

In [1]:

```
1 import pandas as pd
2 import numpy as np
3 import seaborn as sns
4 import matplotlib.pyplot as plt
```

In [4]:

```
1 df = pd.read_csv("uber.csv")
2 df
```

Out[4]:

Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passen
0	24238194	2015-05-07 19:52:06.0000003	7.5	2015-05-07 19:52:06 UTC	-73.999817	40.738354	-73.999512	40.723217
1	27835199	2009-07-17 20:04:56.0000002	7.7	2009-07-17 20:04:56 UTC	-73.994355	40.728225	-73.994710	40.750325
2	44984355	2009-08-24 21:45:00.00000061	12.9	2009-08-24 21:45:00 UTC	-74.005043	40.740770	-73.962565	40.772647
3	25894730	2009-06-26 08:22:21.0000001	5.3	2009-06-26 08:22:21 UTC	-73.976124	40.790844	-73.965316	40.803349
4	17610152	2014-08-28 17:47:00.000000188	16.0	2014-08-28 17:47:00 UTC	-73.925023	40.744085	-73.973082	40.761247
...	...	...	...	...	...	...	...	...
199995	42598914	2012-10-28 10:49:00.00000053	3.0	2012-10-28 10:49:00 UTC	-73.987042	40.739367	-73.986525	40.740297
199996	16382965	2014-03-14 01:09:00.0000008	7.5	2014-03-14 01:09:00 UTC	-73.984722	40.736837	-74.006672	40.739620
199997	27804658	2009-06-29 00:42:00.00000078	30.9	2009-06-29 00:42:00 UTC	-73.986017	40.756487	-73.858957	40.692588
199998	20259894	2015-05-20 14:56:25.0000004	14.5	2015-05-20 14:56:25 UTC	-73.997124	40.725452	-73.983215	40.695415
199999	11951496	2010-05-15 04:08:00.00000076	14.1	2010-05-15 04:08:00 UTC	-73.984395	40.720077	-73.985508	40.768793

200000 rows × 9 columns



In [5]:

```
1 df.head()
```

Out[5]:

Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_c
0	24238194	2015-05-07 19:52:06.0000003	7.5	2015-05-07 19:52:06 UTC	-73.999817	40.738354	-73.999512	40.723217
1	27835199	2009-07-17 20:04:56.0000002	7.7	2009-07-17 20:04:56 UTC	-73.994355	40.728225	-73.994710	40.750325
2	44984355	2009-08-24 21:45:00.00000061	12.9	2009-08-24 21:45:00 UTC	-74.005043	40.740770	-73.962565	40.772647
3	25894730	2009-06-26 08:22:21.0000001	5.3	2009-06-26 08:22:21 UTC	-73.976124	40.790844	-73.965316	40.803349
4	17610152	2014-08-28 17:47:00.000000188	16.0	2014-08-28 17:47:00 UTC	-73.925023	40.744085	-73.973082	40.761247



In [6]:

```
1 df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200000 entries, 0 to 199999
Data columns (total 9 columns):
#   Column                Non-Null Count  Dtype
---  ---
0   Unnamed: 0            200000 non-null   int64
1   key                   200000 non-null   object
2   fare_amount           200000 non-null   float64
3   pickup_datetime       200000 non-null   object
4   pickup_longitude      200000 non-null   float64
5   pickup_latitude       200000 non-null   float64
6   dropoff_longitude     199999 non-null   float64
7   dropoff_latitude      199999 non-null   float64
8   passenger_count       200000 non-null   int64
dtypes: float64(5), int64(2), object(2)
memory usage: 13.7+ MB
```

In [7]:

```
1 df.columns

Out[7]:
Index(['Unnamed: 0', 'key', 'fare_amount', 'pickup_datetime',
      'pickup_longitude', 'pickup_latitude', 'dropoff_longitude',
      'dropoff_latitude', 'passenger_count'],
      dtype='object')
```

In [8]:

```
1 df = df.drop(['Unnamed: 0', 'key'], axis =1)
```

In [9]:

```
1 df.head()

Out[9]:
```

	fare_amount		pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
0	7.5	2015-05-07 19:52:06 UTC	-73.999817	40.738354	-73.999512	40.723217	1	
1	7.7	2009-07-17 20:04:56 UTC	-73.994355	40.728225	-73.994710	40.750325	1	
2	12.9	2009-08-24 21:45:00 UTC	-74.005043	40.740770	-73.962565	40.772647	1	
3	5.3	2009-06-26 08:22:21 UTC	-73.976124	40.790844	-73.965316	40.803349	3	
4	16.0	2014-08-28 17:47:00 UTC	-73.925023	40.744085	-73.973082	40.761247	5	

In [10]:

```
1 df.shape

Out[10]:
(200000, 7)
```

In [11]:

```
1 df.dtypes

Out[11]:
fare_amount           float64
pickup_datetime       object
pickup_longitude      float64
pickup_latitude       float64
dropoff_longitude     float64
dropoff_latitude      float64
passenger_count       int64
dtype: object
```

In [12]:

```
1 df.describe()
```

Out[12]:

	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
count	200000.000000	200000.000000	200000.000000	199999.000000	199999.000000	200000.000000
mean	11.359955	-72.527638	39.935885	-72.525292	39.923890	1.684535
std	9.901776	11.437787	7.720539	13.117408	6.794829	1.385997
min	-52.000000	-1340.648410	-74.015515	-3356.666300	-881.985513	0.000000
25%	6.000000	-73.992065	40.734796	-73.991407	40.733823	1.000000
50%	8.500000	-73.981823	40.752592	-73.980093	40.753042	1.000000
75%	12.500000	-73.967154	40.767158	-73.963658	40.768001	2.000000
max	499.000000	57.418457	1644.421482	1153.572603	872.697628	208.000000

In [13]:

```
1 df.isnull().sum()
```

Out[13]:

```
fare_amount      0
pickup_datetime  0
pickup_longitude  0
pickup_latitude  0
dropoff_longitude 1
dropoff_latitude  1
passenger_count  0
dtype: int64
```

In [14]:

```
1 df['dropoff_latitude'].fillna(value=df['dropoff_latitude'].mean(), inplace=True)
```

In [15]:

```
1 df.isnull().sum()
```

Out[15]:

```
fare_amount      0
pickup_datetime  0
pickup_longitude  0
pickup_latitude  0
dropoff_longitude 1
dropoff_latitude  0
passenger_count  0
dtype: int64
```

In [16]:

```
1 df['dropoff_longitude'].fillna(value=df['dropoff_longitude'].median(), inplace=True)
```

In [17]:

```
1 df.isnull().sum()
```

Out[17]:

```
fare_amount      0
pickup_datetime  0
pickup_longitude  0
pickup_latitude  0
dropoff_longitude 0
dropoff_latitude  0
passenger_count  0
dtype: int64
```

In [18]:

```
1 df.dtypes
```

Out[18]:

```
fare_amount      float64
pickup_datetime  object
pickup_longitude float64
pickup_latitude  float64
dropoff_longitude float64
dropoff_latitude float64
passenger_count  int64
dtype: object
```

In [19]:

```
1 df.pickup_datetime = pd.to_datetime(df.pickup_datetime, errors='coerce')
```

In [20]:

```
1 df.dtypes
```

Out[20]:

```
fare_amount      float64
pickup_datetime  datetime64[ns, UTC]
pickup_longitude float64
pickup_latitude  float64
dropoff_longitude float64
dropoff_latitude float64
passenger_count  int64
dtype: object
```

In [21]:

```
1 df= df.assign(hour = df.pickup_datetime.dt.hour,
2               day = df.pickup_datetime.dt.day,
3               month = df.pickup_datetime.dt.month,
4               year = df.pickup_datetime.dt.year,
5               dayofweek = df.pickup_datetime.dt.dayofweek)
6
```

In [22]:

```
1 df.head
```

Out[22]:

<bound method NDFrame.head of											
				fare_amount		pickup_datetime		pickup_longitude \			
0	7.5	2015-05-07	19:52:06+00:00					-73.999817			
1	7.7	2009-07-17	20:04:56+00:00					-73.994355			
2	12.9	2009-08-24	21:45:00+00:00					-74.005043			
3	5.3	2009-06-26	08:22:21+00:00					-73.976124			
4	16.0	2014-08-28	17:47:00+00:00					-73.925023			
...	...	...	...					...			
199995	3.0	2012-10-28	10:49:00+00:00					-73.987042			
199996	7.5	2014-03-14	01:09:00+00:00					-73.984722			
199997	30.9	2009-06-29	00:42:00+00:00					-73.986017			
199998	14.5	2015-05-20	14:56:25+00:00					-73.997124			
199999	14.1	2010-05-15	04:08:00+00:00					-73.984395			

						\					
			pickup_latitude		dropoff_longitude	dropoff_latitude		passenger_count			
0			40.738354		-73.999512	40.723217		1			
1			40.728225		-73.994710	40.750325		1			
2			40.740770		-73.962565	40.772647		1			
3			40.790844		-73.965316	40.803349		3			
4			40.744085		-73.973082	40.761247		5			
...			...		...	...		...			
199995			40.739367		-73.986525	40.740297		1			
199996			40.736837		-74.006672	40.739620		1			
199997			40.756487		-73.858957	40.692588		2			
199998			40.725452		-73.983215	40.695415		1			
199999			40.720077		-73.985508	40.768793		1			

						\					
			hour	day	month	year	dayofweek				
0			19	7	5	2015	3				
1			20	17	7	2009	4				
2			21	24	8	2009	0				
3			8	26	6	2009	4				
4			17	28	8	2014	3				
...			...	...	...	...	...				
199995			10	28	10	2012	6				
199996			1	14	3	2014	4				
199997			0	29	6	2009	0				
199998			14	20	5	2015	2				
199999			4	15	5	2010	5				

[200000 rows x 12 columns]>

In [41]:

```
1 df = df.drop('pickup_datetime', axis =1)
```

```
-----
KeyError                                Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_19836\3060153060.py in <module>
----> 1 df = df.drop('pickup_datetime', axis =1)

C:\ProgramData\Anaconda3\lib\site-packages\pandas\util\_decorators.py in wrapper(*args, **kwargs)
    309         stacklevel=stacklevel,
    310     )
--> 311     return func(*args, **kwargs)
    312
    313     return wrapper

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\frame.py in drop(self, labels, axis, index, columns, level, inplace, errors)
    4955         weight 1.0      0.8
    4956         """
-> 4957     return super().drop(
    4958         labels=labels,
    4959         axis=axis,

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\generic.py in drop(self, labels, axis, index, columns, level, inplace, errors)
    4265     for axis, labels in axes.items():
    4266         if labels is not None:
-> 4267         obj = obj._drop_axis(labels, axis, level=level, errors=errors)
    4268
    4269     if inplace:

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\generic.py in _drop_axis(self, labels, axis, level, errors, consolidate, only_slice)
    4309         new_axis = axis.drop(labels, level=level, errors=errors)
    4310     else:
-> 4311         new_axis = axis.drop(labels, errors=errors)
    4312         indexer = axis.get_indexer(new_axis)
    4313

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\indexes\base.py in drop(self, labels, errors)
    6659     if mask.any():
    6660         if errors != "ignore":
-> 6661             raise KeyError(f"{list(labels[mask])} not found in axis")
    6662         indexer = indexer[~mask]
    6663     return self.delete(indexer)
```

**KeyError:** "['pickup\_datetime'] not found in axis"

In [ ]:

```
1 df.head
```

In [25]:

```
1 df.dtypes
```

Out[25]:

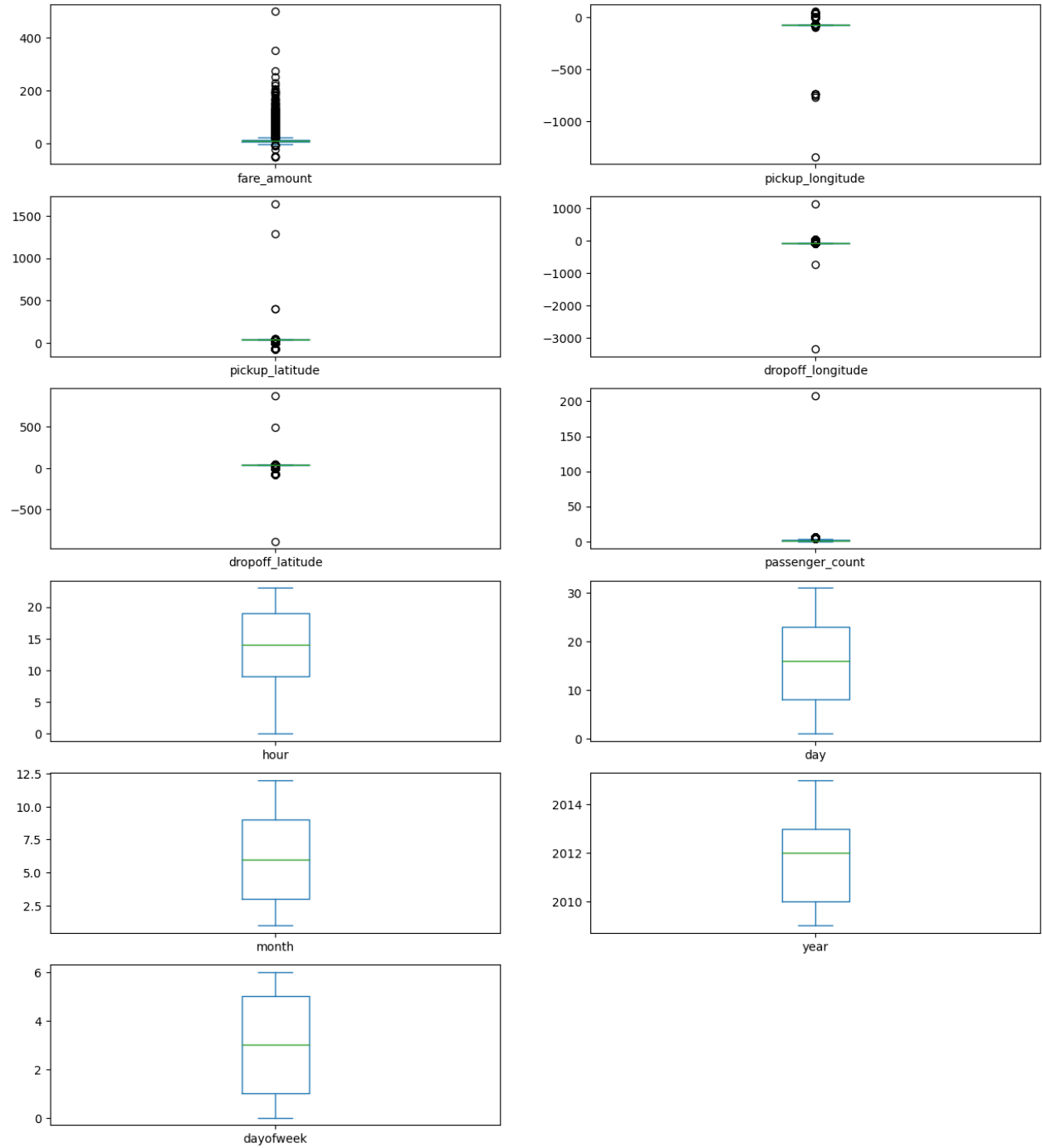
```
fare_amount      float64
pickup_longitude  float64
pickup_latitude   float64
dropoff_longitude float64
dropoff_latitude  float64
passenger_count   int64
hour              int64
day              int64
month            int64
year             int64
dayofweek         int64
dtype: object
```

In [26]:

```
1 df.plot(kind="box", subplots = True,layout =(7,2),figsize=(15,20))
```

Out[26]:

```
fare_amount      AxesSubplot(0.125,0.786098;0.352273x0.0939024)
pickup_longitude AxesSubplot(0.547727,0.786098;0.352273x0.0939024)
pickup_latitude  AxesSubplot(0.125,0.673415;0.352273x0.0939024)
dropoff_longitude AxesSubplot(0.547727,0.673415;0.352273x0.0939024)
dropoff_latitude AxesSubplot(0.125,0.560732;0.352273x0.0939024)
passenger_count  AxesSubplot(0.547727,0.560732;0.352273x0.0939024)
hour             AxesSubplot(0.125,0.448049;0.352273x0.0939024)
day              AxesSubplot(0.547727,0.448049;0.352273x0.0939024)
month            AxesSubplot(0.125,0.335366;0.352273x0.0939024)
year             AxesSubplot(0.547727,0.335366;0.352273x0.0939024)
dayofweek        AxesSubplot(0.125,0.222683;0.352273x0.0939024)
dtype: object
```



In [27]:

```
1 def remove_outlier(df1, col):
2     Q1 =df1[col].quantile(0.25)
3     Q3= df1[col].quantile (0.75)
4     IQR= Q3 - Q1
5     lower_whisker= Q1-1.5*IQR
6     upper_whisker= Q3+1.5*IQR
7     df[col] = np.clip(df1[col], lower_whisker, upper_whisker)
8     return df1
9
10 def treat_outliers_all(df1, col_list):
11     for c in col_list:
12         df1 = remove_outlier(df, c)
13     return df1
```

In [28]:

```
1 df = treat_outliers_all(df,df.iloc[:, 0::])
```

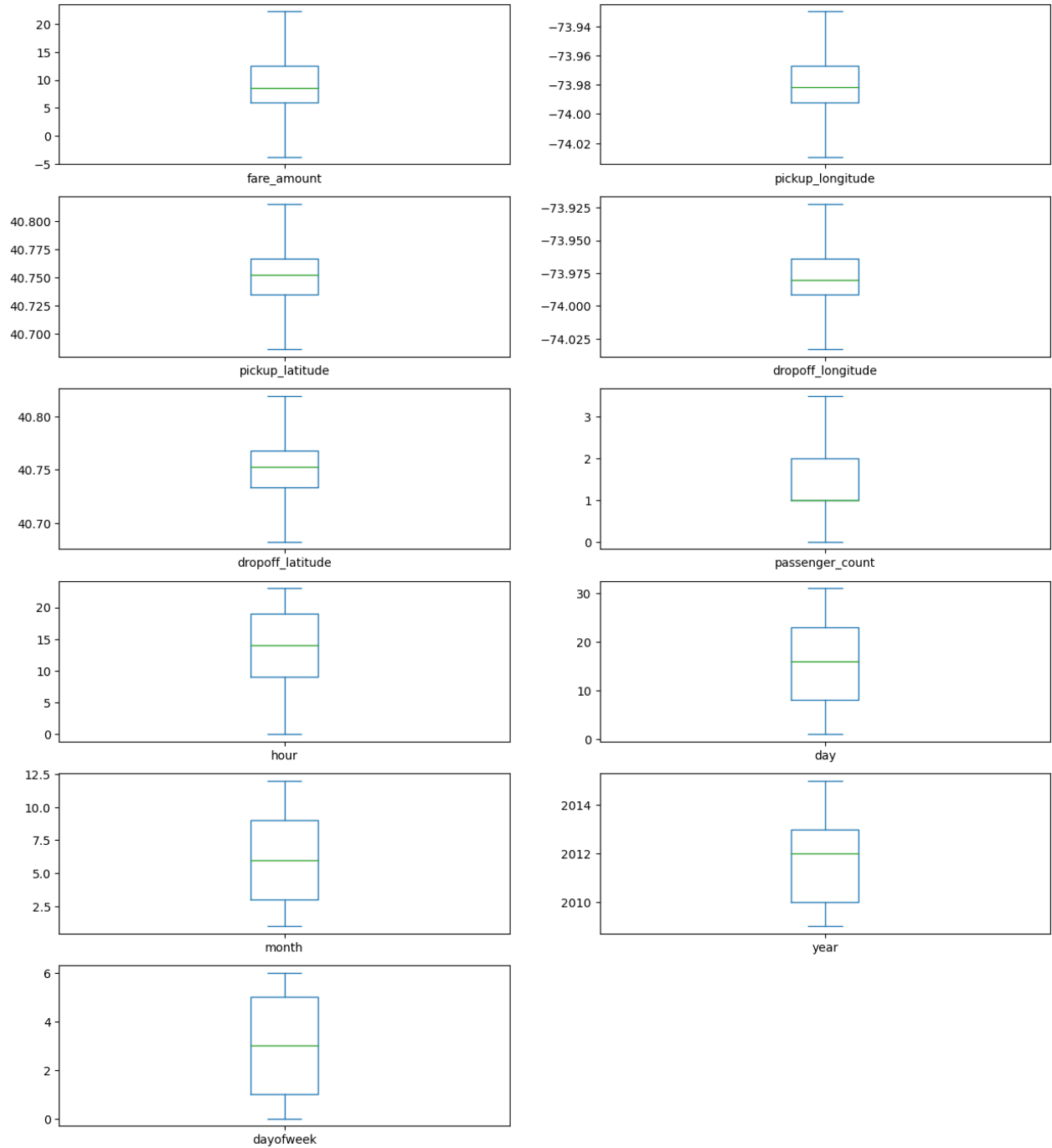


In [29]:

```
1 df.plot(kind = "box",subplots = True,layout=(7,2),figsize=(15,20))
```

Out[29]:

```
fare_amount      AxesSubplot(0.125,0.786098;0.352273x0.0939024)
pickup_longitude AxesSubplot(0.547727,0.786098;0.352273x0.0939024)
pickup_latitude  AxesSubplot(0.125,0.673415;0.352273x0.0939024)
dropoff_longitude AxesSubplot(0.547727,0.673415;0.352273x0.0939024)
dropoff_latitude AxesSubplot(0.125,0.560732;0.352273x0.0939024)
passenger_count  AxesSubplot(0.547727,0.560732;0.352273x0.0939024)
hour             AxesSubplot(0.125,0.448049;0.352273x0.0939024)
day              AxesSubplot(0.547727,0.448049;0.352273x0.0939024)
month            AxesSubplot(0.125,0.335366;0.352273x0.0939024)
year             AxesSubplot(0.547727,0.335366;0.352273x0.0939024)
dayofweek        AxesSubplot(0.125,0.222683;0.352273x0.0939024)
dtype: object
```



In [ ]:

```
1 pip install haversine
```

In [42]:

```

1 import haversine as hs
2 travel_dist = []
3 for pos in range(len(df['pickup_longitude'])):
4     long1,lati1,long2,lati2 = [df['pickup_longitude'][pos],df['pickup_latitude'][pos],df['dropoff_longitude'][pos],df['dropoff_
5     loc1 =(lati1,long1)
6     loc2 =(lati2,long2)
7     c = hs.haversine(loc1,loc2)
8     travel_dist.append(c)
9
10 print(travel_dist)
11 df['dist_travel_km']= travel_dist
12 df.head

```

IOPub data rate exceeded.

The notebook server will temporarily stop sending output to the client in order to avoid crashing it. To change this limit, set the config variable `--NotebookApp.iopub\_data\_rate\_limit`.

Current values:

NotebookApp.iopub\_data\_rate\_limit=1000000.0 (bytes/sec)

NotebookApp.rate\_limit\_window=3.0 (secs)

Out[42]:

```

<bound method NDFrame.head of
0      7.50      -73.999817      40.738354      -73.999512
1      7.70      -73.994355      40.728225      -73.994710
2     12.90      -74.005043      40.740770      -73.962565
3      5.30      -73.976124      40.790844      -73.965316
4     16.00      -73.929786      40.744085      -73.973082
...
199995      3.00      -73.987042      40.739367      -73.986525
199996      7.50      -73.984722      40.736837      -74.006672
199997     22.25      -73.986017      40.756487      -73.922036
199998     14.50      -73.997124      40.725452      -73.983215
199999     14.10      -73.984395      40.720077      -73.985508

      dropoff_latitude  passenger_count  hour  day  month  year  dayofweek  \
0      40.723217           1.0    19    7    5    2015         3
1      40.750325           1.0    20    17   7    2009         4
2      40.772647           1.0    21    24   8    2009         0
3      40.803349           3.0     8    26   6    2009         4
4      40.761247           3.5    17    28   8    2014         3
...
199995      40.740297           1.0    10    28  10    2012         6
199996      40.739620           1.0     1    14   3    2014         4
199997      40.692588           2.0     0    29   6    2009         0
199998      40.695415           1.0    14    20   5    2015         2
199999      40.768793           1.0     4    15   5    2010         5

      dist_travel_km
0      12758.286154
1      12756.625962
2      12754.446942
3      12760.320445
4      12755.985008
...
199995      12756.954766
199996      12758.913797
199997      12751.688525
199998      12755.039558
199999      12754.696728

```

[200000 rows x 12 columns]>

In [43]:

```

1 df = df.loc[(df.dist_travel_km>=1) |(df.dist_travel_km<=130) ]
2 print("Remaining obervation:" , df.shape)

```

Remaining obervation: (200000, 12)

In [44]:

```
1 incorrect_coordinates =df.loc[(df.pickup_latitude>90) | (df.pickup_latitude< -90)|
2                               (df.dropoff_latitude>90) | (df.dropoff_latitude< -90) |
3                               (df.pickup_longitude>180) | (df.pickup_longitude< -180)|
4                               (df.dropoff_latitude>90) | (df.dropoff_latitude< -90)
5                               ]
6
7
```

In [45]:

```
1 df.drop(incorrect_coordinates, inplace = True, errors ='ignore')
```

In [46]:

```
1 df.head()
```

Out[46]:

	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count	hour	day	month	year	dayofweek	dist_1
0	7.5	-73.999817	40.738354	-73.999512	40.723217	1.0	19	7	5	2015	3	1275
1	7.7	-73.994355	40.728225	-73.994710	40.750325	1.0	20	17	7	2009	4	1275
2	12.9	-74.005043	40.740770	-73.962565	40.772647	1.0	21	24	8	2009	0	1275
3	5.3	-73.976124	40.790844	-73.965316	40.803349	3.0	8	26	6	2009	4	1275
4	16.0	-73.929786	40.744085	-73.973082	40.761247	3.5	17	28	8	2014	3	1275

In [47]:

```
1 df.isnull().sum()
```

Out[47]:

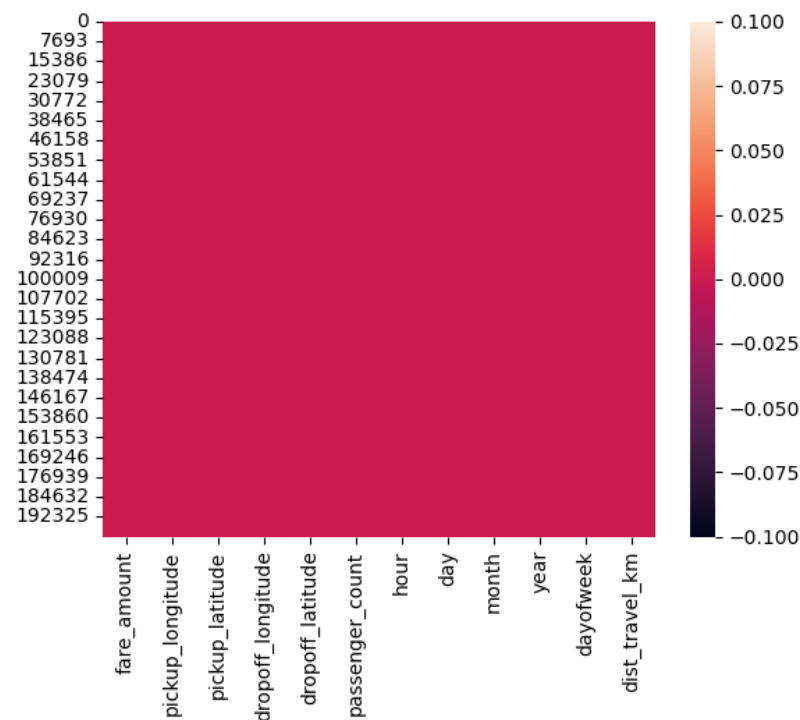
```
fare_amount      0
pickup_longitude  0
pickup_latitude  0
dropoff_longitude 0
dropoff_latitude 0
passenger_count  0
hour             0
day              0
month            0
year             0
dayofweek        0
dist_travel_km   0
dtype: int64
```

In [48]:

```
1 sns.heatmap(df.isnull())
```

Out[48]:

<AxesSubplot:>



In [49]:

```
1 corr = df.corr()
```

In [50]:

```
1 corr
```

Out[50]:

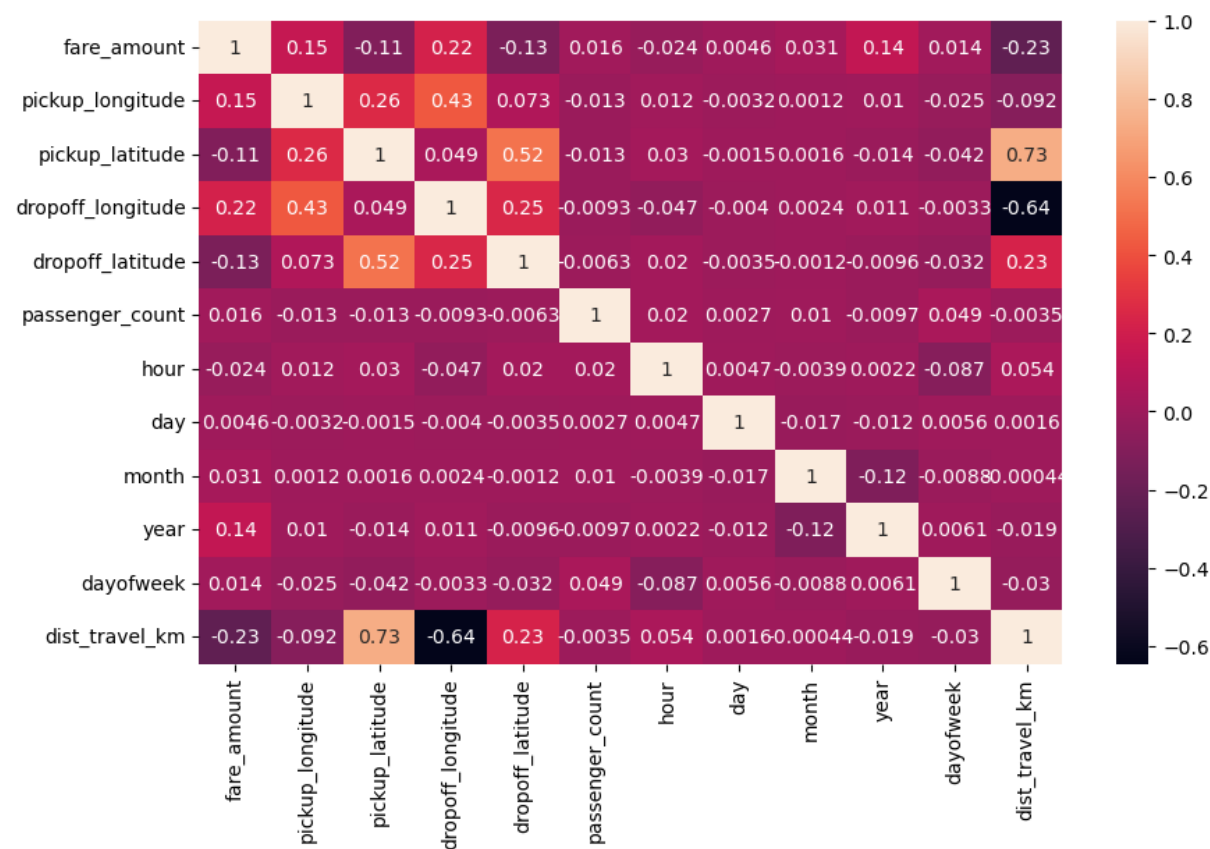
	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count	hour	day	mo
fare_amount	1.000000	0.154069	-0.110842	0.218675	-0.125898	0.015778	-0.023623	0.004534	0.030
pickup_longitude	0.154069	1.000000	0.259497	0.425619	0.073290	-0.013213	0.011579	-0.003204	0.001
pickup_latitude	-0.110842	0.259497	1.000000	0.048889	0.515714	-0.012889	0.029681	-0.001553	0.001
dropoff_longitude	0.218675	0.425619	0.048889	1.000000	0.245667	-0.009303	-0.046558	-0.004007	0.002
dropoff_latitude	-0.125898	0.073290	0.515714	0.245667	1.000000	-0.006308	0.019783	-0.003479	-0.001
passenger_count	0.015778	-0.013213	-0.012889	-0.009303	-0.006308	1.000000	0.020274	0.002712	0.010
hour	-0.023623	0.011579	0.029681	-0.046558	0.019783	0.020274	1.000000	0.004677	-0.003
day	0.004534	-0.003204	-0.001553	-0.004007	-0.003479	0.002712	0.004677	1.000000	-0.017
month	0.030817	0.001169	0.001562	0.002391	-0.001193	0.010351	-0.003926	-0.017360	1.000
year	0.141277	0.010198	-0.014243	0.011346	-0.009603	-0.009749	0.002156	-0.012170	-0.115
dayofweek	0.013652	-0.024652	-0.042310	-0.003336	-0.031919	0.048550	-0.086947	0.005617	-0.008
dist_travel_km	-0.233982	-0.091832	0.731839	-0.644884	0.227001	-0.003515	0.054475	0.001546	-0.000

In [100]:

```
1 fig,axis = plt.subplots(figsize= (10,6))
2 sns.heatmap(df.corr(), annot = True)
```

Out[100]:

<Axes: >



In [51]:

```
1 df.dtypes
```

Out[51]:

```
fare_amount      float64
pickup_longitude  float64
pickup_latitude   float64
dropoff_longitude float64
dropoff_latitude  float64
passenger_count   float64
hour              int64
day               int64
month             int64
year              int64
dayofweek         int64
dist_travel_km    float64
dtype: object
```

In [59]:

```
1 x = df[['pickup_longitude', 'pickup_latitude', 'dropoff_longitude', 'passenger_count', 'hour', 'day', 'month', 'year', 'dayofweek']]
```

In [60]:

```
1 y = df['fare_amount']
```

In [61]:

```
1 from sklearn.model_selection import train_test_split
2 x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.33)
```

In [62]:

```
1 from sklearn.linear_model import LinearRegression
2 regression = LinearRegression()
```

In [63]:

```
1 regression.fit(x_train,y_train)
```

Out[63]:

LinearRegression()

In [65]:

```
1 regression.intercept_
```

Out[65]:

2810.586867670241

In [66]:

```
1 regression.coef_
```

Out[66]:

```
array([ 3.36184179e+01, -1.54274575e+06,  1.54272668e+06,  9.24908560e-02,
        1.12392328e-02,  2.68517483e-03,  6.27287508e-02,  3.78329486e-01,
        1.17565459e-02,  1.38740317e+04])
```

In [67]:

```
1 prediction = regression.predict(x_test)
```

In [68]:

```
1 print(prediction)
```

```
[23.55019657 13.25135417  7.99420584 ...  9.33548389  8.53506384
  9.26098324]
```

In [69]:

```
1 y_test
```

Out[69]:

```
168553    22.25
24295     12.50
54374      3.70
77398     16.00
80636      4.00
...
114623     6.10
194330     14.50
52536      4.50
136642     12.00
84566      6.00
Name: fare_amount, Length: 66000, dtype: float64
```

In [72]:

```
1 from sklearn.metrics import r2_score
```

In [73]:

```
1 r2_score(y_test,prediction)
```

Out[73]:

0.41928436137240455

In [74]:

```
1 from sklearn.metrics import mean_squared_error
```

In [75]:

```
1 MSE = mean_squared_error(y_test,prediction)
```

In [76]:

```
1 MSE
```

Out[76]:

16.961152324796327

In [ ]:

```
1
```