```
import pandas as pd
df = pd.read_csv('sample_data/dirty_data.csv')
```

```
df.head()
```

	date	station	PRCP	SNOW	SNWD	TMAX	TMIN	TOBS	WESF	inclement
0	2018-01-01T00:00:00	?	0.0	0.0	-inf	5505.0	-40.0	NaN	NaN	
1	2018-01-01T00:00:00	?	0.0	0.0	-inf	5505.0	-40.0	NaN	NaN	
2	2018-01-01T00:00:00	?	0.0	0.0	-inf	5505.0	-40.0	NaN	NaN	
3	2018-01-02T00:00:00	GHCND:USC00280907	0.0	0.0	-inf	-8.3	-16.1	-12.2	NaN	
4	2018-01-03T00:00:00	GHCND:USC00280907	0.0	0.0	-inf	-4.4	-13.9	-13.3	NaN	

```
df.describe()
```

```
/usr/local/lib/python3.10/dist-packages/numpy/lib/function_base.py:4655: RuntimeWarning
  diff_b_a = subtract(b, a)
```

	PRCP	SNOW	SNWD	TMAX	TMIN	TOBS	WESF
count	765.000000	577.000000	577.0	765.000000	765.000000	398.000000	11.000000
mean	5.360392	4.202773	NaN	2649.175294	-15.914379	8.632161	16.290909
std	10.002138	25.086077	NaN	2744.156281	24.242849	9.815054	9.489832
min	0.000000	0.000000	-inf	-11.700000	-40.000000	-16.100000	1.800000
25%	0.000000	0.000000	NaN	13.300000	-40.000000	0.150000	8.600000
50%	0.000000	0.000000	NaN	32.800000	-11.100000	8.300000	19.300000
75%	5.800000	0.000000	NaN	5505.000000	6.700000	18.300000	24.900000
max	61.700000	229.000000	inf	5505.000000	23.900000	26.100000	28.700000

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 765 entries, 0 to 764
Data columns (total 10 columns):
 #   Column             Non-Null Count  Dtype
```

```
---  ------             --------------  -----
 0   date               765 non-null    object
 1   station            765 non-null    object
 2   PRCP               765 non-null    float64
 3   SNOW               577 non-null    float64
 4   SNWD               577 non-null    float64
 5   TMAX               765 non-null    float64
 6   TMIN               765 non-null    float64
 7   TOBS               398 non-null    float64
 8   WESF               11 non-null     float64
 9   inclement_weather  408 non-null    object
dtypes: float64(7), object(3)
memory usage: 59.9+ KB
```

```
contain_nulls = df[
 df.SNOW.isnull() | df.SNWD.isna()\
 | pd.isnull(df.TOBS) | pd.isna(df.WESF)\
 | df.inclement_weather.isna()
 ]
contain_nulls.shape[0]
```

```
765
```

```
contain_nulls.head(10)
```

	date	station	PRCP	SNOW	SNWD	TMAX	TMIN	TOBS	WESF	inclemen
0	2018-01-01T00:00:00	?	0.0	0.0	-inf	5505.0	-40.0	NaN	NaN	
1	2018-01-01T00:00:00	?	0.0	0.0	-inf	5505.0	-40.0	NaN	NaN	
2	2018-01-01T00:00:00	?	0.0	0.0	-inf	5505.0	-40.0	NaN	NaN	
3	2018-01-02T00:00:00	GHCND:USC00280907	0.0	0.0	-inf	-8.3	-16.1	-12.2	NaN	
4	2018-01-03T00:00:00	GHCND:USC00280907	0.0	0.0	-inf	-4.4	-13.9	-13.3	NaN	
5	2018-01-03T00:00:00	GHCND:USC00280907	0.0	0.0	-inf	-4.4	-13.9	-13.3	NaN	
6	2018-01-03T00:00:00	GHCND:USC00280907	0.0	0.0	-inf	-4.4	-13.9	-13.3	NaN	
7	2018-01-04T00:00:00	?	20.6	229.0	inf	5505.0	-40.0	NaN	19.3	
8	2018-01-04T00:00:00	?	20.6	229.0	inf	5505.0	-40.0	NaN	19.3	
9	2018-01-05T00:00:00	?	0.3	NaN	NaN	5505.0	-40.0	NaN	NaN	

```
df[df.inclement_weather == 'NaN'].shape[0]
```

```
0
```

```
import numpy as np
df[df.inclement_weather == np.nan].shape[0]
```

```
0
```

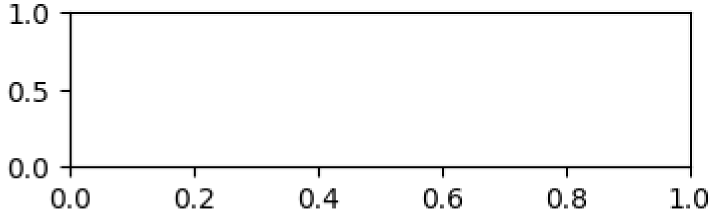
```
df[df.inclement_weather.isna()].shape[0]

```

```
357
```

```
df[df.SNWD.isin([-np.inf, np.inf])].shape[0]
```

```
577
```

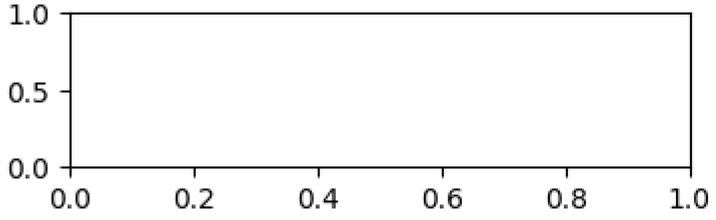


```python
import numpy as np
def get_inf_count(df):
 """Find the number of inf/-inf values per column in the dataframe"""
 return {
 col : df[df[col].isin([np.inf, -np.inf])].shape[0] for col in df.columns
 }
get_inf_count(df)
```

```
{'date': 0,
 'station': 0,
 'PRCP': 0,
 'SNOW': 0,
 'SNWD': 577,
 'TMAX': 0,
 'TMIN': 0,
 'TOBS': 0,
 'WESF': 0,
 'inclement_weather': 0}
```

```python
pd.DataFrame({
 'np.inf Snow Depth': df[df.SNWD == np.inf].SNOW.describe(),
 '-np.inf Snow Depth': df[df.SNWD == -np.inf].SNOW.describe()
}).T
```

	count	mean	std	min	25%	50%	75%	max
np.inf Snow Depth	24.0	101.041667	74.498018	13.0	25.0	120.5	152.0	229.0
-np.inf Snow Depth	553.0	0.000000	0.000000	0.0	0.0	0.0	0.0	0.0



```python
df.describe(include='object')
```

	date	station	inclement_weather
count	765	765	408
unique	324	2	2
top	2018-07-05T00:00:00	GHCND:USC00280907	False
freq	8	398	384

```
df[df.duplicated()].shape[0]
```

```
284
```

```
df[df.duplicated(keep=False)].shape[0]
```

```
482
```

```
df[df.duplicated(['date', 'station'])].shape[0]
```

```
284
```

```
df[df.duplicated()].head()
```

	date	station	PRCP	SNOW	SNWD	TMAX	TMIN	TOBS	WESF	inclemen
1	2018-01-01T00:00:00	?	0.0	0.0	-inf	5505.0	-40.0	NaN	NaN	
2	2018-01-01T00:00:00	?	0.0	0.0	-inf	5505.0	-40.0	NaN	NaN	
5	2018-01-03T00:00:00	GHCND:USC00280907	0.0	0.0	-inf	-4.4	-13.9	-13.3	NaN	
6	2018-01-03T00:00:00	GHCND:USC00280907	0.0	0.0	-inf	-4.4	-13.9	-13.3	NaN	
8	2018-01-04T00:00:00	?	20.6	229.0	inf	5505.0	-40.0	NaN	19.3	

```
df[df.WESF.notna()].station.unique()
```

```
array(['?'], dtype=object)
```

```
# save this information for later
station_qm_wesf = df[df.station == '?'].WESF
# sort ? to the bottom
df.sort_values('station', ascending=False, inplace=True)
# drop duplicates based on the date column keeping the first occurrence
# which will be the valid station if it has data
df_deduped = df.drop_duplicates('date').drop(
 # remove the station column because we are done with it
 # and WESF because we need to replace it later
 columns=['station', 'WESF']
 ).sort_values('date').assign( # sort by the date
 # add back the WESF column which will be properly matched because of the index
 WESF=station_qm_wesf
)
df_deduped.shape

```

```
(324, 9)
```

```
df_deduped.head()
```

	date	PRCP	SNOW	SNWD	TMAX	TMIN	TOBS	inclement_weather	WESF
0	2018-01-01T00:00:00	0.0	0.0	-inf	5505.0	-40.0	NaN	NaN	NaN
3	2018-01-02T00:00:00	0.0	0.0	-inf	-8.3	-16.1	-12.2	False	NaN
6	2018-01-03T00:00:00	0.0	0.0	-inf	-4.4	-13.9	-13.3	False	NaN
8	2018-01-04T00:00:00	20.6	229.0	inf	5505.0	-40.0	NaN	True	19.3
11	2018-01-05T00:00:00	14.2	127.0	inf	-4.4	-13.9	-13.9	True	NaN

```
df_deduped.dropna().shape
```

```
(0, 9)
```

```
df_deduped.dropna(how='all').shape
```

```
(324, 9)
```

```
df_deduped.dropna(
 how='all', subset=['inclement_weather', 'SNOW', 'SNWD']
).shape
```

```
(293, 9)
```

```
df_deduped.dropna(axis='columns', thresh=df_deduped.shape[0]*.75).columns
```

```
Index(['date', 'PRCP', 'SNOW', 'SNWD', 'TMAX', 'TMIN', 'TOBS',
       'inclement_weather'],
      dtype='object')
```

```
df_deduped.loc[:,'WESF'].fillna(0, inplace=True)
df_deduped.head()
```

	date	PRCP	SNOW	SNWD	TMAX	TMIN	TOBS	inclement_weather	WESF
0	2018-01-01T00:00:00	0.0	0.0	-inf	5505.0	-40.0	NaN	NaN	0.0
3	2018-01-02T00:00:00	0.0	0.0	-inf	-8.3	-16.1	-12.2	False	0.0
6	2018-01-03T00:00:00	0.0	0.0	-inf	-4.4	-13.9	-13.3	False	0.0
8	2018-01-04T00:00:00	20.6	229.0	inf	5505.0	-40.0	NaN	True	19.3
11	2018-01-05T00:00:00	14.2	127.0	inf	-4.4	-13.9	-13.9	True	0.0

```
df_deduped.assign(
 TMAX=lambda x: x.TMAX.replace(5505, np.nan).fillna(method='ffill'),
 TMIN=lambda x: x.TMIN.replace(-40, np.nan).fillna(method='ffill')
).head()
```

	date	PRCP	SNOW	SNWD	TMAX	TMIN	TOBS	inclement_weather	WESF
0	2018-01-01T00:00:00	0.0	0.0	-inf	NaN	NaN	NaN	NaN	0.0
3	2018-01-02T00:00:00	0.0	0.0	-inf	-8.3	-16.1	-12.2	False	0.0
6	2018-01-03T00:00:00	0.0	0.0	-inf	-4.4	-13.9	-13.3	False	0.0
8	2018-01-04T00:00:00	20.6	229.0	inf	-4.4	-13.9	NaN	True	19.3
11	2018-01-05T00:00:00	14.2	127.0	inf	-4.4	-13.9	-13.9	True	0.0

```
df_deduped.assign(
 SNWD=lambda x: np.nan_to_num(x.SNWD)
).head()
```

	date	PRCP	SNOW	SNWD	TMAX	TMIN	TOBS	inclement_weather	WESF
0	2018-01-01T00:00:00	0.0	0.0	-1.797693e+308	5505.0	-40.0	NaN	NaN	0.0
3	2018-01-02T00:00:00	0.0	0.0	-1.797693e+308	-8.3	-16.1	-12.2	False	0.0
6	2018-01-03T00:00:00	0.0	0.0	-1.797693e+308	-4.4	-13.9	-13.3	False	0.0
8	2018-01-04T00:00:00	20.6	229.0	1.797693e+308	5505.0	-40.0	NaN	True	19.3
11	2018-01-05T00:00:00	14.2	127.0	1.797693e+308	-4.4	-13.9	-13.9	True	0.0

```
df_deduped.assign(
 TMAX=lambda x: x.TMAX.replace(5505, np.nan).fillna(x.TMAX.median()),
 TMIN=lambda x: x.TMIN.replace(-40, np.nan).fillna(x.TMIN.median()),
 # average of TMAX and TMIN
 TOBS=lambda x: x.TOBS.fillna((x.TMAX + x.TMIN) / 2)
).head()

```

	date	PRCP	SNOW	SNWD	TMAX	TMIN	TOBS	inclement_weather	WESF
0	2018-01-01T00:00:00	0.0	0.0	-inf	22.8	0.0	11.4	NaN	0.0
3	2018-01-02T00:00:00	0.0	0.0	-inf	-8.3	-16.1	-12.2	False	0.0
6	2018-01-03T00:00:00	0.0	0.0	-inf	-4.4	-13.9	-13.3	False	0.0
8	2018-01-04T00:00:00	20.6	229.0	inf	22.8	0.0	11.4	True	19.3
11	2018-01-05T00:00:00	14.2	127.0	inf	-4.4	-13.9	-13.9	True	0.0

```
df_deduped.assign(
 # make TMAX and TMIN NaN where appropriate
 TMAX=lambda x: x.TMAX.replace(5505, np.nan),
 TMIN=lambda x: x.TMIN.replace(-40, np.nan)
).set_index('date').apply(
 # rolling calculations will be covered in chapter 4, this is a rolling 7 day median
 # we set min_periods (# of periods required for calculation) to 0 so we always get a re
 lambda x: x.fillna(x.rolling(7, min_periods=0).median())
).head(10)
```

	PRCP	SNOW	SNWD	TMAX	TMIN	TOBS	inclement_weather	WESF
date								
2018-01-01T00:00:00	0.0	0.0	-inf	NaN	NaN	NaN	NaN	0.0
2018-01-02T00:00:00	0.0	0.0	-inf	-8.30	-16.1	-12.20	False	0.0
2018-01-03T00:00:00	0.0	0.0	-inf	-4.40	-13.9	-13.30	False	0.0
2018-01-04T00:00:00	20.6	229.0	inf	-6.35	-15.0	-12.75	True	19.3
2018-01-05T00:00:00	14.2	127.0	inf	-4.40	-13.9	-13.90	True	0.0
2018-01-06T00:00:00	0.0	0.0	-inf	-10.00	-15.6	-15.00	False	0.0
2018-01-07T00:00:00	0.0	0.0	-inf	-11.70	-17.2	-16.10	False	0.0
2018-01-08T00:00:00	0.0	0.0	-inf	-7.80	-16.7	-8.30	False	0.0

```
df_deduped.assign(
 # make TMAX and TMIN NaN where appropriate
 TMAX=lambda x: x.TMAX.replace(5505, np.nan),
 TMIN=lambda x: x.TMIN.replace(-40, np.nan),
 date=lambda x: pd.to_datetime(x.date)
).set_index('date').reindex(
 pd.date_range('2018-01-01', '2018-12-31', freq='D')
).apply(
 lambda x: x.interpolate()
).head(10)
```