import pandas as pd

## First step(Import Pandas)

pd.read\_csv("/content/PQ Exercise 1d.csv")

$\overline{\Rightarrow}$		Metric	Store	Cat	01/01/2017	02/01/2017	03/01/2017	Q1 2017	04/01/2017	05/01/2017	06/01/2017	• • •	
	0	Sales	1	1	NaN	NaN	NaN	NaN	19403.5400	21827.9000	21043.3900		(
	1	NaN	1	2	50605.2700	44682.7400	47928.8900	143216.9000	44292.8700	48397.9800	43751.9400	1	ľ
	2	NaN	1	3	13740.1200	10887.8400	11523.4700	36151.4300	11135.1700	12275.5800	10123.4500		;
	3	NaN	1	4	39954.0400	35351.2100	36826.9500	112132.2000	34660.1600	38086.1900	32668.6700	1	1(
	4	NaN	2	1	35034.0600	60483.7000	58221.5200	153739.2800	25962.3200	27372.0500	28660.8700		{
	5	NaN	2	2	74661.1600	65487.4600	70853.5800	211002.2000	64963.9000	68428.6400	66622.0300	2	<u>2</u> (
	6	NaN	2	3	16873.2000	13821.0100	14607.2800	45301.4900	15635.9500	14895.9600	13061.5600		4
	7	NaN	2	4	47681.9600	44197.9500	46131.1400	138011.0500	42126.7100	46937.8100	42489.2100	1	1;
	8	Margin	1	1	NaN	NaN	NaN	NaN	0.5432	0.5432	0.5432		
	9	NaN	1	2	0.5542	0.5542	0.5542	0.5542	0.5542	0.5542	0.5542		
	10	NaN	1	3	0.5212	0.5212	0.5212	0.5212	0.5212	0.5212	0.5212		
	11	NaN	1	4	0.5462	0.5462	0.5462	0.5462	0.5462	0.5462	0.5462		
	12	NaN	2	1	0.5432	0.5432	0.5432	0.5432	0.5432	0.5432	0.5432		
	13	NaN	2	2	0.5542	0.5542	0.5542	0.5542	0.5542	0.5542	0.5542		
	14	NaN	2	3	0.5212	0.5212	0.5212	0.5212	0.5212	0.5212	0.5212		
	15	NaN	2	4	0.5462	0.5462	0.5462	0.5462	0.5462	0.5462	0.5462		

16 rows × 37 columns

Second Step(Copy Path file)

q = pd.read\_csv("/content/PQ Exercise 1d.csv")

Third Step(Assign a variable to our Table Link)

q

<b>→</b>		Metric	Store	Cat	01/01/2017	02/01/2017	03/01/2017	Q1 2017	04/01/2017	05/01/2017	06/01/2017	•••	
	0	Sales	1	1	NaN	NaN	NaN	NaN	19403.5400	21827.9000	21043.3900		(
	1	NaN	1	2	50605.2700	44682.7400	47928.8900	143216.9000	44292.8700	48397.9800	43751.9400		1:
	2	NaN	1	3	13740.1200	10887.8400	11523.4700	36151.4300	11135.1700	12275.5800	10123.4500		;
	3	NaN	1	4	39954.0400	35351.2100	36826.9500	112132.2000	34660.1600	38086.1900	32668.6700		1(
	4	NaN	2	1	35034.0600	60483.7000	58221.5200	153739.2800	25962.3200	27372.0500	28660.8700		{
	5	NaN	2	2	74661.1600	65487.4600	70853.5800	211002.2000	64963.9000	68428.6400	66622.0300		20
	6	NaN	2	3	16873.2000	13821.0100	14607.2800	45301.4900	15635.9500	14895.9600	13061.5600		4
	7	NaN	2	4	47681.9600	44197.9500	46131.1400	138011.0500	42126.7100	46937.8100	42489.2100		13
	8	Margin	1	1	NaN	NaN	NaN	NaN	0.5432	0.5432	0.5432		
	9	NaN	1	2	0.5542	0.5542	0.5542	0.5542	0.5542	0.5542	0.5542		
	10	NaN	1	3	0.5212	0.5212	0.5212	0.5212	0.5212	0.5212	0.5212		
	11	NaN	1	4	0.5462	0.5462	0.5462	0.5462	0.5462	0.5462	0.5462		
	12	NaN	2	1	0.5432	0.5432	0.5432	0.5432	0.5432	0.5432	0.5432		
	13	NaN	2	2	0.5542	0.5542	0.5542	0.5542	0.5542	0.5542	0.5542		
	14	NaN	2	3	0.5212	0.5212	0.5212	0.5212	0.5212	0.5212	0.5212		
	15	NaN	2	4	0.5462	0.5462	0.5462	0.5462	0.5462	0.5462	0.5462		

16 rows × 37 columns

Fourth step is to call the variable(q) and inspect our Table

# q.head()

<b>→</b>		Metric	Store	Cat	01/01/2017	02/01/2017	03/01/2017	Q1 2017	04/01/2017	05/01/2017	06/01/2017	 Q2
	0	Sales	1	1	NaN	NaN	NaN	NaN	19403.54	21827.90	21043.39	 622
	1	NaN	1	2	50605.27	44682.74	47928.89	143216.90	44292.87	48397.98	43751.94	 13644
	2	NaN	1	3	13740.12	10887.84	11523.47	36151.43	11135.17	12275.58	10123.45	 3350
	3	NaN	1	4	39954.04	35351.21	36826.95	112132.20	34660.16	38086.19	32668.67	 1054 <sup>-</sup>
	4	NaN	2	1	35034.06	60483.70	58221.52	153739.28	25962.32	27372.05	28660.87	 8199

5 rows × 37 columns

Checking top 5 rows

## q.tail()

₹		Metric	Store	Cat	01/01/2017	02/01/2017	03/01/2017	Q1 2017	04/01/2017	05/01/2017	06/01/2017	 Q2 2018
	11	NaN	1	4	0.5462	0.5462	0.5462	0.5462	0.5462	0.5462	0.5462	 0.5462
	12	NaN	2	1	0.5432	0.5432	0.5432	0.5432	0.5432	0.5432	0.5432	 0.5432
	13	NaN	2	2	0.5542	0.5542	0.5542	0.5542	0.5542	0.5542	0.5542	 0.5542
	14	NaN	2	3	0.5212	0.5212	0.5212	0.5212	0.5212	0.5212	0.5212	 0.5212
	15	NaN	2	4	0.5462	0.5462	0.5462	0.5462	0.5462	0.5462	0.5462	 0.5462

5 rows × 37 columns

#### Checking bottom 5 rows

```
q.columns.values
```

```
array(['Metric', 'Store', 'Cat', '01/01/2017', '02/01/2017', '03/01/2017', 'Q1 2017', '04/01/2017', '05/01/2017', '06/01/2017', 'Q2 2017', '07/01/2017', '08/01/2017', '09/01/2017', 'Q3 2017', '10/01/2017', '11/01/2017', '12/01/2017', 'Q4 2017', '2017', '01/01/2018', '02/01/2018', '03/01/2018', 'Q1 2018', '04/01/2018', '05/01/2018', '06/01/2018', 'Q2 2018', '07/01/2018', '08/01/2018', '09/01/2018', 'Q3 2018', '10/01/2018', '11/01/2018', '12/01/2018', 'Q4 2018', '2018'], dtype=object)
```

#### Checking for the Structure of our Table

#### q.describe()

<del></del>		Store	Cat	01/01/2017	02/01/2017	03/01/2017	Q1 2017	04/01/2017	05/01/2017	06
	count	16.000000	16.000000	14.000000	14.000000	14.000000	14.000000	16.000000	16.000000	1
	mean	1.500000	2.500000	19896.685457	19636.835457	20435.472600	59968.452600	16136.559350	17389.152475	1615
	std	0.516398	1.154701	25059.413452	24896.231649	25856.264584	75090.505316	20673.575131	22269.116083	2095
	min	1.000000	1.000000	0.521200	0.521200	0.521200	0.521200	0.521200	0.521200	
	25%	1.000000	1.750000	0.546200	0.546200	0.546200	0.546200	0.545450	0.545450	
	50%	1.500000	2.500000	6870.337100	5444.197100	5762.012100	18075.992100	5567.862100	6138.067100	506
	75%	2.000000	3.250000	38724.045000	41986.265000	43805.092500	131541.337500	28136.780000	30050.585000	2966
	max	2.000000	4.000000	74661.160000	65487.460000	70853.580000	211002.200000	64963.900000	68428.640000	6662

# 8 rows × 36 columns

#### Describe the Table

q.melt(id\_vars = ["Metric","Store","Cat"], var\_name = "Date", value\_name = "Sales")

<b>₹</b>		Metric	Store	Cat	Date	Sales	
	0	Sales	1	1	01/01/2017	NaN	11.
	1	NaN	1	2	01/01/2017	50605.2700	
	2	NaN	1	3	01/01/2017	13740.1200	
	3	NaN	1	4	01/01/2017	39954.0400	
	4	NaN	2	1	01/01/2017	35034.0600	
	539	NaN	1	4	2018	0.5462	
	540	NaN	2	1	2018	0.5432	
	541	NaN	2	2	2018	0.5542	
	542	NaN	2	3	2018	0.5212	
	543	NaN	2	4	2018	0.5462	

544 rows × 5 columns

Now is time to Pivot our table to proper format using "MELT" which is same as Pivoting in Python(Listing the Columns to Pivot and the Values you want to use)

```
q = q.melt(id_vars = ["Metric","Store","Cat"], var_name = "Date", value_name = "Sales")
```

Let's assign the Output to our original Table

q

<b>→</b>		Metric	Store	Cat	Date	Sales
	0	Sales	1	1	01/01/2017	NaN
	1	NaN	1	2	01/01/2017	50605.2700
	2	NaN	1	3	01/01/2017	13740.1200
	3	NaN	1	4	01/01/2017	39954.0400
	4	NaN	2	1	01/01/2017	35034.0600
	539	NaN	1	4	2018	0.5462
	540	NaN	2	1	2018	0.5432
	541	NaN	2	2	2018	0.5542
	542	NaN	2	3	2018	0.5212
	543	NaN	2	4	2018	0.5462
		_				

544 rows × 5 columns

Next steps: Generate code with q

View recommended plots

New interactive sheet

print(q.to\_string)

<b>→</b>	<bou< th=""><th>nd method</th><th>Data</th><th>rame</th><th>.to_string o</th><th>f Metric</th><th>Store</th><th>Cat</th><th>Date</th><th>Sales</th></bou<>	nd method	Data	rame	.to_string o	f Metric	Store	Cat	Date	Sales
_	0	Sales	1	1	01/01/2017	NaN				
	1	NaN	1	2	01/01/2017	50605.2700				
	2	NaN	1	3	01/01/2017	13740.1200				
	3	NaN	1	4	01/01/2017	39954.0400				
	4	NaN	2	1	01/01/2017	35034.0600				
	539	NaN	1	4	2018	0.5462				
	540	NaN	2	1	2018	0.5432				
	541	NaN	2	2	2018	0.5542				
	542	NaN	2	3	2018	0.5212				
	543	NaN	2	4	2018	0.5462				

[544 rows x 5 columns]>

Let print our table to see the full table

q.Metric.ffill()

<del>-</del>		Metric
	0	Sales
	1	Sales
	2	Sales
	3	Sales
	4	Sales
	539	Margin
	540	Margin
	541	Margin
	542	Margin
	543	Margin
	544 rc	ws × 1 columns
	dtype	: object

Having seen the Nulls now its time to fill it up in our dataset using "Forward\_Fill" in Python to "Fill\_Down"

```
q.Metric.ffill(inplace = True)
```

/tmp/ipython-input-1434120356.py:1: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on whi

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' o

q.Metric.ffill(inplace = True)

Let's assign the output to our original table

q

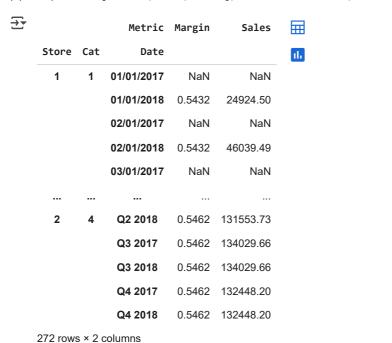
	Metric	Store	Cat	Date	Sales	#
0	Sales	1	1	01/01/2017	NaN	ılı
1	Sales	1	2	01/01/2017	50605.2700	+/
2	Sales	1	3	01/01/2017	13740.1200	
3	Sales	1	4	01/01/2017	39954.0400	
4	Sales	2	1	01/01/2017	35034.0600	
539	Margin	1	4	2018	0.5462	
540	Margin	2	1	2018	0.5432	
541	Margin	2	2	2018	0.5542	
542	Margin	2	3	2018	0.5212	
543	Margin	2	4	2018	0.5462	
544 rc	ws × 5 cc	lumns				

Let's call our Table



This is another way of calling the column to check the changes

q.pivot(index = ["Store","Cat","Date"], columns = "Metric", values = "Sales")

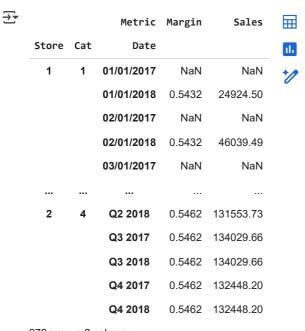


Now is time to Pivot the "SALES & MARGIN COLUMN" to make each have their separate column

```
q = q.pivot(index = ["Store","Cat","Date"], columns = "Metric", values = "Sales")
```

Now is time to assign our output to our original Table

q

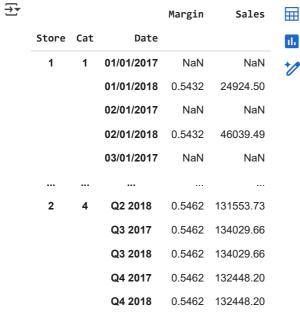


272 rows × 2 columns

q.columns.name = None

Now is time to remove the Name "Metric" as a Column but not the entire column itself

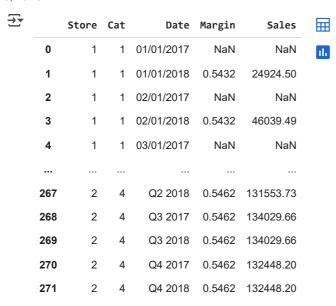
q



272 rows × 2 columns

Next steps: Generate code with q View recommended plots New interactive sheet

q.reset\_index()



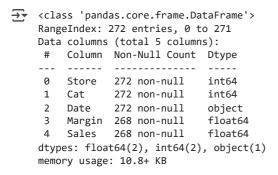
272 rows × 5 columns

Looking at our index after the changes scattered, so we need to reset it back to a proper index

```
q = q.reset_index()
```

Now let's assign the output to our original Table

#### q.info()



Let's now check for our Table and its columns datatype using info()

```
pd.to_datetime(q.Date, errors = "coerce", format='mixed')
```

₹		Date
	0	2017-01-01
	1	2018-01-01
	2	2017-02-01
	3	2018-02-01
	4	2017-03-01
	267	NaT
	268	NaT
	269	NaT
	270	NaT
	271	NaT
	272 rc	ws × 1 columns
	dtype	: datetime64[ns]

The original datatype of our "Date\_Column" is "Object", Let's change to a proper datatype ie "Datetime"

```
q.Date = pd.to_datetime(q.Date, errors = "coerce", format='mixed')
```

Now is time to effect the output on our original date\_column

q

<u>-</u>		Store	Cat	Date	Margin	Sales	E
	0	1	1	2017-01-01	NaN	NaN	
	1	1	1	2018-01-01	0.5432	24924.50	+
	2	1	1	2017-02-01	NaN	NaN	
	3	1	1	2018-02-01	0.5432	46039.49	
	4	1	1	2017-03-01	NaN	NaN	
	267	2	4	NaT	0.5462	131553.73	
	268	2	4	NaT	0.5462	134029.66	
	269	2	4	NaT	0.5462	134029.66	
	270	2	4	NaT	0.5462	132448.20	
	271	2	4	NaT	0.5462	132448.20	
2	272 rc	ws × 5 c	columr	ns			

q.dropna()



205 rows × 5 columns

It's time to replace the "NULL\_VALUES" In our Table

q.dropna(inplace = True)

Let's effect the result or output in our original Table

q

₹		Store	Cat	Date	Margin	Sales	<b>=</b>
	1	1	1	2018-01-01	0.5432	24924.50	11.
	3	1	1	2018-02-01	0.5432	46039.49	+//
	5	1	1	2018-03-01	0.5432	41595.55	
	6	1	1	2017-04-01	0.5432	19403.54	
	7	1	1	2018-04-01	0.5432	19403.54	
	259	2	4	2018-11-01	0.5462	43463.55	
	260	2	4	2017-12-01	0.5462	43703.76	
	261	2	4	2018-12-01	0.5462	43703.76	
	262	2	4	2017-01-01	0.5462	536042.64	
	263	2	4	2018-01-01	0.5462	536042.64	

205 rows × 5 columns

Next steps: ( Generate code with q )

View recommended plots

New interactive sheet

q.reset\_index(drop = True)



205 rows × 5 columns

Let's "RESET\_INDEX"

q.reset\_index(drop = True, inplace = True)

Let's effect the Changes in our Original Table

q

<b>→</b>		Store	Cat	Date	Margin	Sales	<b>=</b>
	0	1	1	2018-01-01	0.5432	24924.50	11.
	1	1	1	2018-02-01	0.5432	46039.49	+/
	2	1	1	2018-03-01	0.5432	41595.55	
	3	1	1	2017-04-01	0.5432	19403.54	
	4	1	1	2018-04-01	0.5432	19403.54	
	200	2	4	2018-11-01	0.5462	43463.55	
	201	2	4	2017-12-01	0.5462	43703.76	
	202	2	4	2018-12-01	0.5462	43703.76	
	203	2	4	2017-01-01	0.5462	536042.64	
	204	2	4	2018-01-01	0.5462	536042.64	

205 rows × 5 columns

Next steps: (Generate code with q)

View recommended plots

New interactive sheet

q.Sales\*q.Margin

<b>→</b>		0
	0	13538.988400
	1	25008.650968
	2	22594.702760
	3	10540.002928
	4	10540.002928
	200	23739.791010
	201	23870.993712
	202	23870.993712
	203	292786.489968
	204	292786.489968
	205 rc	ows × 1 columns

dtype: float64

OUR TABLE IS PROPERLY\_CLEANED(TIME FOR REAL LIFE ANALYSIS). QUSTION 1: CALCULATE PROFIT AND ADD THE COLUMN IN THE ORIGINAL TABLE(ANSWER ABOVE). "PROFIT = SALES \* MARGIN"

```
q["Profit"] = q.Sales * q.Margin
```

Now let's add the column to our original Table using the above format

q["Profit"] = q["Profit"].round(2)

Let's "Round" the Number to 2 dec

q

<b>→</b>		Store	Cat	Date	Margin	Sales	Profit	
	0	1	1	2018-01-01	0.5432	24924.50	13538.99	11.
	1	1	1	2018-02-01	0.5432	46039.49	25008.65	+/
	2	1	1	2018-03-01	0.5432	41595.55	22594.70	
	3	1	1	2017-04-01	0.5432	19403.54	10540.00	
	4	1	1	2018-04-01	0.5432	19403.54	10540.00	
	200	2	4	2018-11-01	0.5462	43463.55	23739.79	
	201	2	4	2017-12-01	0.5462	43703.76	23870.99	
	202	2	4	2018-12-01	0.5462	43703.76	23870.99	
	203	2	4	2017-01-01	0.5462	536042.64	292786.49	
	204	2	4	2018-01-01	0.5462	536042.64	292786.49	
	205 rc	ows × 6 d	columr	าร				

Next steps: ( Generate code with q View recommended plots

New interactive sheet

Let's check the original Table for the result

q.Sales - q.Profit

<b>→</b>		0
	0	11385.51
	1	21030.84
	2	19000.85
	3	8863.54
	4	8863.54
	200	19723.76
	201	19832.77
	202	19832.77
	203	243256.15
	204	243256.15
	205 rc	ws × 1 columns

dtype: float64

Now is time to "Calculate the Cost\_Of\_Goods\_Sold". Cost = Sales - Profit

q["COGs"] = q.Sales - q.Profit

It's time to add the Column in our original dataset

q["COGs"] = q["COGs"].round(2)

Let's "Round" the Number to 2 dec

q

<b>→</b>		Store	Cat	Date	Margin	Sales	Profit	COGs	$\blacksquare$
	0	1	1	2018-01-01	0.5432	24924.50	13538.99	11385.51	11.
	1	1	1	2018-02-01	0.5432	46039.49	25008.65	21030.84	+/
	2	1	1	2018-03-01	0.5432	41595.55	22594.70	19000.85	_
	3	1	1	2017-04-01	0.5432	19403.54	10540.00	8863.54	
	4	1	1	2018-04-01	0.5432	19403.54	10540.00	8863.54	
	200	2	4	2018-11-01	0.5462	43463.55	23739.79	19723.76	
	201	2	4	2017-12-01	0.5462	43703.76	23870.99	19832.77	
	202	2	4	2018-12-01	0.5462	43703.76	23870.99	19832.77	
	203	2	4	2017-01-01	0.5462	536042.64	292786.49	243256.15	
	204	2	4	2018-01-01	0.5462	536042.64	292786.49	243256.15	
	205 rd	nwe x 7 a	columi	ne					

205 rows × 7 columns

Next steps: ( Generate code with q

View recommended plots

New interactive sheet

q.Date.dt.month\_name()

<b>→</b>		Date
	0	January
	1	February
	2	March
	3	April
	4	April
	200	November
	201	December
	202	December
	203	January
	204	January
	205 rc	ws × 1 columns
	dtype	: object

It's time to add "MONTH" to our Table for "Monthly Revenue Analysis" using the above format

q["Month"] = q.Date.dt.month\_name()

Let's effect the changes in our original Table

q

<b>→</b>		Store	Cat	Date	Margin	Sales	Profit	COGs	Month	
	0	1	1	2018-01-01	0.5432	24924.50	13538.99	11385.51	January	ılı
	1	1	1	2018-02-01	0.5432	46039.49	25008.65	21030.84	February	+/
	2	1	1	2018-03-01	0.5432	41595.55	22594.70	19000.85	March	_
	3	1	1	2017-04-01	0.5432	19403.54	10540.00	8863.54	April	
	4	1	1	2018-04-01	0.5432	19403.54	10540.00	8863.54	April	
	200	2	4	2018-11-01	0.5462	43463.55	23739.79	19723.76	November	
	201	2	4	2017-12-01	0.5462	43703.76	23870.99	19832.77	December	
	202	2	4	2018-12-01	0.5462	43703.76	23870.99	19832.77	December	
	203	2	4	2017-01-01	0.5462	536042.64	292786.49	243256.15	January	
	204	2	4	2018-01-01	0.5462	536042.64	292786.49	243256.15	January	
	205 rd	ows × 8 d	colum	ns						

Next steps: ( Generate code with q

View recommended plots

New interactive sheet

Call the Table\_Out

q.Date.dt.year

_		
<b>→</b>		Date
	0	2018
	1	2018
	2	2018
	3	2017
	4	2018
	200	2018
	201	2017
	202	2018
	203	2017
	204	2018
	205 rc	ws × 1 columns
	dtype	: int32

Now is time to also add "YEAR" in our original Table for "Yearly Revenue Analysis" using the above format

q["Year"] = q.Date.dt.year

Let's effect this in our original Table using the format above by creating a column(Year) in our original Table.

q

<b>→</b>		Store	Cat	Date	Margin	Sales	Profit	COGs	Month	Year	
	0	1	1	2018-01-01	0.5432	24924.50	13538.99	11385.51	January	2018	11.
	1	1	1	2018-02-01	0.5432	46039.49	25008.65	21030.84	February	2018	+/
	2	1	1	2018-03-01	0.5432	41595.55	22594.70	19000.85	March	2018	
	3	1	1	2017-04-01	0.5432	19403.54	10540.00	8863.54	April	2017	
	4	1	1	2018-04-01	0.5432	19403.54	10540.00	8863.54	April	2018	
	200	2	4	2018-11-01	0.5462	43463.55	23739.79	19723.76	November	2018	
	201	2	4	2017-12-01	0.5462	43703.76	23870.99	19832.77	December	2017	
	202	2	4	2018-12-01	0.5462	43703.76	23870.99	19832.77	December	2018	
	203	2	4	2017-01-01	0.5462	536042.64	292786.49	243256.15	January	2017	
	204	2	4	2018-01-01	0.5462	536042.64	292786.49	243256.15	January	2018	
	205 rd	ws × 9 c	olumr	าร							

Next steps: ( Generate code with q

View recommended plots

New interactive sheet

Call out the Table

q.groupby("Store")["Profit"].sum()

 $\overline{\mathbf{T}}$ 

Profit

Store

Now is time to "Buth a Groupby Function" to check or find the "Total\_Profit" the Business made by stores. To know which store perform well 4342447.0

q.pivot\_table(index = ["Store", "Cat"], values = "Profit", aggfunc = "sum")



- Profit 🚃
- ore Cat

  1 1 714898.98
  - **2** 1216161.08
  - **3** 267867.50
  - 4 935243.44
- **1** 1062919.02
  - **2** 1771931.26
  - 3 336450.76
  - **4** 1171145.96

Now is time to also use "PIVOT\_TABLE" in Python to check for thesame thing. Profit by Store and Category that made the highest Profit.

q.pivot\_table(index = ["Store","Cat"],values =["Profit","Sales"],aggfunc ="sum")



		Profit	Sales	
Store	Cat			ılı
1	1	714898.98	1316087.94	
	2	1216161.08	2194444.36	
	3	267867.50	513943.80	
	4	935243.44	1712272.88	
2	1	1062919.02	1956772.84	
	2	1771931.26	3197277.76	