

# P4\_en

December 24, 2018

```
In [1]: from urllib import request
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
```

```
In [2]: request.urlretrieve ("https://raw.githubusercontent.com/jakevdp/data-CDCbirths/master/births.csv")
births = pd.read_csv("births.csv")
```

```
In [3]: births.describe()
```

```
Out[3]:
```

	year	month	day	births
count	15547.000000	15547.000000	15067.000000	15547.000000
mean	1979.037435	6.515919	17.769894	9762.293561
std	6.728340	3.449632	15.284034	28552.465810
min	1969.000000	1.000000	1.000000	1.000000
25%	1974.000000	4.000000	8.000000	4358.000000
50%	1979.000000	7.000000	16.000000	4814.000000
75%	1984.000000	10.000000	24.000000	5289.500000
max	2008.000000	12.000000	99.000000	199622.000000

```
In [4]: births.head()
```

```
Out[4]:
```

	year	month	day	gender	births
0	1969	1	1.0	F	4046
1	1969	1	1.0	M	4440
2	1969	1	2.0	F	4454
3	1969	1	2.0	M	4548
4	1969	1	3.0	F	4548

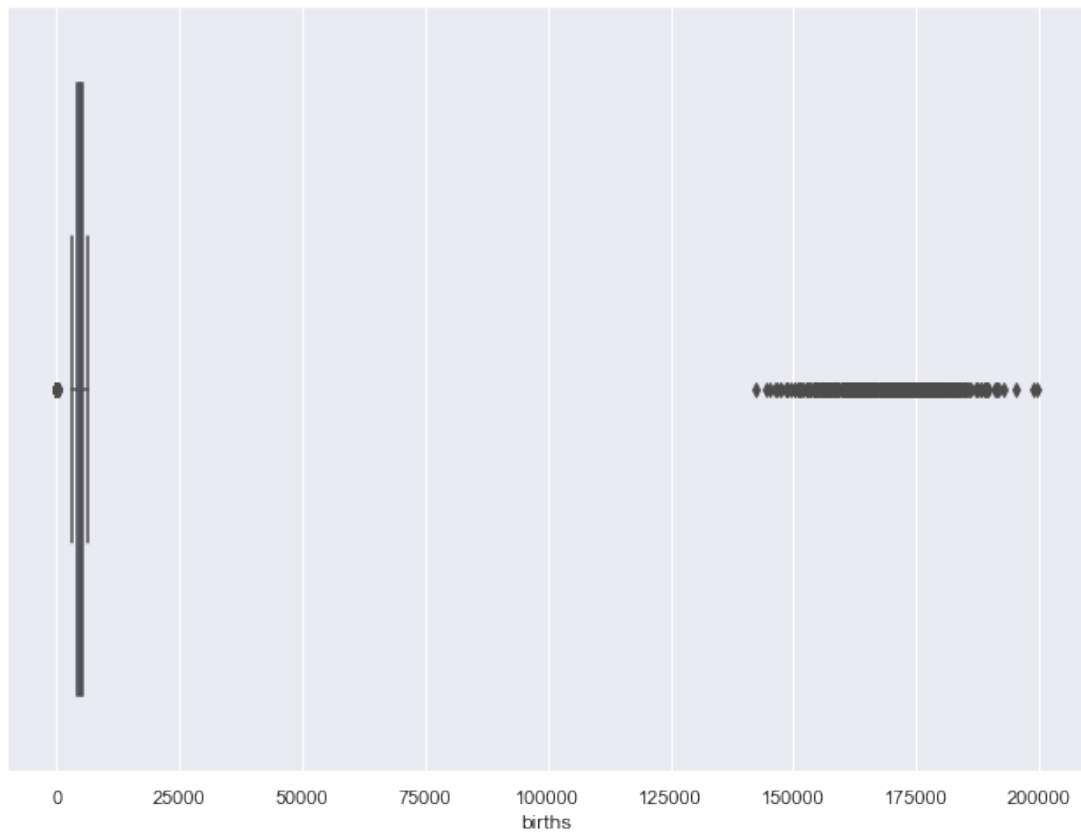
```
In [5]: births['decade'] = 10 * (births['year'] // 10)
```

```
In [6]: births.head()
```

```
Out[6]:
```

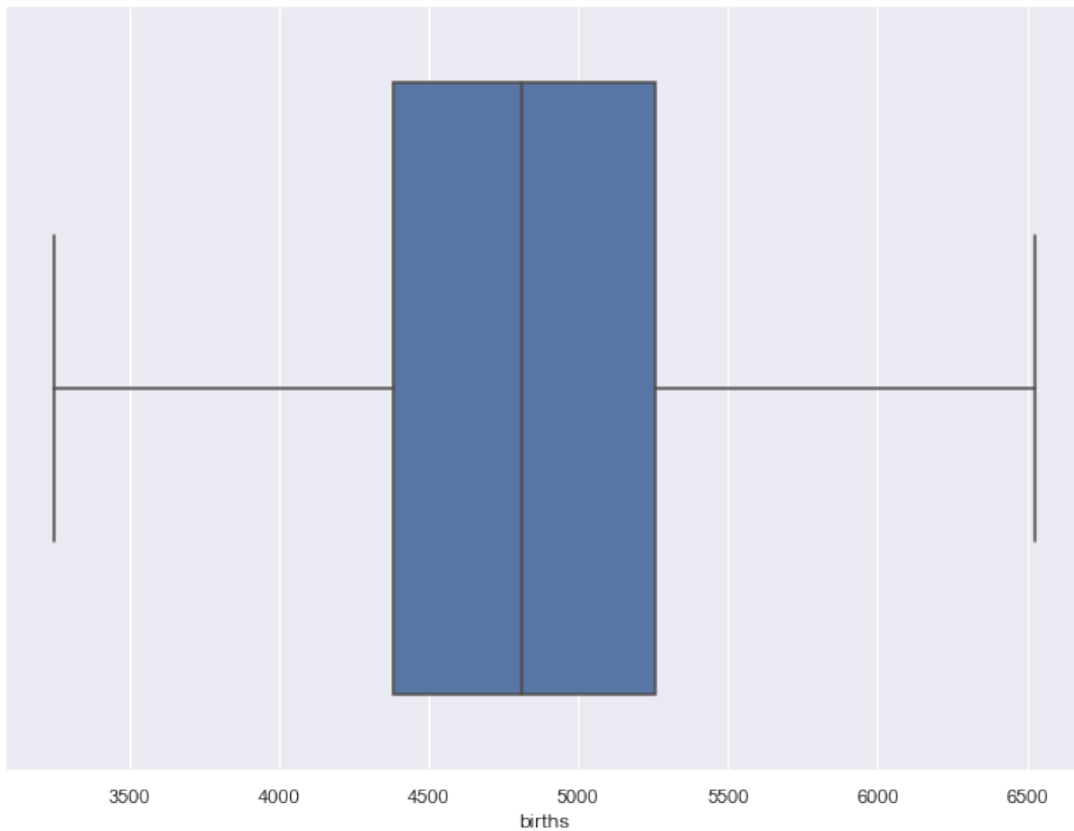
	year	month	day	gender	births	decade
0	1969	1	1.0	F	4046	1960
1	1969	1	1.0	M	4440	1960
2	1969	1	2.0	F	4454	1960
3	1969	1	2.0	M	4548	1960
4	1969	1	3.0	F	4548	1960

```
In [7]: fig = plt.figure(figsize=(11,8))  
fig = sns.boxplot(births.births)
```



Let's take care of outliers.

```
In [8]: births = births.query('(births > 1000) & (births < 100000)')  
fig = plt.figure(figsize=(11,8))  
fig = sns.boxplot(births.births)
```



```
In [9]: births.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 14610 entries, 0 to 15066
Data columns (total 6 columns):
year      14610 non-null int64
month     14610 non-null int64
day       14610 non-null float64
gender    14610 non-null object
births    14610 non-null int64
decade    14610 non-null int64
dtypes: float64(1), int64(4), object(1)
memory usage: 799.0+ KB
```

Days, months and years must be integers.

```
In [10]: for field in ["day", "month", "year"]:
         births[field] = births[field].astype(int)
```

Extracting day of the week. Always use built-in libraries to work with dates.

```
In [11]: # create a datetime index from the year, month, day
births.index = pd.to_datetime(10000 * births.year +
                               100 * births.month +
                               births.day, format='%Y%m%d')
```

```
births['dayofweek'] = births.index.dayofweek
```

```
In [12]: births.head()
```

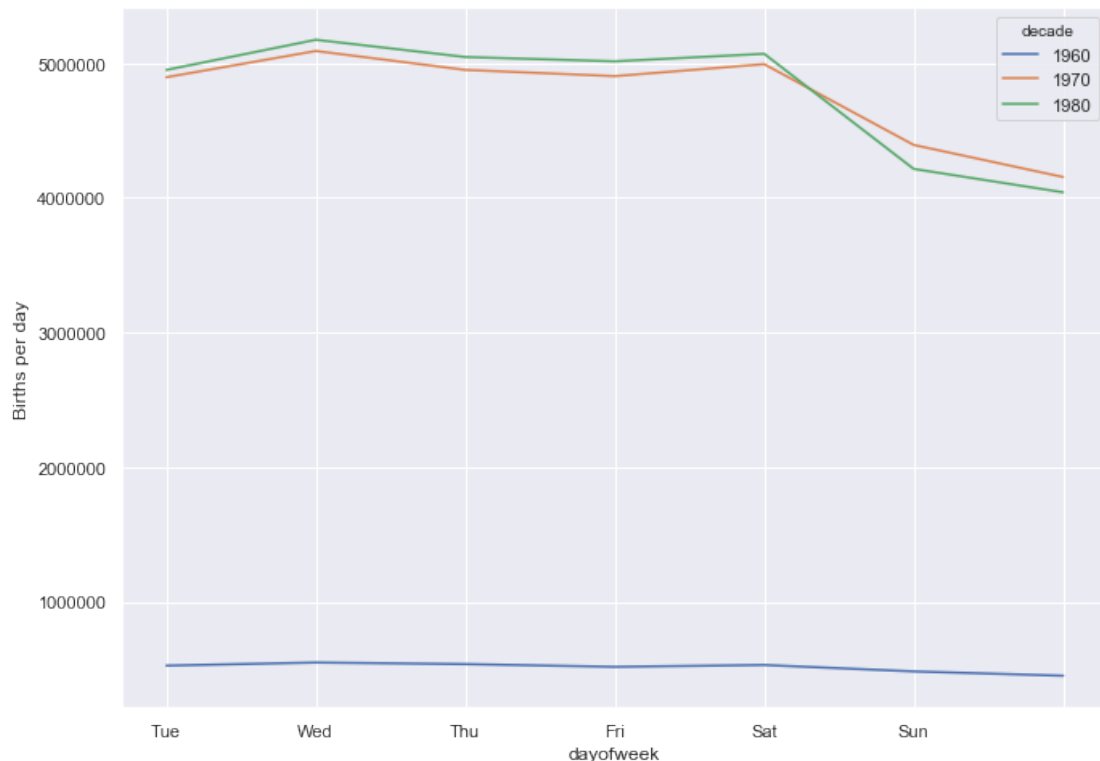
```
Out[12]:
```

	year	month	day	gender	births	decade	dayofweek
1969-01-01	1969	1	1	F	4046	1960	2
1969-01-01	1969	1	1	M	4440	1960	2
1969-01-02	1969	1	2	F	4454	1960	3
1969-01-02	1969	1	2	M	4548	1960	3
1969-01-03	1969	1	3	F	4548	1960	4

And finally, number of births per day.

```
In [13]: fig = plt.figure(figsize=(11,8))
births_per_day = births.pivot_table('births', index='dayofweek',
                                     columns='decade', aggfunc='sum')
births_per_day.plot(figsize=(11,8))
plt.gca().set_xticklabels(['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'])
plt.ylabel('Births per day');
```

<Figure size 792x576 with 0 Axes>



This part is optional. The low number of births in the 1960s is most likely due to missing data in your dataset. You cannot do anything about this, but it probably makes more sense to have a look at the average number of births:

```
In [14]: fig = plt.figure(figsize=(11,8))
        births_per_day = births.pivot_table('births', index='dayofweek',
        columns='decade', aggfunc='mean')
        births_per_day.plot(figsize=(11,8))
        plt.gca().set_xticklabels(['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'])
        plt.ylabel('Births per day');
```

<Figure size 792x576 with 0 Axes>

