Project: Querying and Filtering Pokemon data

This project will help you practice your pandas querying and filtering skills. Let's begin!

Photo by Mikel (https://unsplash.com/@mykelgran?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText) on Unsplash (https://unsplash.com/s/photos/pokemon?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText).

Task 0 - Setup

There isn't much to do here, we'll provide the required imports and the read the pokemon CSV we'll be working with.

In [1]:
```
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df = pd.read_csv("pokemon.csv")
```

In [3]:
```
df.head()
```

Out[3]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Leg
0	1	Bulbasaur	Grass	Poison	318	45	49	49	65	65	45	1	
1	2	Ivysaur	Grass	Poison	405	60	62	63	80	80	60	1	
2	3	Venusaur	Grass	Poison	525	80	82	83	100	100	80	1	
3	4	Charmander	Fire	NaN	309	39	52	43	60	50	65	1	
4	5	Charmeleon	Fire	NaN	405	58	64	58	80	65	80	1	

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 721 entries, 0 to 720
Data columns (total 13 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   #           721 non-null    int64 
 1   Name        721 non-null    object
 2   Type 1      721 non-null    object
 3   Type 2      359 non-null    object
 4   Total       721 non-null    int64 
 5   HP          721 non-null    int64 
 6   Attack      721 non-null    int64 
 7   Defense     721 non-null    int64 
 8   Sp. Atk     721 non-null    int64 
 9   Sp. Def     721 non-null    int64 
 10  Speed       721 non-null    int64 
 11  Generation  721 non-null    int64 
 12  Legendary   721 non-null    bool  
dtypes: bool(1), int64(9), object(3)
memory usage: 68.4+ KB
```

In [5]:
```
df.describe()
```

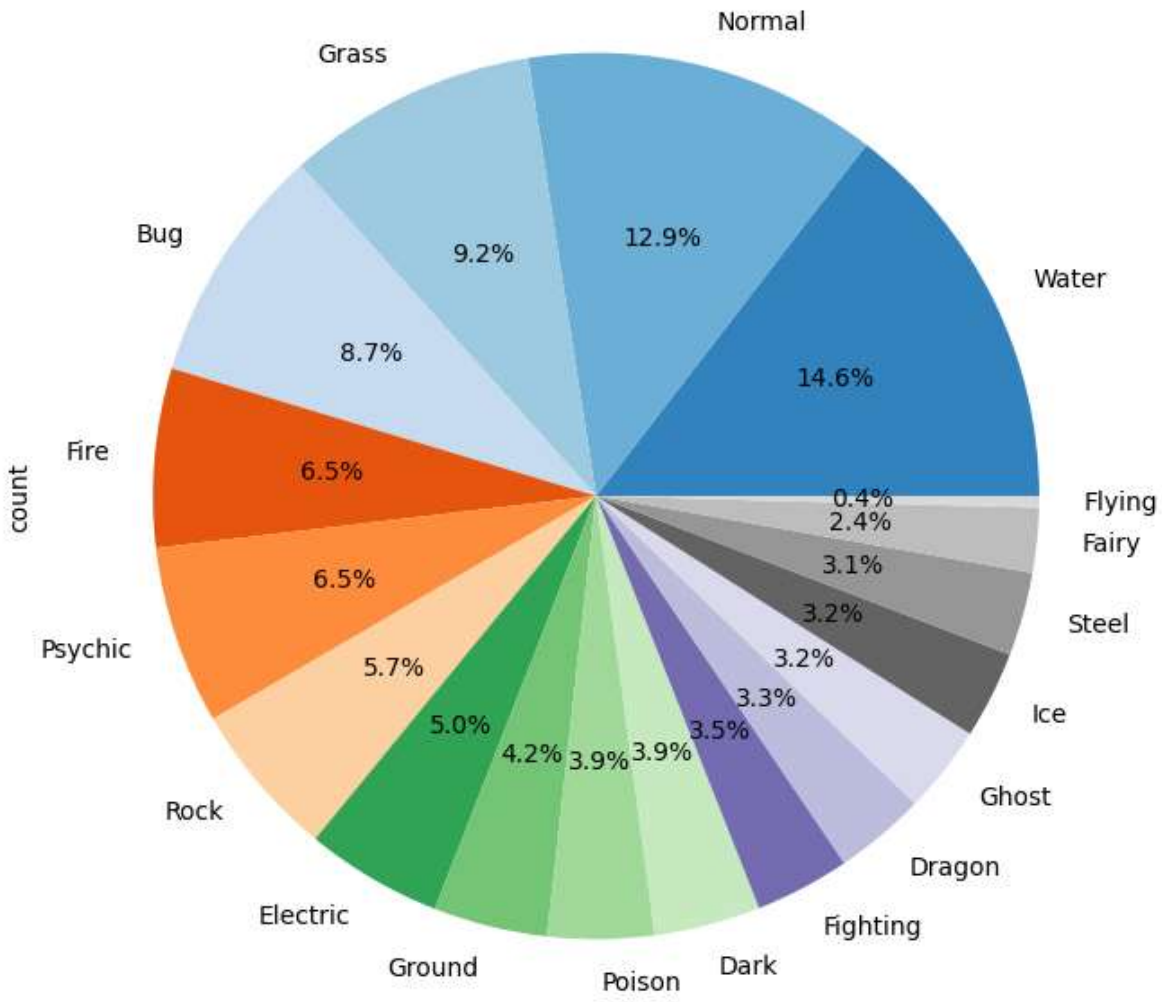
Out[5]:

	#	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	
count	721.00000	721.000000	721.000000	721.000000	721.000000	721.000000	721.000000	721.(
mean	361.00000	417.945908	68.380028	75.124827	70.697642	68.848821	69.180305	65.7
std	208.27906	109.663671	25.848272	29.070335	29.194941	28.898590	26.899364	27.2
min	1.00000	180.000000	1.000000	5.000000	5.000000	10.000000	20.000000	5.(
25%	181.00000	320.000000	50.000000	54.000000	50.000000	45.000000	50.000000	45.(
50%	361.00000	424.000000	65.000000	75.000000	65.000000	65.000000	65.000000	65.(
75%	541.00000	499.000000	80.000000	95.000000	85.000000	90.000000	85.000000	85.(
max	721.00000	720.000000	255.000000	165.000000	230.000000	154.000000	230.000000	160.(

Distribution of Pokemon Types:

In [6]:
```
df['Type 1'].value_counts().plot(kind='pie', autopct='%1.1f%%', cmap='tab20c',
```

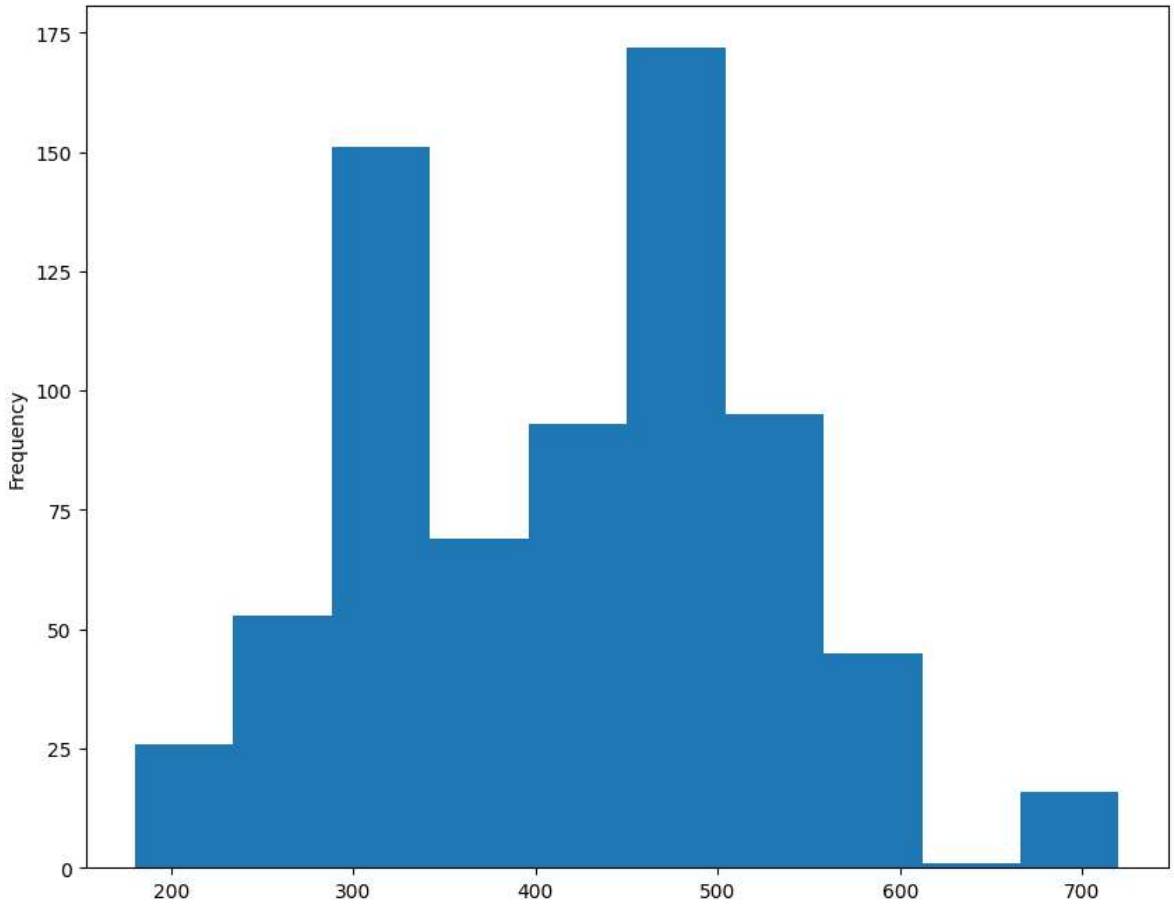
Out[6]:
```
<Axes: ylabel='count'>
```



Distribution of Pokemon Totals:

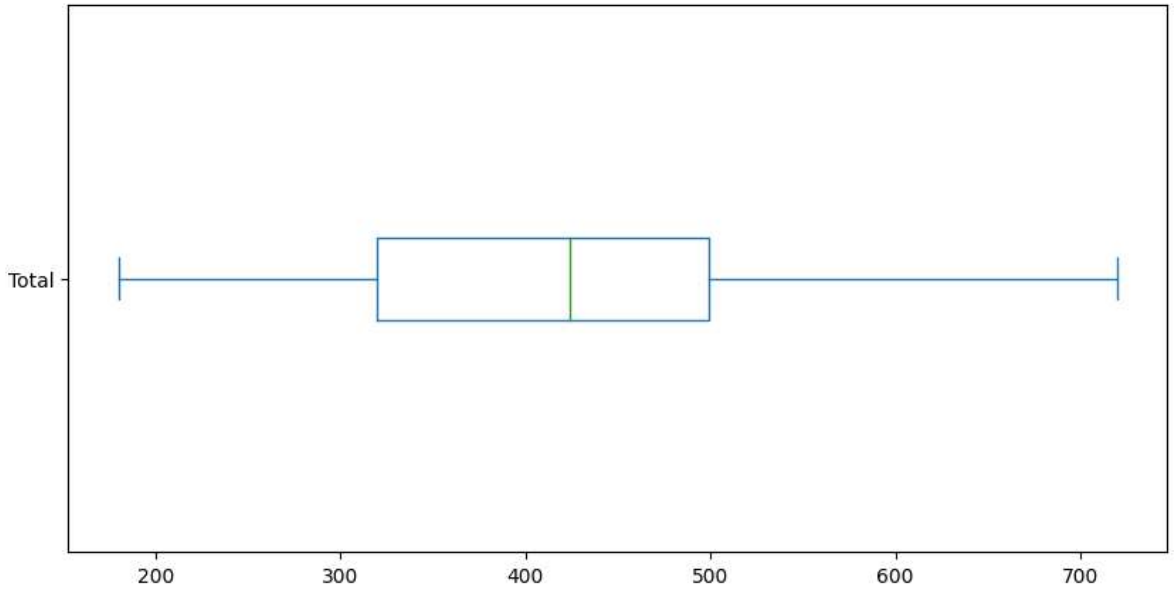
In [7]:
```
df['Total'].plot(kind='hist', figsize=(10, 8))
```

Out[7]: <Axes: ylabel='Frequency'>



In [8]:
```
df['Total'].plot(kind='box', vert=False, figsize=(10, 5))
```

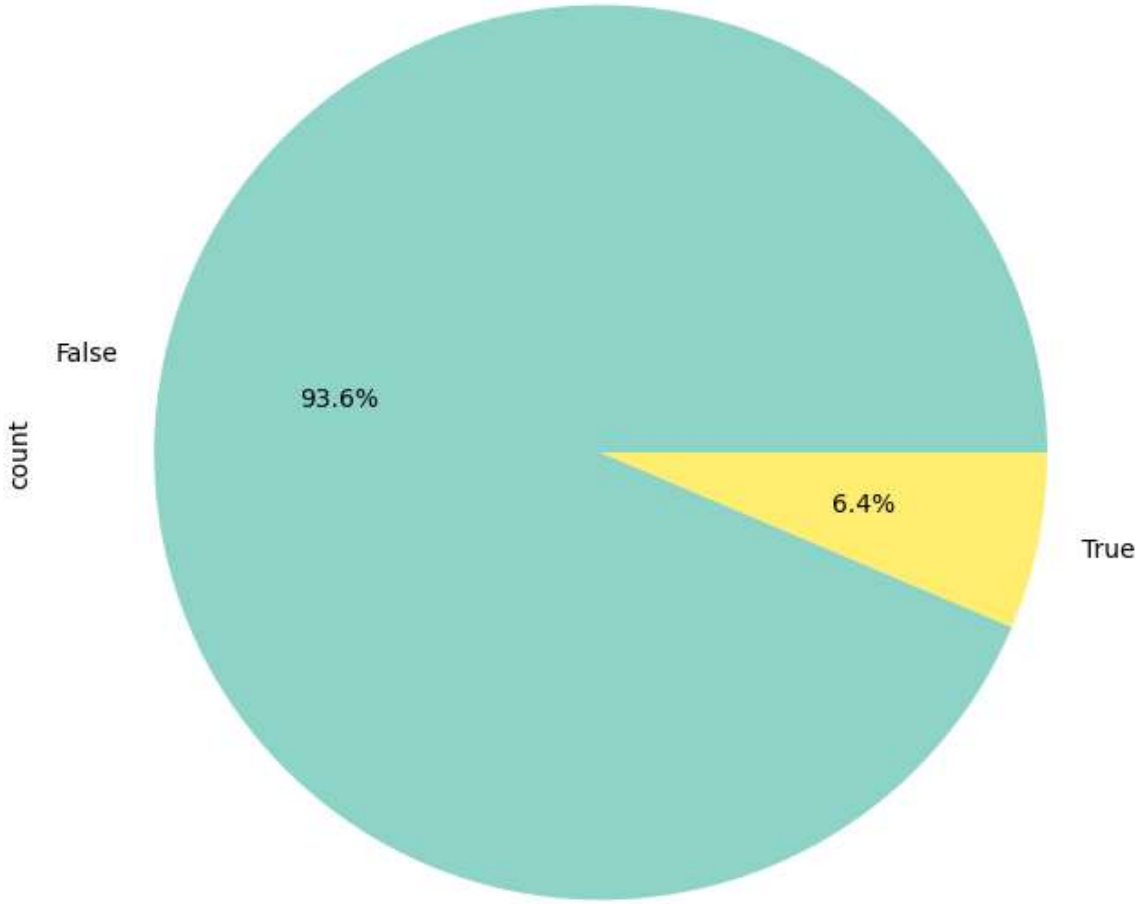
Out[8]: <Axes: >



Distribution of Legendary Pokemons:

In [9]:
```
df['Legendary'].value_counts().plot(kind='pie', autopct='%1.1f%%', cmap='Set3'
```

Out[9]:
```
<Axes: ylabel='count'>
```



Basic filtering

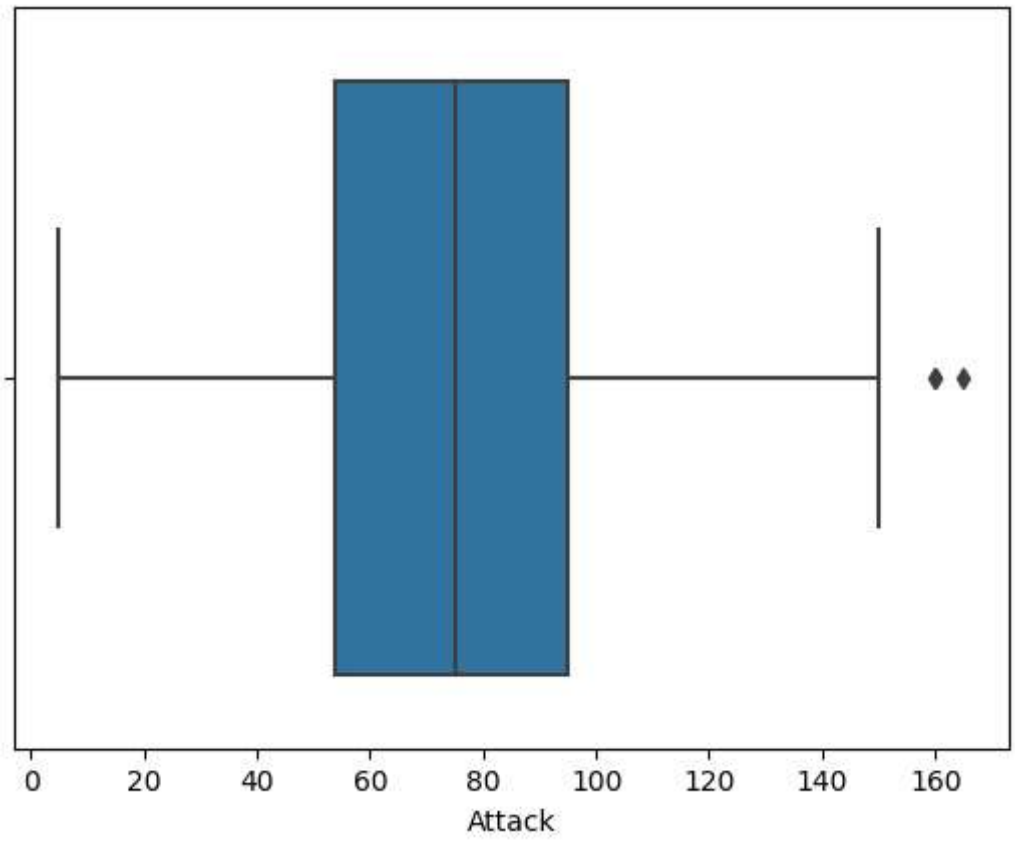
Let's start with a few simple activities regarding filtering.

1. How many Pokemons exist with an `Attack` value greater than 150?

Doing a little bit of visual exploration, we can have a sense of the most "powerful" pokemons (defined by their "Attack" feature). A boxplot is a great way to visualize this:

In [10]:
```
sns.boxplot(data=df, x='Attack')
```

Out[10]:
```
<Axes: xlabel='Attack'>
```



In [12]:
```
df[df["Attack"]>150]
```

Out[12]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation
288	289	Slaking	Normal	NaN	670	150	160	100	95	65	100	3
408	409	Rampardos	Rock	NaN	495	97	165	60	65	50	58	4
485	486	Regigigas	Normal	NaN	670	110	160	110	80	110	100	4

In [13]:
```
df.loc[df["Attack"]>150]
```

Out[13]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation
288	289	Slaking	Normal	NaN	670	150	160	100	95	65	100	3
408	409	Rampardos	Rock	NaN	495	97	165	60	65	50	58	4
485	486	Regigigas	Normal	NaN	670	110	160	110	80	110	100	4

In [14]:
```
df.query("Attack>150")
```

Out[14]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation
288	289	Slaking	Normal	NaN	670	150	160	100	95	65	100	3
408	409	Rampardos	Rock	NaN	495	97	165	60	65	50	58	4
485	486	Regigigas	Normal	NaN	670	110	160	110	80	110	100	4

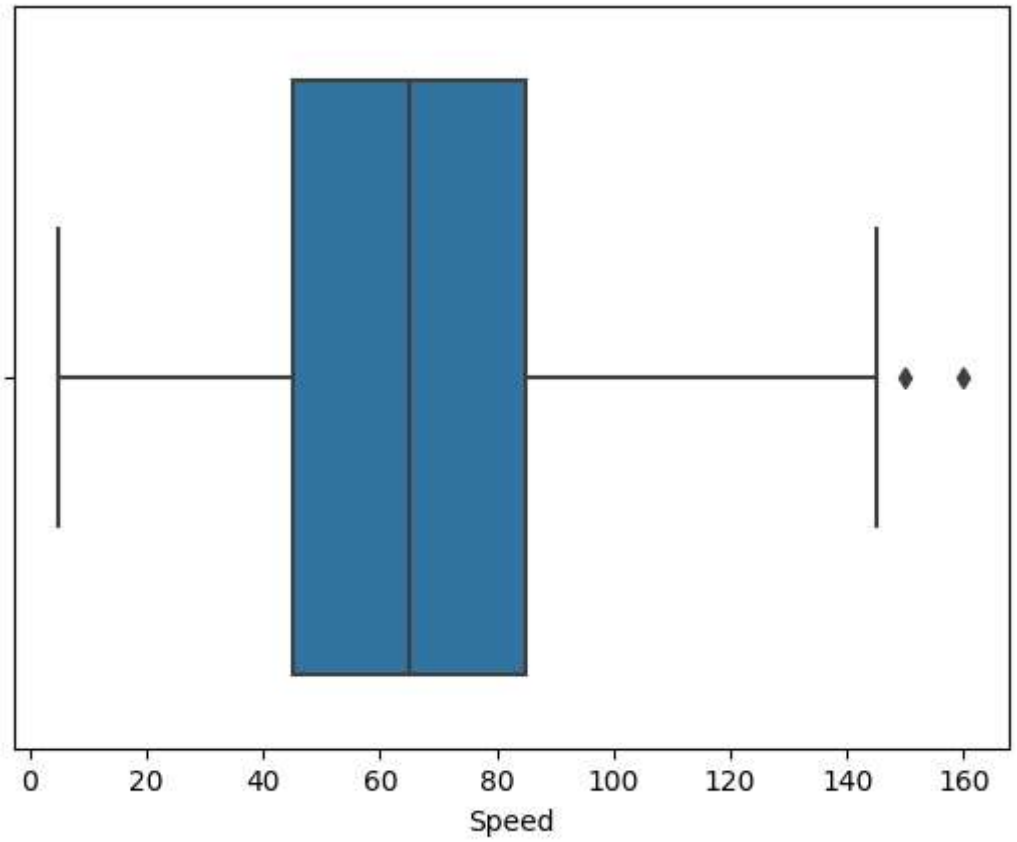
In [24]:
```
(df["Attack"]>150).sum()
```

Out[24]:
```
3
```

2. Select all pokemons with a Speed of `10` or less

In [15]:
```
sns.boxplot(data=df, x='Speed')
```

Out[15]:
```
<Axes: xlabel='Speed'>
```



In [21]:
```
df[df["Speed"]<=10]
```

Out[21]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	L
212	213	Shuckle	Bug	Rock	505	20	10	230	10	230	5	2	
327	328	Trapinch	Ground	NaN	290	45	100	45	45	45	10	3	
437	438	Bonsly	Rock	NaN	290	50	80	95	10	45	10	4	
445	446	Munchlax	Normal	NaN	390	135	85	40	40	85	5	4	
596	597	Ferroseed	Grass	Steel	305	44	50	91	24	86	10	5	

In [20]:
```
df.query("Speed<=10")
```

Out[20]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	L
212	213	Shuckle	Bug	Rock	505	20	10	230	10	230	5	2	
327	328	Trapinch	Ground	NaN	290	45	100	45	45	45	10	3	
437	438	Bonsly	Rock	NaN	290	50	80	95	10	45	10	4	
445	446	Munchlax	Normal	NaN	390	135	85	40	40	85	5	4	
596	597	Ferroseed	Grass	Steel	305	44	50	91	24	86	10	5	

In [23]:
```
(df["Speed"]<=10).sum()
```

Out[23]:
```
5
```

In [17]:
```
df.loc[df["Speed"]<=10]
```

Out[17]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	L
212	213	Shuckle	Bug	Rock	505	20	10	230	10	230	5	2	
327	328	Trapinch	Ground	NaN	290	45	100	45	45	45	10	3	
437	438	Bonsly	Rock	NaN	290	50	80	95	10	45	10	4	
445	446	Munchlax	Normal	NaN	390	135	85	40	40	85	5	4	
596	597	Ferroseed	Grass	Steel	305	44	50	91	24	86	10	5	

In [18]:
```
slow_pokemons_df = df.loc[df["Speed"]<=10]
```

3. How many Pokemons have a `Sp. Def` value of 25 or less?

In [26]:
```
(df["Sp. Def"]<=25).sum()
```

Out[26]: 17

In [28]:
```
df.loc[df["Sp. Def"]<=25].shape
```

Out[28]: (17, 13)

4. Select all the Legendary pokemons

In [29]:
```
legendary_df = df.loc[df["Legendary"]]
legendary_df.head()
```

Out[29]:

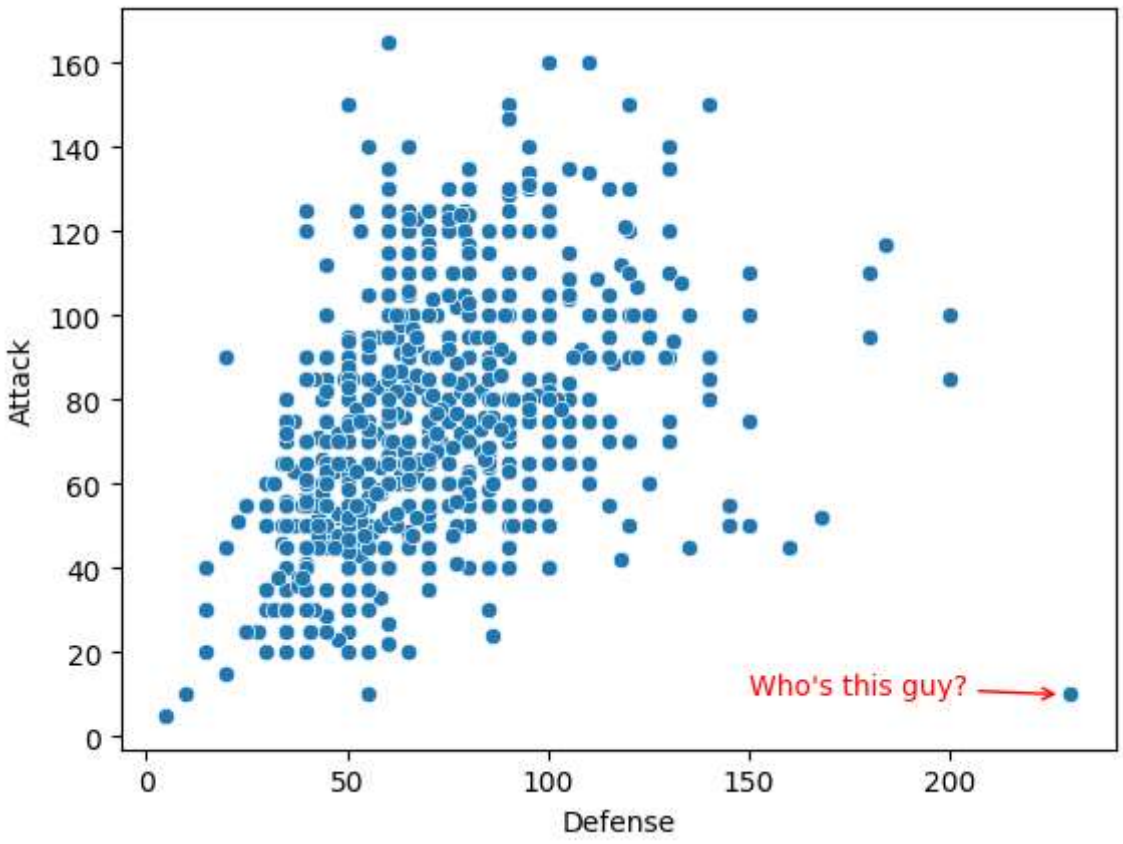
	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation
143	144	Articuno	Ice	Flying	580	90	85	100	95	125	85	1
144	145	Zapdos	Electric	Flying	580	90	90	85	125	90	100	1
145	146	Moltres	Fire	Flying	580	90	100	90	125	85	90	1
149	150	Mewtwo	Psychic	Fighting	680	106	110	90	154	90	130	1
242	243	Raikou	Electric	NaN	580	90	85	75	115	100	115	2

5. Find the outlier

Find the pokemon that is clearly an outlier in terms of Attack / Defense:

In [31]:
```
ax = sns.scatterplot(data=df, x="Defense", y="Attack")
ax.annotate(
    "Who's this guy?", xy=(228, 10), xytext=(150, 10), color='red',
    arrowprops=dict(arrowstyle="->", color='red')
)
```

Out[31]: Text(150, 10, "Who's this guy?")



In [ ]:
```
# your code
```

In [34]:
```
df.sort_values(by="Defense",ascending=False).head()
```

Out[34]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Le
212	213	Shuckle	Bug	Rock	505	20	10	230	10	230	5	2	
376	377	Regirock	Rock	NaN	580	80	100	200	50	100	50	3	
207	208	Steelix	Steel	Ground	510	75	85	200	55	65	30	2	
712	713	Avalugg	Ice	NaN	514	95	117	184	44	46	28	6	
90	91	Cloyster	Water	Ice	525	50	95	180	85	45	70	1	

In [38]:
```
df.sort_values(by=["Defense","Attack"],ascending=[False,True]).head()
```

Out[38]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Le
212	213	Shuckle	Bug	Rock	505	20	10	230	10	230	5	2	
207	208	Steelix	Steel	Ground	510	75	85	200	55	65	30	2	
376	377	Regirock	Rock	NaN	580	80	100	200	50	100	50	3	
712	713	Avalugg	Ice	NaN	514	95	117	184	44	46	28	6	
90	91	Cloyster	Water	Ice	525	50	95	180	85	45	70	1	

In [37]:
```
#df.sort_values(by="Attack").head()
```

Advanced selection

Now let's use boolean operators to create more advanced expressions

6. How many Fire-Flying Pokemons are there?

In [41]:
```
df[(df["Type 1"]=="Fire")&(df["Type 2"]=="Flying")].shape
```

Out[41]: (5, 13)

In [43]:
```
((df["Type 1"]=="Fire")&(df["Type 2"]=="Flying")).sum()
```

Out[43]: 5

In [46]:
```
df.loc[(df["Type  1"]=="Fire")&(df["Type 2"]=="Flying")]
```

Out[46]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	L
5	6	Charizard	Fire	Flying	534	78	84	78	109	85	100	1	
145	146	Moltres	Fire	Flying	580	90	100	90	125	85	90	1	
249	250	Ho-oh	Fire	Flying	680	106	130	90	110	154	90	2	
661	662	Fletchinder	Fire	Flying	382	62	73	55	56	52	84	6	
662	663	Talonflame	Fire	Flying	499	78	81	71	74	69	126	6	

In [47]:
```
df.query("`Type 1`=='Fire' and `Type 2`=='Flying'")
```

Out[47]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	L
5	6	Charizard	Fire	Flying	534	78	84	78	109	85	100	1	
145	146	Moltres	Fire	Flying	580	90	100	90	125	85	90	1	
249	250	Ho-oh	Fire	Flying	680	106	130	90	110	154	90	2	
661	662	Fletchinder	Fire	Flying	382	62	73	55	56	52	84	6	
662	663	Talonflame	Fire	Flying	499	78	81	71	74	69	126	6	

7. How many 'Poison' pokemons are across both types?

In [55]:
```
((df["Type 1"]=="Poison")|(df["Type 2"]=="Poison")).sum()
```

Out[55]: 59

In [51]:
```
df.query("`Type 1`=='Poison' or `Type 2`=='Poison'").shape
```

Out[51]: (59, 13)

In [56]:
```
df[(df["Type 1"]=="Poison")|(df["Type 2"]=="Poison")].shape
```

Out[56]: (59, 13)

8. What pokemon of `Type 1` Ice has the strongest defense?

In [57]:
```
df.loc[df["Type 1"]=="Ice","Defense"].max()
```

Out[57]: 184

In [58]:
```
df.loc[
    (df["Type 1"]=="Ice") &
    (df["Defense"]==df.loc[df["Type 1"]=="Ice","Defense"].max())
]
```

Out[58]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Legen
712	713	Avalugg	Ice	NaN	514	95	117	184	44	46	28	6	

In [59]:
```
df.loc[df["Type 1"]=="Ice"].sort_values("Defense",ascending=False).head()
```

Out[59]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Leg
712	713	Avalugg	Ice	NaN	514	95	117	184	44	46	28	6	
470	471	Glaceon	Ice	NaN	525	65	60	110	130	95	65	4	
143	144	Articuno	Ice	Flying	580	90	85	100	95	125	85	1	
377	378	Regice	Ice	NaN	580	80	50	100	100	200	50	3	
364	365	Walrein	Ice	Water	530	110	80	90	95	90	65	3	

In [61]:
```
df.loc[
    (df["Type 1"]=="Ice") &
    (df["Defense"]==df.loc[df["Type 1"]=="Ice","Defense"].max())
].iloc[0,1]
```

Out[61]:
```
'Avalugg'
```

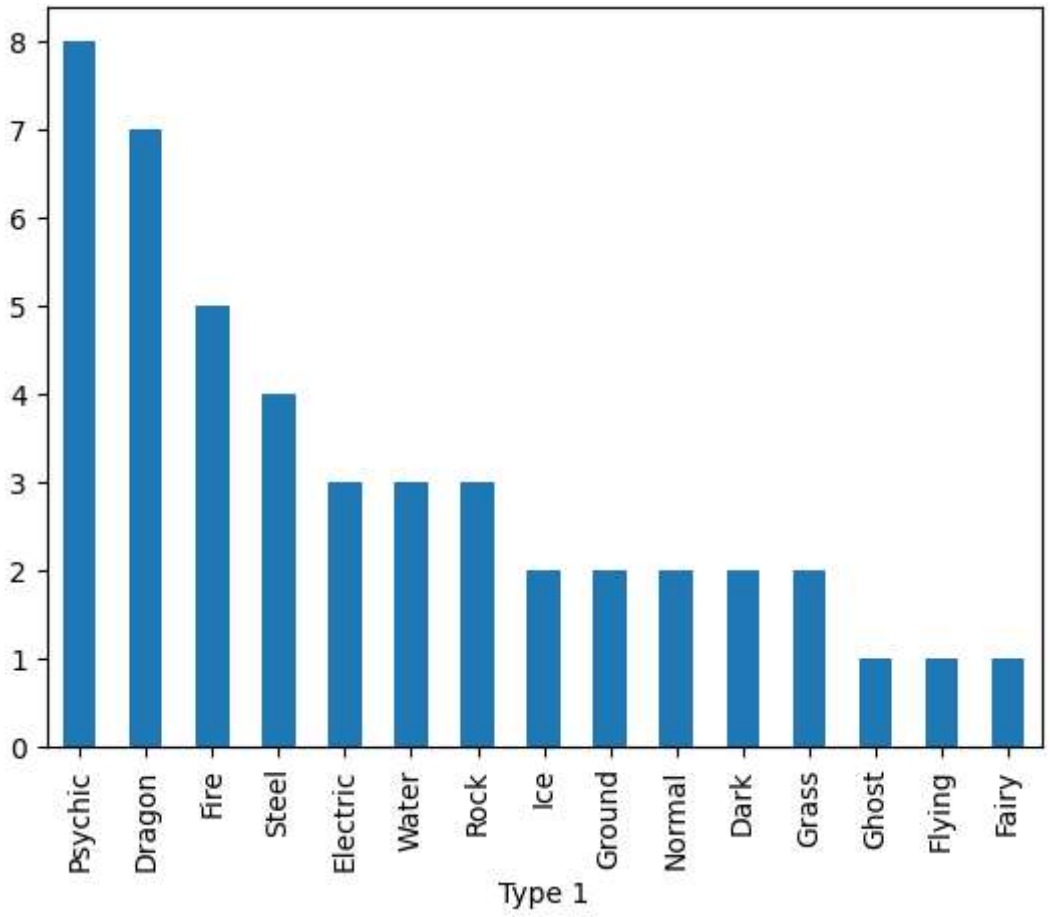
9. What's the most common type of Legendary Pokemons?

In [63]:
```
df.loc[df["Legendary"],"Type 1"].value_counts()
```

Out[63]:
```
Type 1
Psychic     8
Dragon      7
Fire        5
Steel       4
Electric    3
Water       3
Rock        3
Ice         2
Ground      2
Normal      2
Dark        2
Grass       2
Ghost       1
Flying      1
Fairy       1
Name: count, dtype: int64
```

In [64]:
```
df.loc[df["Legendary"],"Type 1"].value_counts().plot(kind="bar")
```

Out[64]:
```
<Axes: xlabel='Type 1'>
```



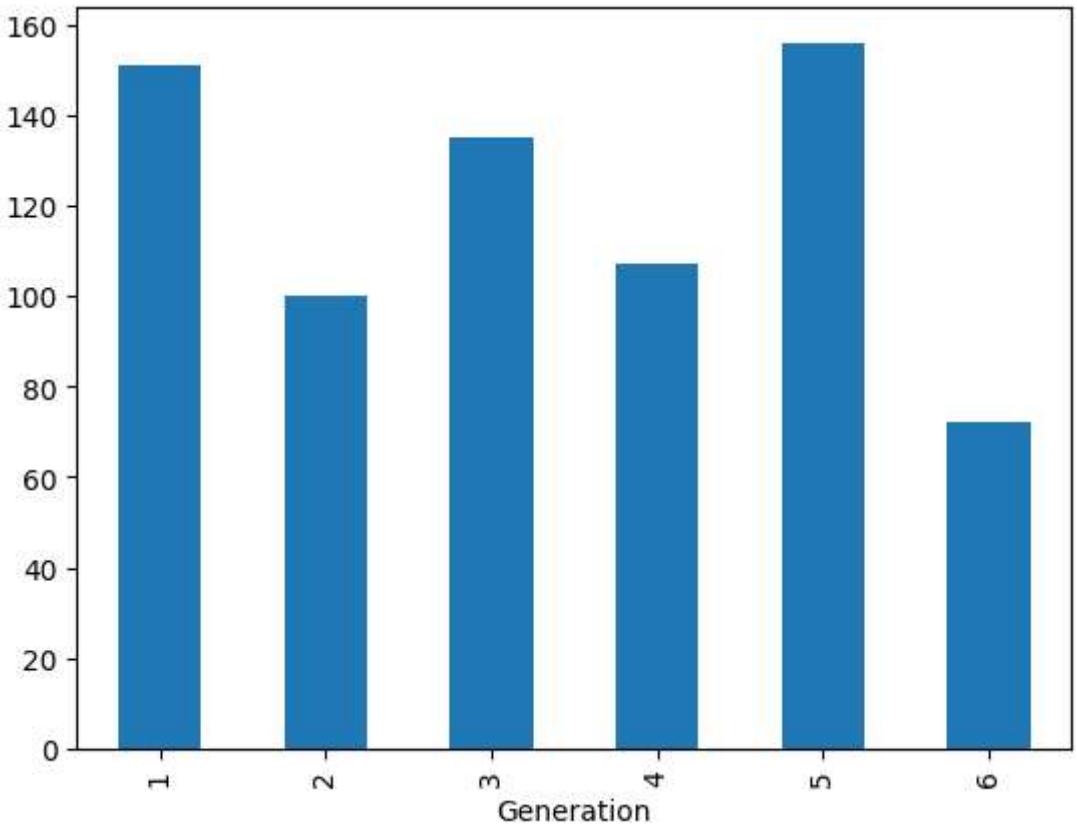
10. What's the most powerful pokemon from the first 3 generations, of type water?

In [66]:
```
df["Generation"].value_counts(sort=False)
```

Out[66]:
```
Generation
1    151
2    100
3    135
4    107
5    156
6     72
Name: count, dtype: int64
```

In [67]:
```
df["Generation"].value_counts(sort=False).plot(kind="bar")
```

Out[67]: <Axes: xlabel='Generation'>



In [65]:
```
df.loc[df["Type 1"]=="Water"]
```

Out[65]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Le
6	7	Squirtle	Water	NaN	314	44	48	65	50	64	43	1	
7	8	Wartortle	Water	NaN	405	59	63	80	65	80	58	1	
8	9	Blastoise	Water	NaN	530	79	83	100	85	105	78	1	
53	54	Psyduck	Water	NaN	320	50	52	48	65	50	55	1	
54	55	Golduck	Water	NaN	500	80	82	78	95	80	85	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	
655	656	Froakie	Water	NaN	314	41	56	40	62	44	71	6	
656	657	Frogadier	Water	NaN	405	54	63	52	83	56	97	6	
657	658	Greninja	Water	Dark	530	72	95	67	103	71	122	6	
691	692	Clauncher	Water	NaN	330	50	53	62	58	63	44	6	
692	693	Clawitzer	Water	NaN	500	71	73	88	120	89	59	6	

105 rows × 13 columns

In [69]:
```
(
    (df["Generation"]==1)|
    (df["Generation"]==2)|
    (df["Generation"]==3)
).sum()
```

Out[69]: 386

In [70]:
```
df["Generation"].isin([1,2,3]).sum()
```

Out[70]: 386

In [73]:
```
df.loc[
    (df["Type 1"]=="Water")&
    df["Generation"].isin([1,2,3])
].sort_values(by="Total",ascending=False).head()
```

Out[73]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	I
381	382	Kyogre	Water	NaN	670	100	100	90	150	140	90	3	
244	245	Suicune	Water	NaN	580	100	75	115	90	115	85	2	
349	350	Milotic	Water	NaN	540	95	60	79	100	125	81	3	
229	230	Kingdra	Water	Dragon	540	75	95	95	95	95	85	2	
129	130	Gyarados	Water	Flying	540	95	125	79	60	100	81	1	

11. What's the most powerful Dragon from the last two generations?

In [74]:
```
df.loc[
    ((df["Type 1"]=="Dragon")|(df["Type 2"]=="Dragon")) &
    df["Generation"].isin([5,6])
].sort_values(by="Total",ascending=False).head()
```

Out[74]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation
643	644	Zekrom	Dragon	Electric	680	100	150	120	120	100	90	5
642	643	Reshiram	Dragon	Fire	680	100	120	100	150	120	90	5
645	646	Kyurem	Dragon	Ice	660	125	130	90	130	90	95	5
634	635	Hydreigon	Dark	Dragon	600	92	105	90	125	90	98	5
705	706	Goodra	Dragon	NaN	600	90	100	70	110	150	80	6

In [75]:
```
df.loc[
    ((df["Type 1"]=="Dragon")|(df["Type 2"]=="Dragon")) &
    (df["Generation"].isin({5,6}))
].sort_values(by="Total",ascending=False).head()
```

Out[75]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation
643	644	Zekrom	Dragon	Electric	680	100	150	120	120	100	90	5
642	643	Reshiram	Dragon	Fire	680	100	120	100	150	120	90	5
645	646	Kyurem	Dragon	Ice	660	125	130	90	130	90	95	5
634	635	Hydreigon	Dark	Dragon	600	92	105	90	125	90	98	5
705	706	Goodra	Dragon	NaN	600	90	100	70	110	150	80	6

In [79]:
```
df.query(
    "(`Type 1`=='Dragon' or `Type 2`=='Dragon') and Generation in [5,6]"
).sort_values(by="Total",ascending=False).head()
```

Out[79]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation
643	644	Zekrom	Dragon	Electric	680	100	150	120	120	100	90	5
642	643	Reshiram	Dragon	Fire	680	100	120	100	150	120	90	5
645	646	Kyurem	Dragon	Ice	660	125	130	90	130	90	95	5
634	635	Hydreigon	Dark	Dragon	600	92	105	90	125	90	98	5
705	706	Goodra	Dragon	NaN	600	90	100	70	110	150	80	6

12. Select most powerful Fire-type pokemons

In [76]:
```
df.loc[
    (df["Attack"]>100)&
    (df["Type 1"]=="Fire")
].head()
```

Out[76]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	L
58	59	Arcanine	Fire	NaN	555	90	110	80	100	80	95	1	
135	136	Flareon	Fire	NaN	525	65	130	60	95	110	65	1	
243	244	Entei	Fire	NaN	580	115	115	85	90	75	100	2	
249	250	Ho-oh	Fire	Flying	680	106	130	90	110	154	90	2	
256	257	Blaziken	Fire	Fighting	530	80	120	70	110	70	80	3	

In [81]:
```
powerful_fire_df = df.query("Attack >100 and `Type 1`=='Fire'")
powerful_fire_df
```

Out[81]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Ger
58	59	Arcanine	Fire	NaN	555	90	110	80	100	80	95	
135	136	Flareon	Fire	NaN	525	65	130	60	95	110	65	
243	244	Entei	Fire	NaN	580	115	115	85	90	75	100	
249	250	Ho-oh	Fire	Flying	680	106	130	90	110	154	90	
256	257	Blaziken	Fire	Fighting	530	80	120	70	110	70	80	
391	392	Infernape	Fire	Fighting	534	76	104	71	104	71	108	
499	500	Emboar	Fire	Fighting	528	110	123	65	100	65	65	
554	555	DarmanitanStandard Mode	Fire	Psychic	480	105	140	55	30	55	95	
720	721	Volcanion	Fire	Water	600	80	110	120	130	90	70	

13. Select all Water-type, Flying-type pokemons

In [83]:
```
water_flying_df = df.query("`Type 1`=='Water' and `Type 2`=='Flying'")
water_flying_df
```

Out[83]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Le
129	130	Gyarados	Water	Flying	540	95	125	79	60	100	81	1	
225	226	Mantine	Water	Flying	465	65	40	70	80	140	70	2	
277	278	Wingull	Water	Flying	270	40	30	30	55	30	85	3	
278	279	Pelipper	Water	Flying	430	60	50	100	85	70	65	3	
457	458	Mantyke	Water	Flying	345	45	20	50	60	120	50	4	
579	580	Ducklett	Water	Flying	305	62	44	50	44	50	55	5	
580	581	Swanna	Water	Flying	473	75	87	63	87	63	98	5	

14. Select specific columns of Legendary pokemons of type Fire

In [87]:
```
legendary_fire_df = df.loc[
    (
        (df["Type 1"]=="Fire") &
        df["Legendary"]
    ),
    ["Name","Attack","Generation"]
]
legendary_fire_df
```

Out[87]:

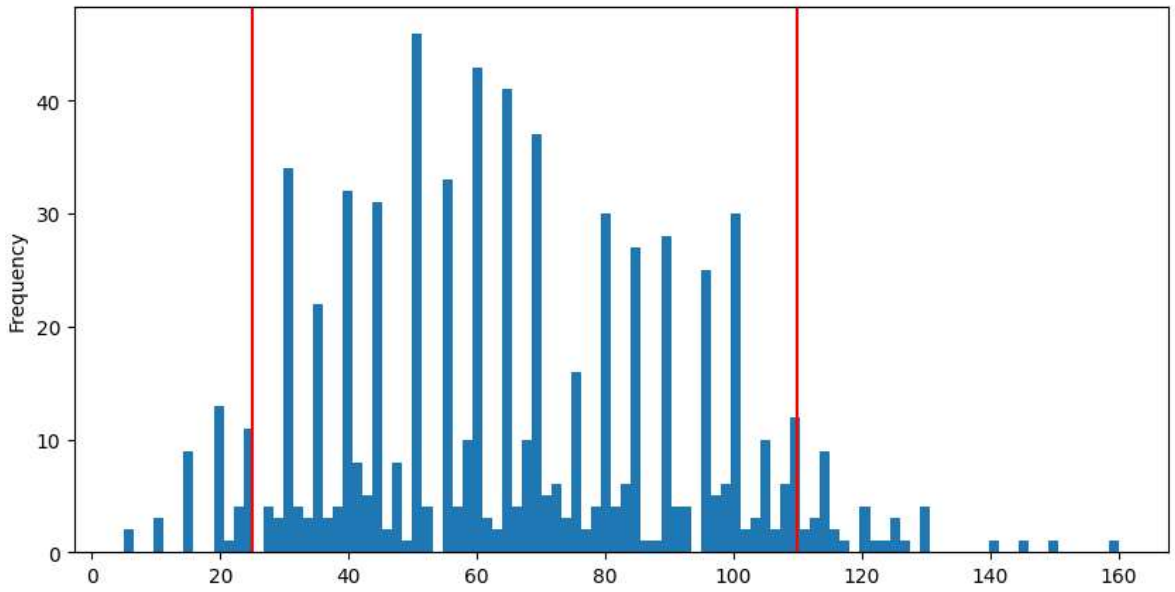
	Name	Attack	Generation
145	Moltres	100	1
243	Entei	115	2
249	Ho-oh	130	2
484	Heatran	90	4
720	Volcanion	110	6

15. Select Slow and Fast pokemons

This is the distribution of speed of the pokemons. The red lines indicate those bottom 5% and top 5% pokemons by speed:

In [88]:
```
ax = df['Speed'].plot(kind='hist', figsize=(10, 5), bins=100)
ax.axvline(df['Speed'].quantile(.05), color='red')
ax.axvline(df['Speed'].quantile(.95), color='red')
```

Out[88]:
```
<matplotlib.lines.Line2D at 0x7f44357e4210>
```



In [89]:
```
bottom_5=df['Speed'].quantile(.05)
top_5=df['Speed'].quantile(.95)
(bottom_5,top_5)
```

Out[89]:
```
(25.0, 110.0)
```

In [93]:
```
df.loc[
    (df["Speed"]<df['Speed'].quantile(.05)) |
    (df["Speed"]>df['Speed'].quantile(.95))
]
```

Out[93]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation
38	39	Jigglypuff	Normal	Fairy	270	115	45	20	45	25	20	1
50	51	Dugtrio	Ground	NaN	405	35	80	50	50	70	120	1
52	53	Persian	Normal	NaN	440	65	70	60	65	65	115	1
64	65	Alakazam	Psychic	NaN	500	55	50	45	135	95	120	1
73	74	Geodude	Rock	Ground	300	40	80	100	30	30	20	1
...	...	...	...	...	...	...	...	...	...	...	...	...
657	658	Greninja	Water	Dark	530	72	95	67	103	71	122	6
662	663	Talonflame	Fire	Flying	499	78	81	71	74	69	126	6
681	682	Spritzee	Fairy	NaN	341	78	52	60	63	65	23	6
700	701	Hawlucha	Fighting	Flying	500	78	92	75	74	63	118	6
714	715	Noivern	Flying	Dragon	535	85	70	80	97	80	123	6

68 rows × 13 columns

In [94]:
```
df.query("Speed < @bottom_5 or Speed > @top_5")
```

Out[94]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation
38	39	Jigglypuff	Normal	Fairy	270	115	45	20	45	25	20	1
50	51	Dugtrio	Ground	NaN	405	35	80	50	50	70	120	1
52	53	Persian	Normal	NaN	440	65	70	60	65	65	115	1
64	65	Alakazam	Psychic	NaN	500	55	50	45	135	95	120	1
73	74	Geodude	Rock	Ground	300	40	80	100	30	30	20	1
...	...	...	...	...	...	...	...	...	...	...	...	...
657	658	Greninja	Water	Dark	530	72	95	67	103	71	122	6
662	663	Talonflame	Fire	Flying	499	78	81	71	74	69	126	6
681	682	Spritzee	Fairy	NaN	341	78	52	60	63	65	23	6
700	701	Hawlucha	Fighting	Flying	500	78	92	75	74	63	118	6
714	715	Noivern	Flying	Dragon	535	85	70	80	97	80	123	6

68 rows × 13 columns

In [91]:
```
slow_fast_df = df.loc[(df["Speed"]<bottom_5) | (df["Speed"]>top_5)]
slow_fast_df
```

Out[91]:

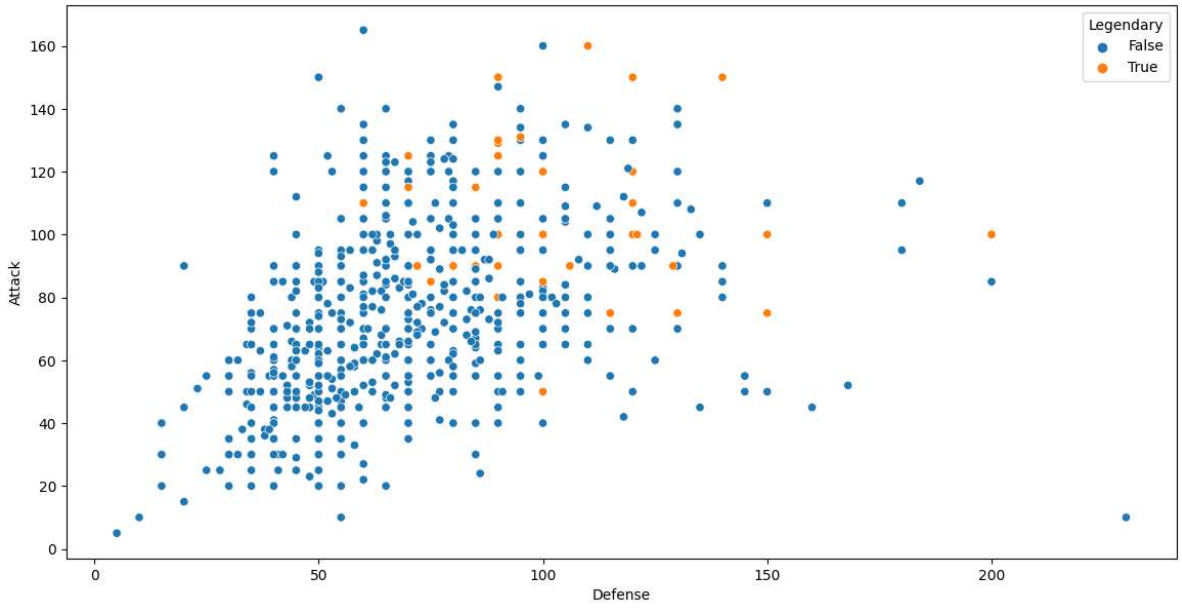
	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation
38	39	Jigglypuff	Normal	Fairy	270	115	45	20	45	25	20	1
50	51	Dugtrio	Ground	NaN	405	35	80	50	50	70	120	1
52	53	Persian	Normal	NaN	440	65	70	60	65	65	115	1
64	65	Alakazam	Psychic	NaN	500	55	50	45	135	95	120	1
73	74	Geodude	Rock	Ground	300	40	80	100	30	30	20	1
...	...	...	...	...	...	...	...	...	...	...	...	...
657	658	Greninja	Water	Dark	530	72	95	67	103	71	122	6
662	663	Talonflame	Fire	Flying	499	78	81	71	74	69	126	6
681	682	Spritzee	Fairy	NaN	341	78	52	60	63	65	23	6
700	701	Hawlucha	Fighting	Flying	500	78	92	75	74	63	118	6
714	715	Noivern	Flying	Dragon	535	85	70	80	97	80	123	6

68 rows × 13 columns

16. Find the Ultra Powerful Legendary Pokemon

In [53]:
```
fig, ax = plt.subplots(figsize=(14, 7))
sns.scatterplot(data=df, x="Defense", y="Attack", hue='Legendary', ax=ax)
```

Out[53]:
```
<Axes: xlabel='Defense', ylabel='Attack'>
```



In [95]:
```
df.loc[df["Legendary"]].sort_values(by=["Attack","Defense"],ascending=False)
```

Out[95]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	G
485	486	Regigigas	Normal	NaN	670	110	160	110	80	110	100	
382	383	Groudon	Ground	Fire	670	100	150	140	100	90	90	
643	644	Zekrom	Dragon	Electric	680	100	150	120	120	100	90	
383	384	Rayquaza	Dragon	Flying	680	105	150	90	150	90	95	
385	386	DeoxysNormal Forme	Psychic	NaN	600	50	150	50	150	50	150	
715	716	Xerneas	Fairy	NaN	680	126	131	95	131	98	99	
716	717	Yveltal	Dark	Flying	680	126	131	95	131	98	99	
249	250	Ho-oh	Fire	Flying	680	106	130	90	110	154	90	
645	646	Kyurem	Dragon	Ice	660	125	130	90	130	90	95	
638	639	Terrakion	Rock	Fighting	580	91	129	90	72	90	108	
644	645	LandorusIncarnate Forme	Ground	Flying	600	89	125	90	115	80	101	
481	482	Azelf	Psychic	NaN	580	75	125	70	125	70	115	
482	483	Dialga	Steel	Dragon	680	100	120	120	150	100	90	
492	493	Arceus	Normal	NaN	720	120	120	120	120	120	120	
483	484	Palkia	Water	Dragon	680	90	120	100	150	120	100	
642	643	Reshiram	Dragon	Fire	680	100	120	100	150	120	90	
243	244	Entei	Fire	NaN	580	115	115	85	90	75	100	
640	641	TornadusIncarnate Forme	Flying	NaN	580	79	115	70	125	80	111	
641	642	ThundurusIncarnate Forme	Electric	Flying	580	79	115	70	125	80	111	
720	721	Volcanion	Fire	Water	600	80	110	120	130	90	70	
149	150	Mewtwo	Psychic	Fighting	680	106	110	90	154	90	130	
719	720	HoopaHoopa Confined	Psychic	Ghost	600	80	110	60	150	130	70	
480	481	Mesprit	Psychic	NaN	580	80	105	105	105	105	80	
376	377	Regirock	Rock	NaN	580	80	100	200	50	100	50	
718	719	Diancie	Rock	Fairy	600	50	100	150	100	150	50	
717	718	Zygarde50% Forme	Dragon	Ground	600	108	100	121	81	95	95	
486	487	GiratinaAltered Forme	Ghost	Dragon	680	150	100	120	100	120	90	
384	385	Jirachi	Steel	Psychic	600	100	100	100	100	100	100	
491	492	ShayminLand Forme	Grass	Flying	600	100	100	100	100	100	100	
493	494	Victini	Psychic	Fire	600	100	100	100	100	100	100	
145	146	Moltres	Fire	Flying	580	90	100	90	125	85	90	
381	382	Kyogre	Water	NaN	670	100	100	90	150	140	90	

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	G
248	249	Lugia	Psychic	Flying	680	106	90	130	90	154	110	
637	638	Cobalion	Steel	Fighting	580	91	90	129	90	72	108	
484	485	Heatran	Fire	Steel	600	91	90	106	130	106	77	
490	491	Darkrai	Dark	NaN	600	70	90	90	135	90	125	
144	145	Zapdos	Electric	Flying	580	90	90	85	125	90	100	
380	381	Latios	Dragon	Psychic	600	80	90	80	130	110	110	
639	640	Virizion	Grass	Fighting	580	91	90	72	90	129	108	
143	144	Articuno	Ice	Flying	580	90	85	100	95	125	85	
242	243	Raikou	Electric	NaN	580	90	85	75	115	100	115	
379	380	Latias	Dragon	Psychic	600	80	80	90	110	130	110	
378	379	Registeel	Steel	NaN	580	80	75	150	75	150	50	
479	480	Uxie	Psychic	NaN	580	75	75	130	75	130	95	
244	245	Suicune	Water	NaN	580	100	75	115	90	115	85	
377	378	Regice	Ice	NaN	580	80	50	100	100	200	50	

In [ ]: