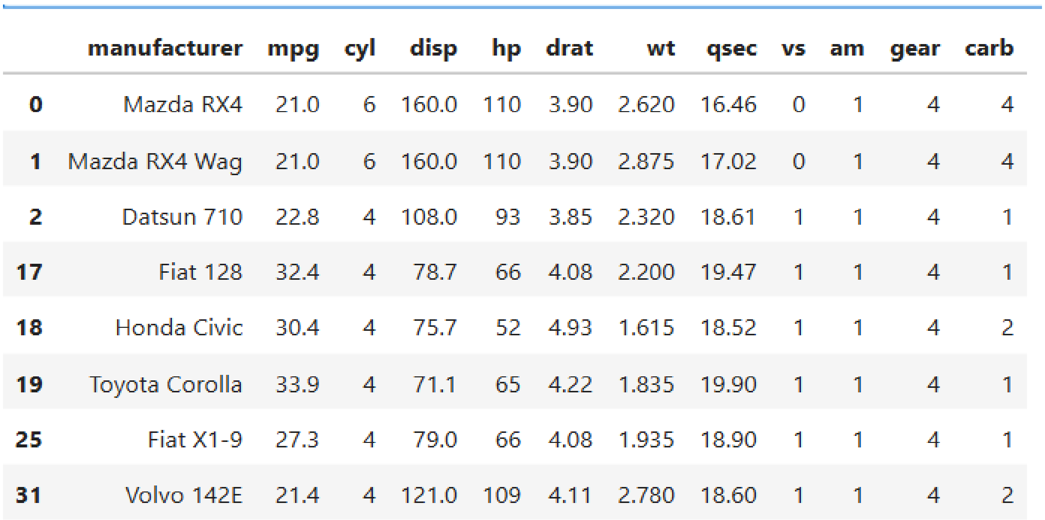
DAY-25

OCTOBER-13

1. Extract 4 gear automatic car data

```
d2[(d2['gear']==4) & (d2['am']==1)]
```

o/p:



	manufacturer	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
0	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
1	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
2	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
17	Fiat 128	32.4	4	78.7	66	4.08	2.200	19.47	1	1	4	1
18	Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	2
19	Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.90	1	1	4	1
25	Fiat X1-9	27.3	4	79.0	66	4.08	1.935	18.90	1	1	4	1
31	Volvo 142E	21.4	4	121.0	109	4.11	2.780	18.60	1	1	4	2

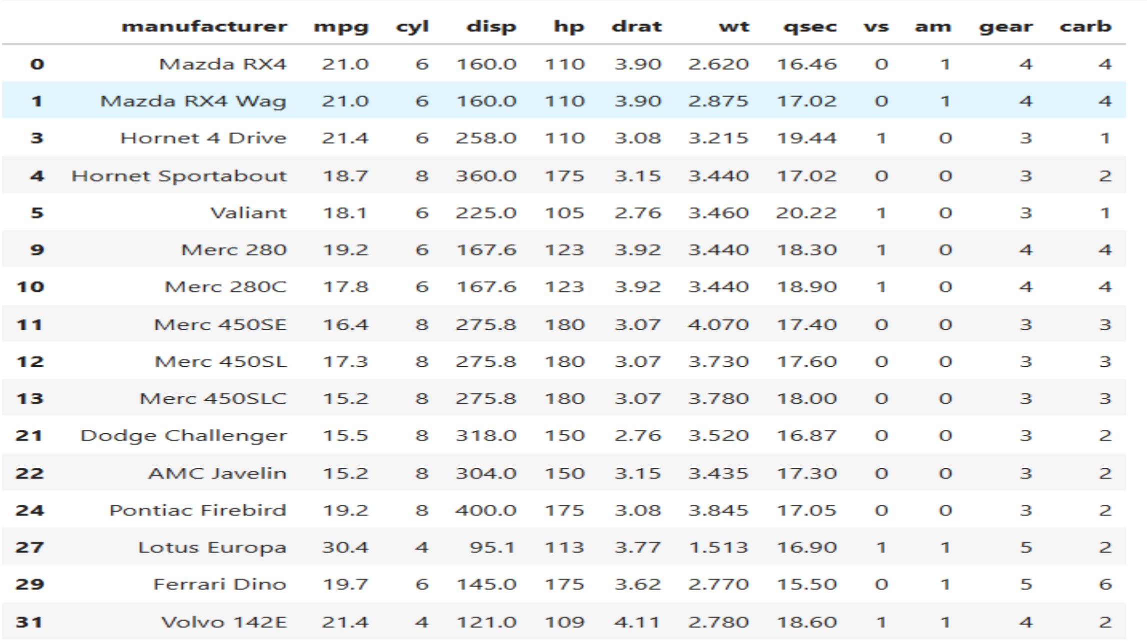
2. Extract data of cars who's hp is ranges between 100 to 200

```
d2[(d2['hp'] >= 100) & (d2['hp'] <= 200)]
```

or

```
d2[d2['hp'].between(100,200)]
```

o/p:

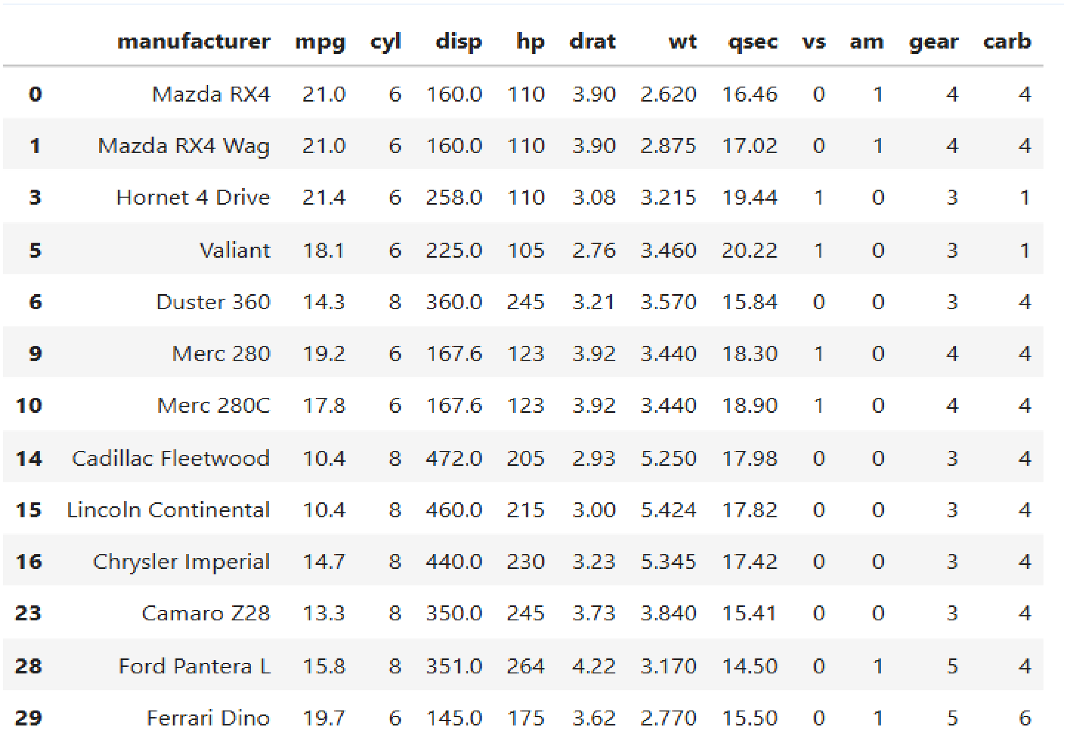


	manufacturer	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
0	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
1	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
3	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
4	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
5	Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
9	Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
10	Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
11	Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3
12	Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	0	3	3
13	Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3
21	Dodge Challenger	15.5	8	318.0	150	2.76	3.520	16.87	0	0	3	2
22	AMC Javelin	15.2	8	304.0	150	3.15	3.435	17.30	0	0	3	2
24	Pontiac Firebird	19.2	8	400.0	175	3.08	3.845	17.05	0	0	3	2
27	Lotus Europa	30.4	4	95.1	113	3.77	1.513	16.90	1	1	5	2
29	Ferrari Dino	19.7	6	145.0	175	3.62	2.770	15.50	0	1	5	6
31	Volvo 142E	21.4	4	121.0	109	4.11	2.780	18.60	1	1	4	2

3. Extract the data which have 6 cyl or 4 carb

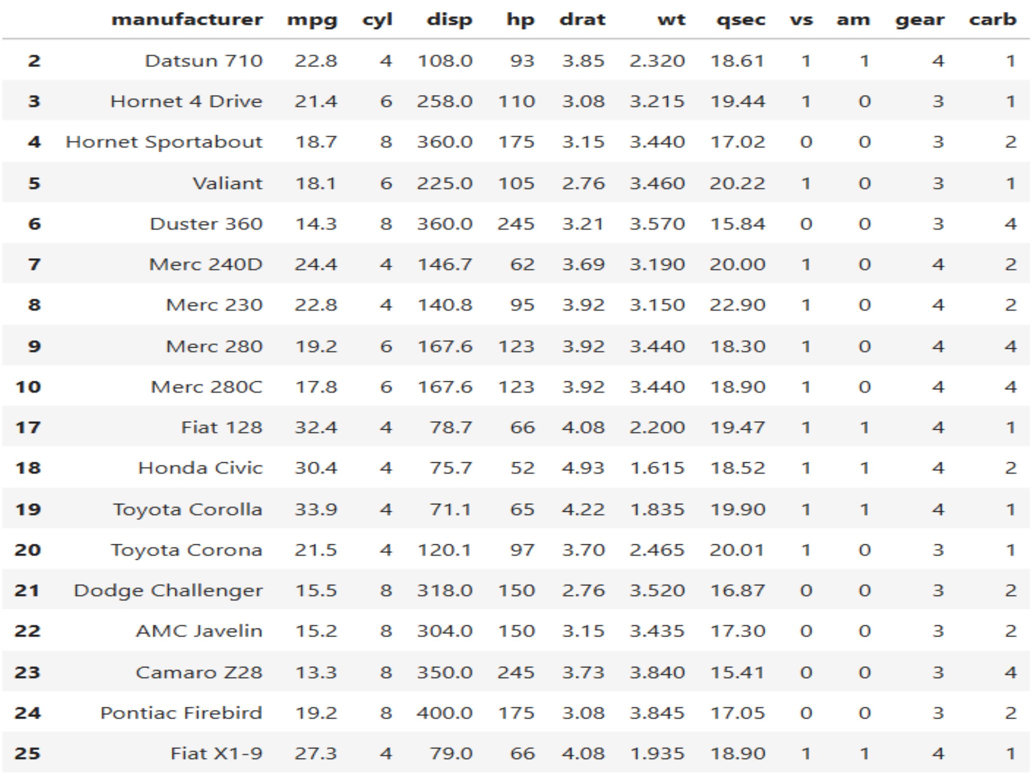
```
d2[(d2['cyl']==6) | (d2['carb']==4)]
```

o/p:



	manufacturer	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
0	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
1	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
3	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
5	Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
6	Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
9	Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
10	Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
14	Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4
15	Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	0	3	4
16	Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.42	0	0	3	4
23	Camaro Z28	13.3	8	350.0	245	3.73	3.840	15.41	0	0	3	4
28	Ford Pantera L	15.8	8	351.0	264	4.22	3.170	14.50	0	1	5	4
29	Ferrari Dino	19.7	6	145.0	175	3.62	2.770	15.50	0	1	5	6

4. Extract the records of cars who's disp range is 300 to 400 or vs should be 1
d2[(d2['disp'].between(300,400)) | (d2['vs']==1)]
o/p:



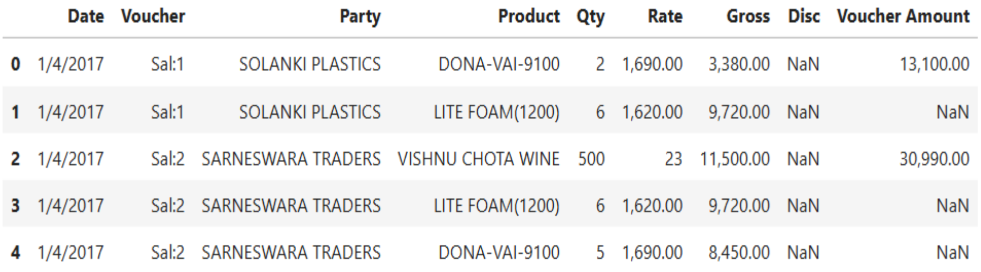
	manufacturer	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
2	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
3	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
4	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
5	Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
6	Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
7	Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
8	Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
9	Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
10	Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
17	Fiat 128	32.4	4	78.7	66	4.08	2.200	19.47	1	1	4	1
18	Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	2
19	Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.90	1	1	4	1
20	Toyota Corona	21.5	4	120.1	97	3.70	2.465	20.01	1	0	3	1
21	Dodge Challenger	15.5	8	318.0	150	2.76	3.520	16.87	0	0	3	2
22	AMC Javelin	15.2	8	304.0	150	3.15	3.435	17.30	0	0	3	2
23	Camaro Z28	13.3	8	350.0	245	3.73	3.840	15.41	0	0	3	4
24	Pontiac Firebird	19.2	8	400.0	175	3.08	3.845	17.05	0	0	3	2
25	Fiat X1-9	27.3	4	79.0	66	4.08	1.935	18.90	1	1	4	1

Insurance and sales data

1. Read the files using pandas
   - sale1=pd.read_csv(r"C:\Users\INDUPRIYA\OneDrive\Attachments\Desktop\MLP\Sales Transactions-2017.csv")
   - sale2=pd.read_csv(r"C:\Users\INDUPRIYA\OneDrive\Attachments\Desktop\MLP\Sales Transactions-2018.csv")
   - sale3=pd.read_csv(r"C:\Users\INDUPRIYA\OneDrive\Attachments\Desktop\MLP\Sales Transactions-2019.csv")
2. As the three data files have same data we can combine all the three files
3. To combine them we need to make sure they have same columns and in same order.

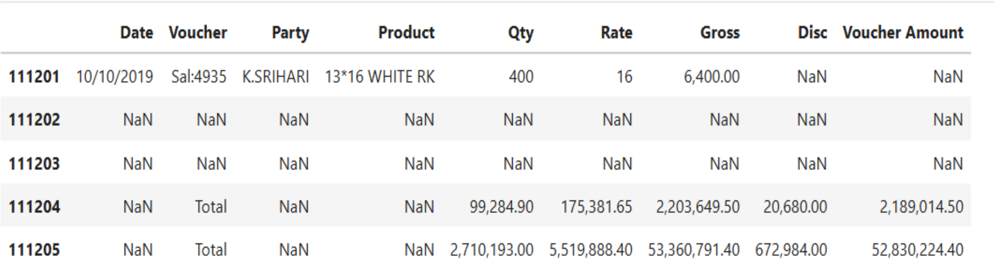
   Final_sale = pd.concat([sale1,sale2,sale3],ignore_index = True)
4. Initial analysis of data:

   The data consists of date hence it is a time frame data.
   - Final_sale.shape : (111206, 9)
   - Final_sale.head()



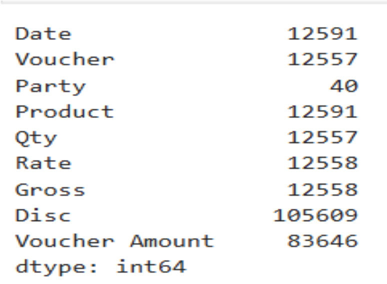
	Date	Voucher	Party	Product	Qty	Rate	Gross	Disc	Voucher Amount
0	1/4/2017	Sal:1	SOLANKI PLASTICS	DONA-VAI-9100	2	1,690.00	3,380.00	NaN	13,100.00
1	1/4/2017	Sal:1	SOLANKI PLASTICS	LITE FOAM(1200)	6	1,620.00	9,720.00	NaN	NaN
2	1/4/2017	Sal:2	SARNESWARA TRADERS	VISHNU CHOTA WINE	500	23	11,500.00	NaN	30,990.00
3	1/4/2017	Sal:2	SARNESWARA TRADERS	LITE FOAM(1200)	6	1,620.00	9,720.00	NaN	NaN
4	1/4/2017	Sal:2	SARNESWARA TRADERS	DONA-VAI-9100	5	1,690.00	8,450.00	NaN	NaN

   - Final_sale.tail()



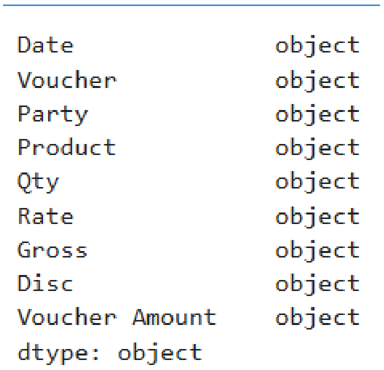
	Date	Voucher	Party	Product	Qty	Rate	Gross	Disc	Voucher Amount
111201	10/10/2019	Sal:4935	K.SRIHARI	13*16 WHITE RK	400	16	6,400.00	NaN	NaN
111202	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
111203	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
111204	NaN	Total	NaN	NaN	99,284.90	175,381.65	2,203,649.50	20,680.00	2,189,014.50
111205	NaN	Total	NaN	NaN	2,710,193.00	5,519,888.40	53,360,791.40	672,984.00	52,830,224.40

   - Final_sale.isna().sum()



```
Date                  12591
Voucher               12557
Party                    40
Product               12591
Qty                   12557
Rate                  12558
Gross                 12558
Disc                 105609
Voucher Amount        83646
dtype: int64
```

   - Final_sale.dtypes



```
Date              object
Voucher           object
Party             object
Product           object
Qty               object
Rate              object
Gross             object
Disc              object
Voucher Amount    object
dtype: object
```

- What are the procedures we can follow to work on null values?

  Use dropna() to remove rows or columns with null values. Parameters can specify whether to drop rows or columns and the conditions for dropping (e.g., any or all nulls).

  Use fillna() to replace null values with a specified value or by using methods such as forward fill (method='ffill'), backward fill (method='bfill'), or interpolations.

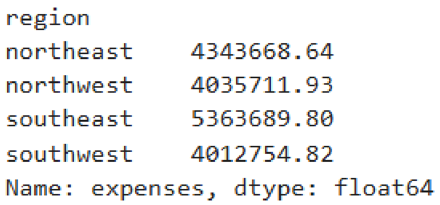
- How to change object data type to numerical data type
  Using pd.to_numeric()
  The pd.to_numeric() function is used to convert non-numeric objects (like strings) into numeric types (int or float).
  Using astype()
  The astype() method converts a Series or DataFrame column to a specified data type. It is useful when you are certain the conversion will succeed.

5. On Insurance data, find the following
   1. Each region wise total expenses

   2. Gender wise average bmi and expenses

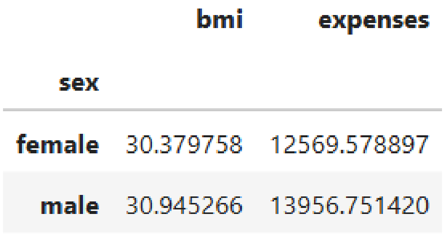
   3. Each region,each children and smoker class wise total expenses

   Loading the data : insurance_data = pd.read_csv(r"C:\Users\INDU PRIYA\OneDrive\Attachments\Desktop\MLP\insurance.csv")
   1. Each region wise total expenses:
      region_expenses = insurance_data.groupby('region')['expenses'].sum()



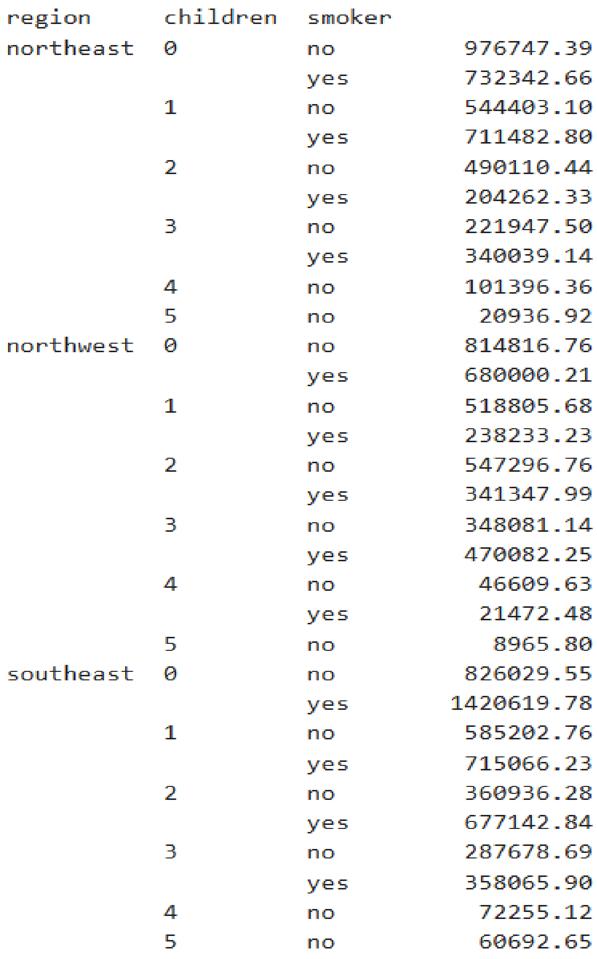
```
region
northeast    4343668.64
northwest    4035711.93
southeast    5363689.80
southwest    4012754.82
Name: expenses, dtype: float64
```

2. Gender wise average bmi and expenses:
   gender_avg = insurance_data.groupby('sex')[['bmi', 'expenses']].mean()



	bmi	**expenses**
sex		
female	30.379758	12569.578897
male	30.945266	13956.751420

3. Each region,each children and smoker class wise total expenses:
   regionwise_class_expenses = insurance_data.groupby(['region', 'children', 'smoker'])['expenses'].sum()



```
region     children  smoker
northeast  0         no         976747.39
                     yes        732342.66
           1         no         544403.10
                     yes        711482.80
           2         no         490110.44
                     yes        204262.33
           3         no         221947.50
                     yes        340039.14
           4         no         101396.36
           5         no          20936.92
northwest  0         no         814816.76
                     yes        680000.21
           1         no         518805.68
                     yes        238233.23
           2         no         547296.76
                     yes        341347.99
           3         no         348081.14
                     yes        470082.25
           4         no          46609.63
                     yes         21472.48
           5         no           8965.80
southeast  0         no         826029.55
                     yes       1420619.78
           1         no         585202.76
                     yes        715066.23
           2         no         360936.28
                     yes        677142.84
           3         no         287678.69
                     yes        358065.90
           4         no          72255.12
           5         no          60692.65
```