Data Exploration & Python

# Importing pandas and matplotlib packages[1].

In [1]:

**import** pandas **as** pd

**import** matplotlib.pyplot **as** plot

# Importing csv file and then loading it into a dataframe, and printing few of the records in the data.[2]

In [8]:

data **=** pd**.**read\_csv('Desktop/AIT 580/Assignment 3/1950-2019\_actual\_tornadoes.csv')

In [9]:

data

Out[9]:

**om yr mo dy date 0** 1 1950 1 3 1950-

01-03

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **time** | **tz** | **st** | **stf** | **stn ... len** | **wid** | **ns** | **sn** | **sg** | **f1** | **f2** |
| 11:00:00 | 3 | MO | 29 | 1 ... 9.50 | 150 | 2 | 0 | 1 | 0 | 0 |
| 11:55:00 | 3 | IL | 17 | 2 ... 3.60 | 130 | 1 | 1 | 1 | 135 | 0 |
| 16:00:00 | 3 | OH | 39 | 1 ... 0.10 | 10 | 1 | 1 | 1 | 161 | 0 |
| 05:25:00 | 3 | AR | 5 | 1 ... 0.60 | 17 | 1 | 1 | 1 | 113 | 0 |
| 19:30:00 | 3 | MO | 29 | 2 ... 2.30 | 300 | 1 | 1 | 1 | 93 | 0 |
| ... | ... | ... | ... | ... ... ... | ... | ... | ... | ... | ... | ... |
| 16:03:00 | 3 | MS | 28 | 0 ... 7.70 | 900 | 1 | 1 | 1 | 7 | 0 |
| 16:13:00 | 3 | MS | 28 | 0 ... 3.82 | 200 | 1 | 1 | 1 | 19 | 0 |
| 16:32:00 | 3 | MS | 28 | 0 ... 2.61 | 200 | 1 | 1 | 1 | 105 | 0 |
| 17:13:00 | 3 | MS | 28 | 0 ... 3.23 | 125 | 1 | 1 | 1 | 101 | 0 |

**1** 2 1950 1 3 1950-

01-03

**2** 3 1950 1 3 1950-

01-03

**3** 4 1950 1 13 1950-

01-13

**4** 5 1950 1 25 1950-

01-25

**...** ... ... ... ... ...

**65157** 618537 2019 12 29 2019-

12-29

**65158** 618538 2019 12 29 2019-

12-29

**65159** 618539 2019 12 29 2019-

12-29

**65160** 618540 2019 12 29 2019-

12-29

**om yr mo dy date time tz st stf stn ... len wid ns sn sg f1 f2**

**65161** 618541 2019 12 29 2019-

12-29

18:50:00 3 AL 1 0 ... 5.07 50 1 1 1 83 0

65162 rows × 29 columns

# Displaying Few Records

In [15]:

print(data**.**head())

om yr mo dy date time tz st stf stn ... len wid \ 0 1 1950 1 3 1950-01-03 11:00:00 3 MO 29 1 ... 9.5 150

1 2 1950 1 3 1950-01-03 11:55:00 3 IL 17 2 ... 3.6 130

2 3 1950 1 3 1950-01-03 16:00:00 3 OH 39 1 ... 0.1 10

3 4 1950 1 13 1950-01-13 05:25:00 3 AR 5 1 ... 0.6 17

4 5 1950 1 25 1950-01-25 19:30:00 3 MO 29 2 ... 2.3 300

ns sn sg f1 f2 f3 f4 fc 0 2 0 1 0 0 0 0 0

1 1 1 1 135 0 0 0 0

2 1 1 1 161 0 0 0 0

3 1 1 1 113 0 0 0 0

4 1 1 1 93 0 0 0 0

[5 rows x 29 columns]

Q1. How many tornadoes were there per year?

[3].

Explanation:- To calculate the number of tornadoes each year, group the years first and then find the length of the column "om" to get the total number of tornadoes. This gives us the annual number of tornadoes. I'm using the group by function to group the data by year, as well as the agg() method

to get the length.

In [18]:

tornadoesPerYear **=** data**.**groupby("yr")**.**agg({'om':len})

In [19]:

tornadoesPerYear

|  |  |  |
| --- | --- | --- |
| Out[19]: | **yr** | **om** |
|  | **1950** | 201 |
|  | **1951** | 260 |
|  | **1952** | 240 |
|  | **1953** | 421 |
|  | **1954** | 550 |
|  |  |  |

|  |  |
| --- | --- |
| **yr** | **om** |
| **2015** | 1177 |
| **2016** | 976 |
| **2017** | 1428 |
| **2018** | 1126 |
| **2019** | 1517 |

70 rows × 1 columns

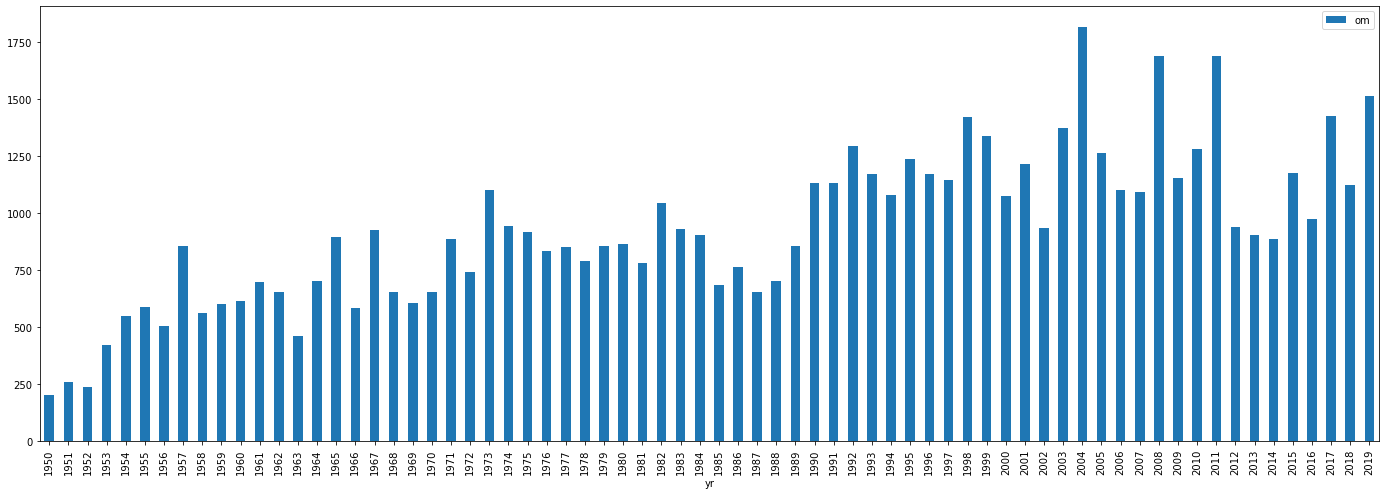
In [20]:

tornadoesPerYear**.**plot(kind**=**'bar',figsize**=**(24,8))

Out[20]:

In [37]:

<AxesSubplot:xlabel='yr'>



# Q2) What month had the most tornadoes? (Bar Chart).

# [3].

Explanation:- To determine the month with the most tornadoes, arrange the data by month and

then calculate the length of "om," which represents the number of tornadoes in a given state. We may deduce from the bar graph and table below that the month 5 (May) had the most tornadoes.

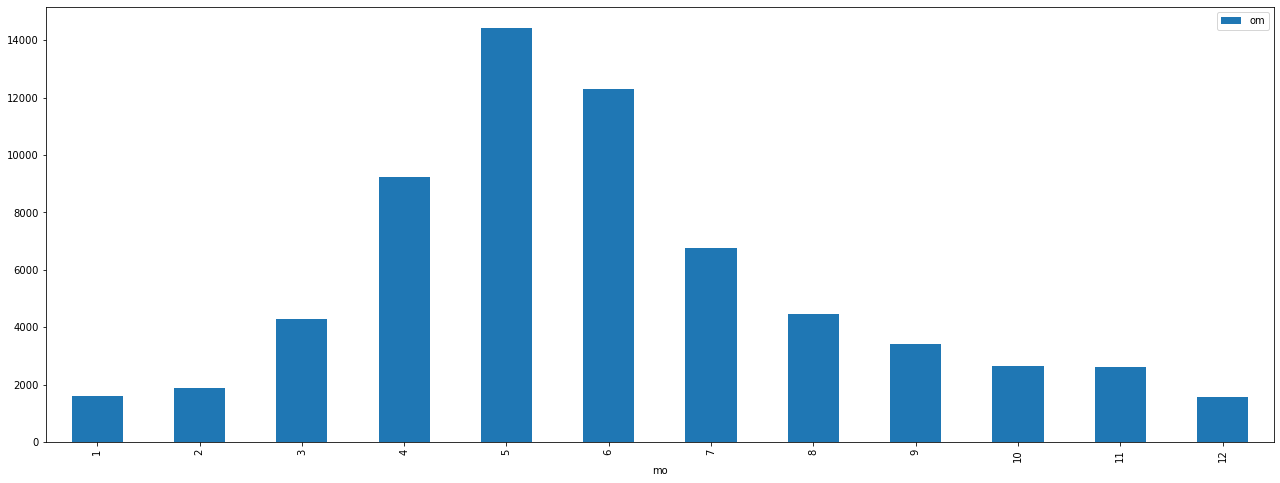
maxTornados **=** data**.**groupby("mo")**.**agg({'om':len})

In [22]:

maxTornados**.**plot(kind**=**'bar',figsize **=** (22,8))

Out[22]:

<AxesSubplot:xlabel='mo'>



In [25]:

maxTornados

|  |  |  |
| --- | --- | --- |
| Out[25]: | **mo** | **om** |
|  | **1** | 1616 |
|  | **2** | 1892 |
|  | **3** | 4293 |
|  | **4** | 9231 |
|  | **5** | 14434 |
|  | **6** | 12296 |
|  | **7** | 6746 |
|  | **8** | 4453 |
|  | **9** | 3405 |
|  | **10** | 2636 |
|  | **11** | 2602 |
|  | **12** | 1558 |

May month has the most Tornadoes.

# Q3) What state had the most tornados? (bar chart)

# [3].

Explanation:- Group the data by state to identify the state with more tornadoes, then find the length of "om" to get the total number of tornadoes per state. We may deduce from the graph and table

below that the state "tx" has the highest number of tornadoes.

In [26]:

maxTornadosState **=**data**.**groupby("st")**.**agg({'om':len})

In [27]:

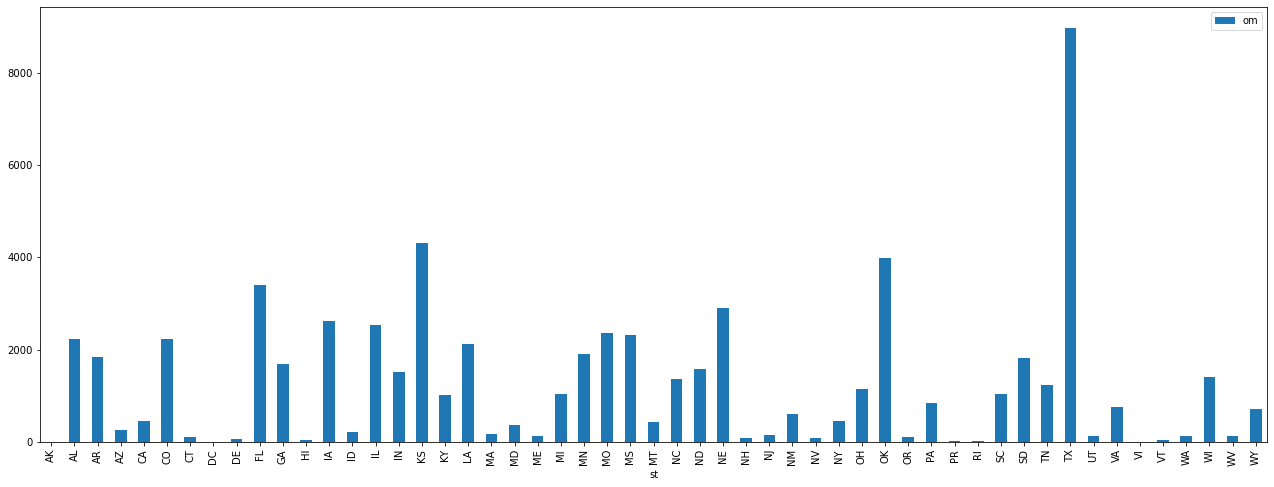
maxTornadosState**.**plot(kind**=**'bar',figsize **=** (22,8))

Out[27]:

In [28]:

maxTornadosState

<AxesSubplot:xlabel='st'>



|  |  |  |
| --- | --- | --- |
| Out[28]: | **st** | **om** |
|  | **AK** | 4 |
|  | **AL** | 2223 |
|  | **AR** | 1838 |
|  | **AZ** | 262 |
|  | **CA** | 450 |
|  | **CO** | 2223 |
|  | **CT** | 105 |
|  | **DC** | 2 |
|  | **DE** | 63 |
|  | **FL** | 3406 |
|  | **GA** | 1694 |
|  | **HI** | 41 |
|  | **IA** | 2626 |
|  | **ID** | 218 |
|  | **IL** | 2546 |
|  | **IN** | 1509 |
|  | **KS** | 4322 |
|  | **KY** | 1009 |
|  | **LA** | 2132 |
|  | **MA** | 171 |
|  | **MD** | 361 |

**om**

**st**

**ME** 133

**MI** 1051

**MN** 1902

**MO** 2362

**MS** 2319

**MT** 427

**NC** 1366

**ND** 1580

**NE** 2901

**NH** 92

**NJ** 155

**NM** 612

**NV** 93

**NY** 452

**OH** 1145

**OK** 3990

**OR** 119

**PA** 853

**PR** 28

**RI** 12

**SC** 1031

**SD** 1817

**TN** 1232

**TX** 8971

**UT** 130

**VA** 761

**VI** 1

**VT** 45

**WA** 121

**WI** 1409

**WV** 141

**WY** 706

# Q4) What states had the most F4 tornados? (count)

# [3].

Explanation:- To determine which states experienced the most F4 tornadoes, I first constructed a

new data frame by deleting any columns with the value "f4" 0. First, we must sort the data by state and determine the length of "om." This tells us the total number of tornadoes that have occurred in each state. Then, in decreasing order, sort it. We may retrieve the top state with the highest f4 count by utilizing the head() method. We may deduce from the graph and table below that the state "tx" has the highest number of tornadoes.

In [29]:

dt **=** data[data["f4"] **>** 0]

In [30]:

maxF4Tornadoes **=**dt**.**groupby("st")**.**agg({'f4':len})**.**sort\_values("f4", ascending **= False**)

In [31]:

maxF4Tornadoes

|  |  |  |
| --- | --- | --- |
| Out[31]: | **st** | **f4** |
|  | **IA** | 25 |
|  | **MS** | 24 |
|  | **NE** | 24 |
|  | **GA** | 21 |
|  | **IL** | 21 |
|  | **KS** | 21 |
|  | **TX** | 20 |
|  | **AR** | 16 |
|  | **OK** | 16 |
|  | **IN** | 15 |
|  | **LA** | 13 |
|  | **AL** | 12 |
|  | **MN** | 10 |
|  | **NC** | 9 |
|  | **TN** | 9 |
|  | **WI** | 9 |
|  | **MO** | 7 |
|  | **KY** | 7 |
|  | **MI** | 6 |
|  | **VA** | 6 |
|  | **OH** | 5 |

**f4**

**st**

**FL** 4

**NH** 1

**NJ** 1

**NY** 1

**PA** 1

**SD** 1

In [32]:

maxF4Tornadoes**.**head(1)

Out[32]: **f4 st**

**IA** 25

# Q5) Is there a relationship between tornado length and width? (plot) [4].

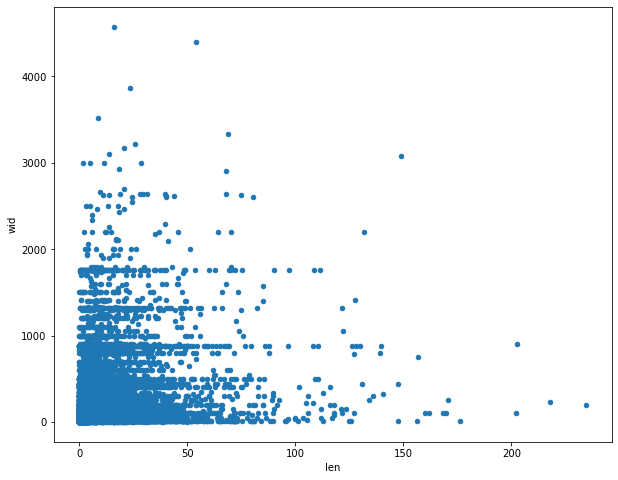
Explanation:- I'm using a scatter plot to determine the relationship between two columns. When the values are smaller, the length and breadth are linked, as seen in the graph below.

In [33]:

data**.**plot**.**scatter(x**=**'len',y**=**'wid',figsize**=**(10,8))

Out[33]:

<AxesSubplot:xlabel='len', ylabel='wid'>



# Q6) Does there appear to be a typical time of day for tornados to strike? (plot). [3].

Explanation:- I used a line chart to figure out when tornadoes were the most common. Data must first be sorted by time, and then the length of the column "om" must be determined. This generates a table containing the time and number of torandos that occurred at that moment. We may deduce from the graph and table that the highest number of tornadoes occurred between 15:00 and 17:30.

In [38]:

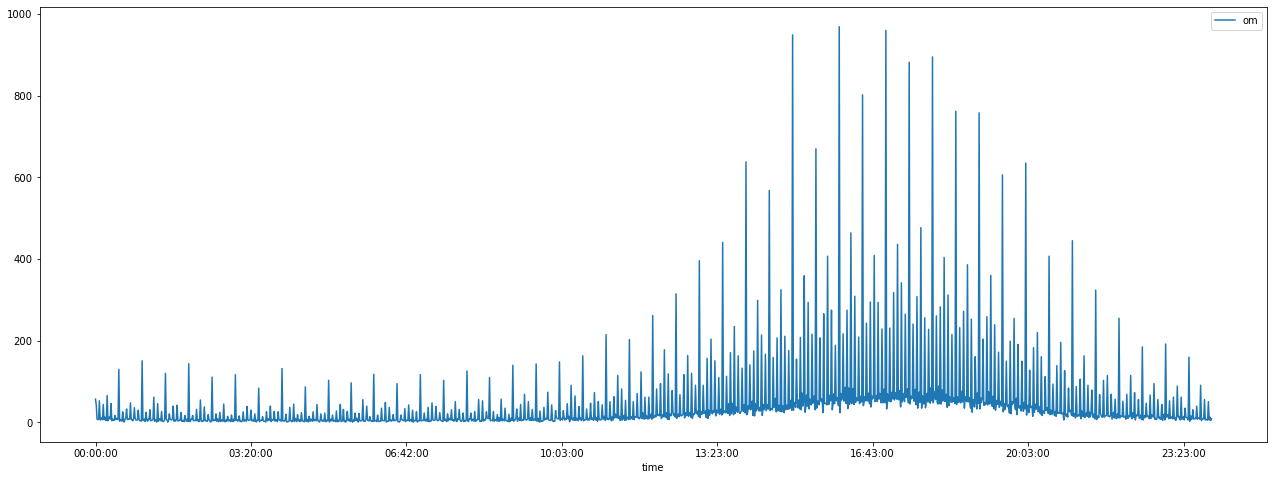
maxTimes **=** data**.**groupby("time")**.**agg({'om':len})

In [40]:

maxTimes**.**plot(kind**=**'line',subplots**=**'true',figsize **=** (22,8))

Out[40]:

array([<AxesSubplot:xlabel='time'>], dtype=object)



In [41]:

maxTimes

|  |  |  |
| --- | --- | --- |
| Out[41]: | **time** | **om** |
|  | **00:00:00** | 57 |
|  | **00:01:00** | 42 |
|  | **00:02:00** | 8 |
|  | **00:03:00** | 7 |
|  | **00:04:00**  **... 23:55:00** | 8  ... 51 |
|  | **23:56:00** | 6 |
|  | **23:57:00** | 13 |
|  | **23:58:00** | 5 |
|  | **23:59:00** | 9 |
|  | 1437 rows | × 1 columns |
| In [ ]: |  |  |

Reference: -

1. *Python Data Analysis with Pandas and Matplotlib*. <https://ourcodingclub.github.io/tutorials/pandas-python-intro/> . Accessed 18 Sep. 2021.(1)
2. *Python Pandas Read\_csv: Load Data from CSV Files | Shane Lynn*. <https://www.shanelynn.ie/python-pandas-read-csv-load-data-from-csv-files/> . Accessed 18 Sep. 2021.
3. “Python - Converting Pandas Groupby.Groups Result into Dataframe, Using Index Tuple Value as Row and Columns Name of Dataframe.” *Stack Overflow*, [https://stackoverflow.com/questions/26260650/converting-pandas-groupby-groups-result-into-dataframe-using-index-tuple-value .](https://stackoverflow.com/questions/26260650/converting-pandas-groupby-groups-result-into-dataframe-using-index-tuple-value%20.%20Accessed%2018%20Sep.%202021)  Accessed 18 Sep. 2021.
4. *Pandas.DataFrame.Plot.Scatter — Pandas 0.25.0 Documentation*. <https://pandas.pydata.org/pandas-docs/version/0.25.0/reference/api/pandas.DataFrame.plot.scatter.html> . Accessed 18 Sep. 2021.