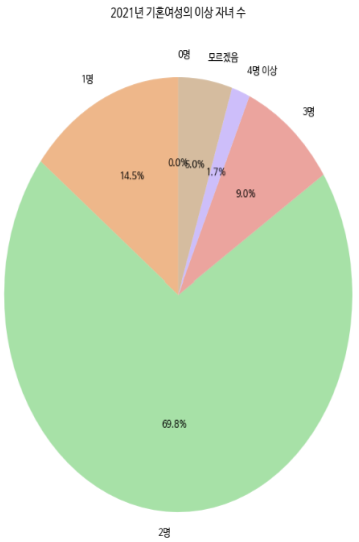












출생아 관한 데이터 시각화

- 신혼부부 4년차 까지 평균 1이 안되는 출생아 수
- OECD 평균 출산나이 1위, 첫째 출산나이 1위
- 기혼여성의 자녀에 관한 인식조사를 보면 아이를 원하는 사람은 많다
- 그러나 실제 자녀수의 데이터를 보면 자녀가 없거나 1명인 신혼부부가 대다수임을 알 수 있다.

결론

- 가장 큰 문제는 늦은 나이에 아이를 출산한다는 것 이다.
- 결혼을 하고 바로 아이를 갖지 않고 2~3년이 지나야 아이를 출산한다.
- 첫째 아이 자체를 늦게 출산하다 보니 둘째를 갖는 것에 나이에 대한부담이 큰 것으로 보인다.
- 실제로 아이가 없거나 있어도 1명인 경우가 다수를 차지한다.

- 두번째로 아이를 낳기 위해서 결혼을 해야 결혼하는 사람의 수는 지속적으로 줄어들고 결혼을 하는 평균 나이도 점점 늘어나는 추세이다.

- 세 번째로는 높은 근무시간 대비 낮은 GDP라고 생각된다. 비교국가로 선정한 일본의 경우 높은 근로시간을 보이지만 실제 연간 GDP는 비교한 5국가 중에는 제일 높은 수치를 보인다. 이탈리아의 경우 최근 근로시간이 상승한 모습을 보이지만 그럼에도 불구하고 높은 GDP와 비교적 여전히 낮은 근로시간을 보인다. 그러나 대한민국의 경우 높은 근로시간 대비 높지 않는 GDP를 보인다.

- 그 외의 우리가 평소에 생각하는 여성의 사회진출(대학, 취업), 집 값, 고용률, 청년 실업률 등은 여러 가지는 데이터 분석으로 볼 때 큰 관련이 있다고 보기 힘들다.