Covid-19 Vaccination Campaign in Germany

The data used here were provided by Robert Koch Institute and the German federal ministry of Health.

These institutions publish the datasets and some analysis on the page impfdashboard.de.

In [1]:
```
# standard library
import datetime
import math
```

In [2]:
```
# third party
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import requests
```

In [3]:
```
today = datetime.datetime.today().strftime('%Y-%m-%d')
today
```

Out[3]:
```
'2021-04-17'
```

In [4]:
```
yesterday = (datetime.datetime.today() + datetime.timedelta(days=-1)).strftime('%Y-%m-%d')
yesterday
```

Out[4]:
```
'2021-04-16'
```

Set Defaults

In [5]:
```
# style like ggplot in R
plt.style.use('ggplot')
```

Get and Transform Data

In [6]:
```
vaccination_data_permalink = 'https://impfdashboard.de/static/data/germany_vaccinations_timeseries_v2.tsv'
vaccinations = pd.read_csv(
    vaccination_data_permalink,
    sep="\t")
```

In [7]:
```
vaccinations.tail(3)
```

Out[7]:

	date	dosen_kumulativ	dosen_differenz_zum_vortag	dosen_erst_differenz_zum_vortag	dosen_zweit_differenz_zum_vortag	dosen_biontech_kumulati
108	2021-04-14	20108105	767050	685188	81862	1489584
109	2021-04-15	20772375	664270	596361	67909	1541762
110	2021-04-16	21332342	559967	490074	69893	1583106

3 rows × 24 columns

In [8]:
```
# Drop unnecessary columns
# No analysis of indication planned:
cols_to_drop = vaccinations.columns[vaccinations.columns.str.contains('indikation_')]
vaccinations.drop(columns=cols_to_drop, inplace=True)
```

In [9]:
```
# Convert datatype of date column
vaccinations.iloc[ : , [0]] = vaccinations.iloc[ : , [0]].apply(pd.to_datetime)
```

In [10]:
```
vaccinations.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 111 entries, 0 to 110
Data columns (total 12 columns):
 #   Column                          Non-Null Count  Dtype         
---  ------                          --------------  -----         
 0   date                            111 non-null    datetime64[ns]
 1   dosen_kumulativ                 111 non-null    int64         
 2   dosen_differenz_zum_vortag      111 non-null    int64         
 3   dosen_erst_differenz_zum_vortag 111 non-null    int64         
 4   dosen_zweit_differenz_zum_vortag 111 non-null    int64         
 5   dosen_biontech_kumulativ        111 non-null    int64         
 6   dosen_moderna_kumulativ         111 non-null    int64         
 7   dosen_astrazeneca_kumulativ     111 non-null    int64         
 8   personen_erst_kumulativ         111 non-null    int64         
 9   personen_voll_kumulativ         111 non-null    int64         
 10  impf_quote_erst                 111 non-null    float64       
 11  impf_quote_voll                 111 non-null    float64       
```

```
dtypes: datetime64[ns](1), float64(2), int64(9)
memory usage: 10.5 KB
```

Last Update

Often the data is not updated on weekends.

In [11]:
```
last_update = vaccinations.loc[vaccinations.index[-1], "date"].strftime('%Y-%m-%d')
```

Doses Used

In [12]:
```
doses = vaccinations.loc[ : , ['date', 'dosen_differenz_zum_vortag']]
doses.columns = ['date', 'doses used']
```

Doses Daily

In [13]:
```
doses_daily = doses.set_index('date', inplace=False)
doses_daily.tail(1)
```

Out[13]:

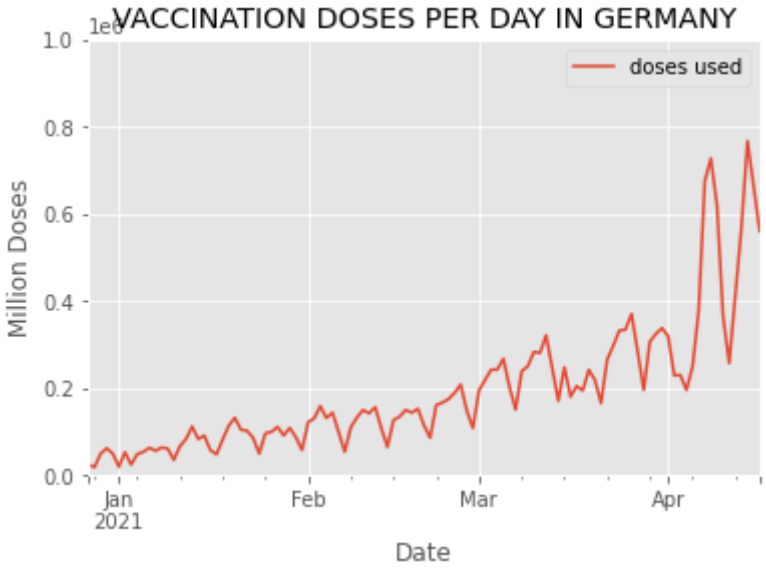
	doses used
date	
2021-04-16	559967

In [14]:
```
# What is the highest number of doses used in a day?
max_doses_daily = max(doses_daily['doses used'])
max_doses_daily
```

Out[14]:
```
767050
```

In [15]:
```
doses_daily.plot(
    ylim=(0,math.ceil(max_doses_daily / 10**6) * 10**6),
    xlabel='Date',
    ylabel='Million Doses',
    title='VACCINATION DOSES PER DAY IN GERMANY')
```

Out[15]:
```
<AxesSubplot:title={'center':'VACCINATION DOSES PER DAY IN GERMANY'}, xlabel='Date', ylabel='Million Doses'>
```



Doses per Week

In [16]:
```
# W-Mon in order to start the week on a Monday, see:
# https://pandas.pydata.org/pandas-docs/stable/user_guide/timeseries.html#anchored-offsets
doses_weekly = doses.groupby(pd.Grouper(key='date',freq='W-Mon')).sum()
doses_weekly.tail()
```

Out[16]:

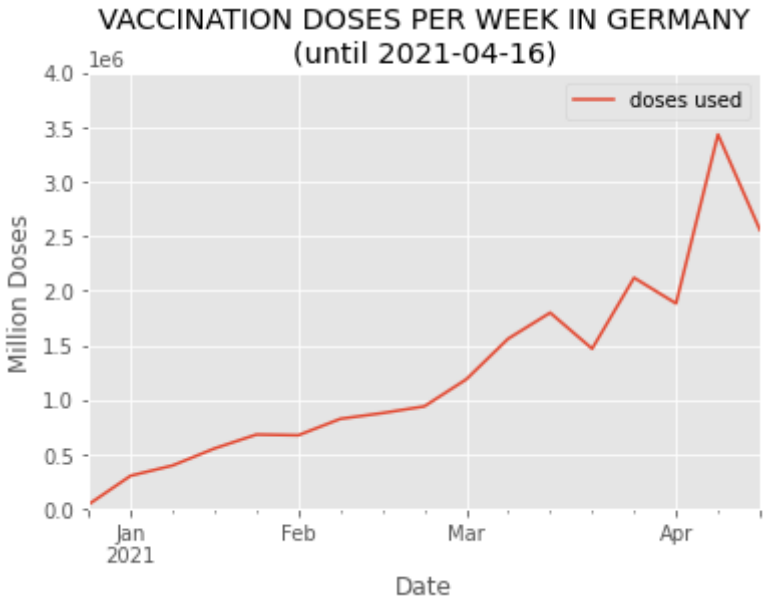
date	doses used
2021-03-22	1470258
2021-03-29	2120416
2021-04-05	1885101
2021-04-12	3433889
2021-04-19	2558788

In [17]:
```
# What is the highest number of doses used in a week?
max_doses_weekly = max(doses_weekly['doses used'])
max_doses_weekly
```

Out[17]: 3433889

In [18]:
```
doses_weekly.plot(
    ylim=(0,math.ceil(max_doses_weekly / 10**6) * 10**6),
    xlabel='Date',
    ylabel='Million Doses',
    title=f"VACCINATION DOSES PER WEEK IN GERMANY\n(until {last_update})")
```

Out[18]: <AxesSubplot:title={'center':'VACCINATION DOSES PER WEEK IN GERMANY\n(until 2021-04-16)'}, xlabel='Date', ylabel='Million Doses'>



Doses per Month

In [19]:
```
# M = month end frequency
doses_monthly = doses.groupby(pd.Grouper(key='date',freq='M')).sum()
doses_monthly.tail()
```

Out[19]:

date	doses used
2020-12-31	203328
2021-01-31	2338412

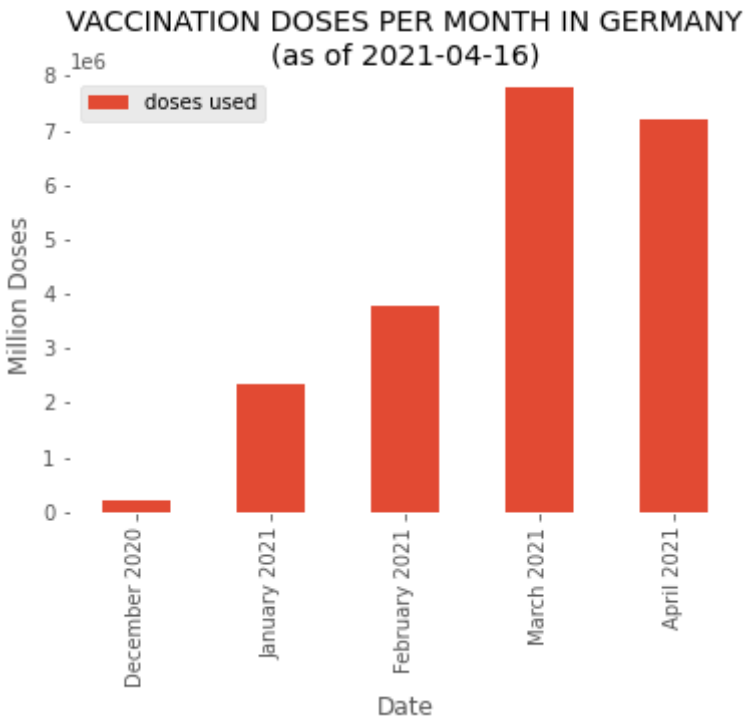
date	doses used
2021-02-28	3765226
2021-03-31	7809576
2021-04-30	7215800

In [20]:
```
max_doses_monthly = max(doses_monthly['doses used'])
max_doses_monthly
doses_monthly['month'] = doses_monthly.index.strftime('%B')
doses_monthly['year'] =  doses_monthly.index.strftime('%Y')
doses_monthly['label'] = doses_monthly['month'] + ' ' + doses_monthly['year']
doses_monthly.drop(columns=['month', 'year'], inplace=True)
doses_monthly.set_index('label', inplace=True)
doses_monthly.tail(6)
```

Out[20]:

label	doses used
December 2020	203328
January 2021	2338412
February 2021	3765226
March 2021	7809576
April 2021	7215800

In [21]:
```
monthly_plot = doses_monthly.plot.bar(
    ylim=(0,math.ceil(max_doses_monthly / 10**6) * 10**6),
    xlabel='Date',
    ylabel='Million Doses',
    title=f"VACCINATION DOSES PER MONTH IN GERMANY\n(as of {last_update})")
monthly_plot.set_facecolor('white')
```



In [22]:
```
fig = monthly_plot.get_figure()
fig.savefig('img/monthly_doses_germany.png')
```

Vaccination Campaign Progress

In [23]:
```
doses_cumulative = vaccinations.loc[ : , ['date', 'personen_erst_kumulativ', 'personen_voll_kumulativ']]
doses_cumulative.set_index('date', inplace=True)
doses_cumulative.head(3)
```

Out[23]:

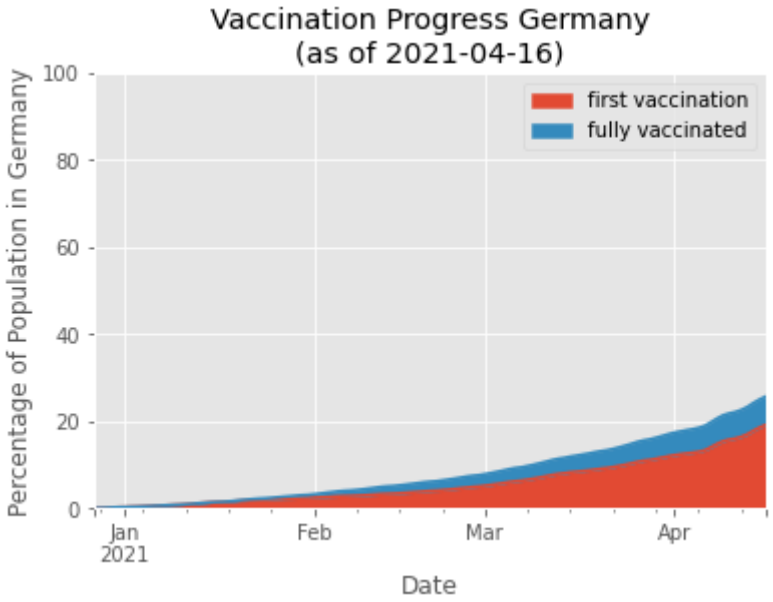
date	personen_erst_kumulativ	personen_voll_kumulativ
2020-12-27	24159	137
2020-12-28	42542	137
2020-12-29	91084	726

In [24]:
```
population_germany = 83_200_000
# Calculate new fields
doses_cumulative['first vaccination'] = round(
    doses_cumulative['personen_erst_kumulativ'] * 100 / population_germany,
    2)
doses_cumulative['fully vaccinated'] = round(
    doses_cumulative['personen_voll_kumulativ'] * 100 / population_germany,
    2)
doses_cumulative.drop(columns=['personen_erst_kumulativ','personen_voll_kumulativ'], inplace=True)
doses_cumulative.tail(3)
```

Out[24]:

date	first vaccination	fully vaccinated
2021-04-14	17.81	6.36
2021-04-15	18.53	6.44
2021-04-16	19.12	6.52

In [25]:
```
doses_area_plot = doses_cumulative.plot.area(
    ylim=(0,100),
    xlabel='Date',
    ylabel='Percentage of Population in Germany',
    title=f"Vaccination Progress Germany\n(as of {last_update})")
```



In [26]:
```
fig = doses_area_plot.get_figure()
fig.savefig('img/vaccinations_germany_area_plot.png')
```

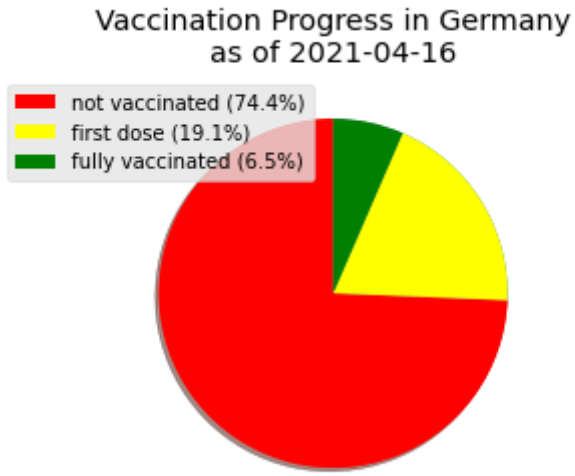
As of Today

In [27]:
```
# get the last line of the data
current_state = doses_cumulative.iloc[-1]
current_state
```

Out[27]:
```
first vaccination    19.12
fully vaccinated      6.52
Name: 2021-04-16 00:00:00, dtype: float64
```

In [28]:
```
percentage_not_vacc = 100 - current_state['first vaccination'] - current_state['fully vaccinated']
labels = [f"not vaccinated ({round(percentage_not_vacc, 1)}%)",
          f"first dose ({round(current_state['first vaccination'],1)}%)",
          f"fully vaccinated ({round(current_state['fully vaccinated'],1)}%)"]
colors = ['red', 'yellow', 'green']
sizes = [percentage_not_vacc,
         current_state['first vaccination'],
         current_state['fully vaccinated']]
fig1, ax1 = plt.subplots()
ax1.pie(sizes, shadow=True, startangle=90)
```

```
ax1.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
patches, texts = plt.pie(sizes, colors=colors, startangle=90)
plt.legend(patches, labels, loc="best")
plt.title(f"Vaccination Progress in Germany\nas of {last_update}")
plt.show()
```



Vaccines in Use

In [29]:
```
vaccine_use = vaccinations.loc[ : , ['date', 'dosen_biontech_kumulativ', 'dosen_moderna_kumulativ', 'dosen_astrazeneca_
# Rename columns
vaccine_use.columns = ['date', 'Biontech', 'Moderna', 'AstraZeneca']
# make 'date' an index
vaccine_use.set_index('date', inplace=True)
vaccine_use.tail(3)
```

Out[29]:

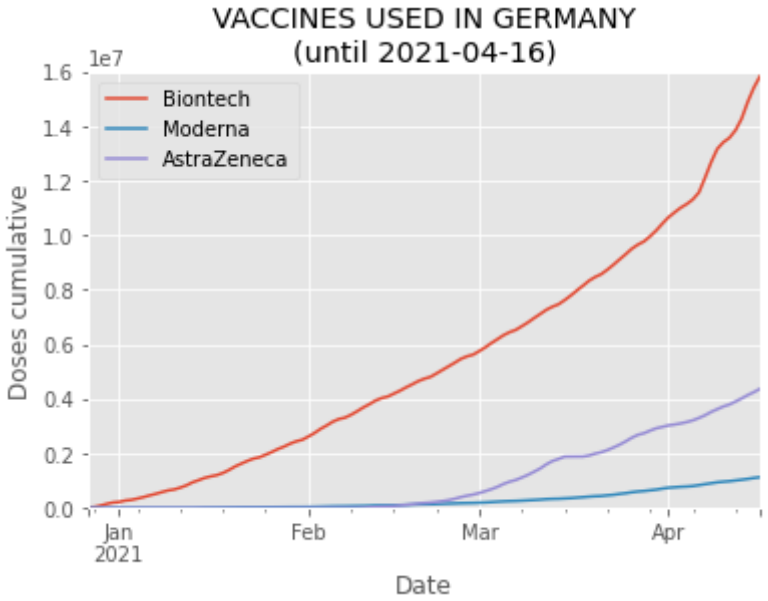
date	Biontech	Moderna	AstraZeneca
2021-04-14	14895847	1068764	4143494
2021-04-15	15417625	1103466	4251284
2021-04-16	15831067	1130287	4370988

In [30]:
```
# To get the maximum for the y axis, round the highest
```

```
# number of doses up to the next million
max_doses = math.ceil(max(vaccine_use.iloc[-1]) / 10**6) * 10**6
max_doses
```

Out[30]: 16000000

In [31]:
```
vaccines_used = vaccine_use.plot(
    ylim=(0,max_doses),
    xlabel='Date',
    ylabel='Doses cumulative',
    title=f"VACCINES USED IN GERMANY\n(until {last_update})")
```



In [32]:
```
fig = vaccines_used.get_figure()
fig.savefig('img/vaccines_used_in_germany.png')
```