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

In [7]:
```
data = pd.read_csv("/content/Ecommerce Purchases.csv")
```

#display top 4 items

In [10]:
```
data.head(4)
```

Out[10]:

	Address	Lot	AM or PM	Browser Info	Company	Crec
0	16629 Pace Camp Apt. 448\nAlexisborough, NE 77...	46 in	PM	Opera/9.56. (X11; Linux x86_64; sl-SI) Presto/2...	Martinez-Herman	6011929061
1	9374 Jasmine Spurs Suite 508\nSouth John, TN 8...	28 rn	PM	Opera/8.93. (Windows 98; Win 9x 4.90; en-US) Pr...	Fletcher, Richards and Whitaker	3337758169
2	Unit 0065 Box 5052\nDPO AP 27450	94 vE	PM	Mozilla/5.0 (compatible; MSIE 9.0; Windows NT ...	Simpson, Williams and Pham	6759576661
3	7780 Julia Fords\nNew Stacy, WA 45798	36 vm	PM	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_8_0 ...	Williams, Marshall and Buchanan	6011578504

#last 4 data

In [11]:
```
data.tail(4)
```

Out[11]:

	Address	Lot	AM or PM	Browser Info	Company	
9996	832 Curtis Dam Suite 785\nNorth Edwardburgh, T...	41 JY	AM	Mozilla/5.0 (compatible; MSIE 9.0; Windows NT ...	Hale, Collins and Wilson	21003
9997	Unit 4434 Box 6343\nDPO AE 28026-0283	74 Zh	AM	Mozilla/5.0 (Macintosh; U; Intel Mac OS X 10_7...	Anderson Ltd	60115
9998	0096 English Rest\nRoystad, IA 12457	74 cL	PM	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_8_8;...	Cook Inc	18000
9999	40674 Barrett Stravenue\nGrimesville, WI 79682	64 Hr	AM	Mozilla/5.0 (X11; Linux i686; rv:1.9.5.20) Gec...	Greene Inc	41399

#checking the datatypes of each column

In [20]:
```
data.dtypes
```

Out[20]:

	0
Address	object
Lot	object
AM or PM	object
Browser Info	object
Company	object
Credit Card	int64
CC Exp Date	object
CC Security Code	int64
CC Provider	object
Email	object
Job	object
IP Address	object
Language	object
Purchase Price	float64

dtype: object

#check null values in datasets

In [25]:
```
data.isnull().sum(axis = 0)
```

Out[25]:

	0
Address	0
Lot	0
AM or PM	0
Browser Info	0
Company	0
Credit Card	0
CC Exp Date	0
CC Security Code	0
CC Provider	0
Email	0
Job	0
IP Address	0
Language	0
Purchase Price	0

dtype: int64

#number of rows and column in data

In [28]:
```
data.shape
```

Out[28]: (10000, 14)

In [37]:
```
print("rows : ", data.shape[0])
print("columns : ", data.shape[1])
```

```
rows :  10000
columns :  14
```

In [32]:
```
len(data.columns)
```

Out[32]: 14

In [33]:
```
len(data.index)
```

Out[33]:
```
10000
```

In [36]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 14 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Address           10000 non-null  object 
 1   Lot               10000 non-null  object 
 2   AM or PM          10000 non-null  object 
 3   Browser Info      10000 non-null  object 
 4   Company           10000 non-null  object 
 5   Credit Card       10000 non-null  int64  
 6   CC Exp Date       10000 non-null  object 
 7   CC Security Code  10000 non-null  int64  
 8   CC Provider       10000 non-null  object 
 9   Email             10000 non-null  object 
 10  Job               10000 non-null  object 
 11  IP Address        10000 non-null  object 
 12  Language          10000 non-null  object 
 13  Purchase Price    10000 non-null  float64
dtypes: float64(1), int64(2), object(11)
memory usage: 1.1+ MB
```

#highest and lowest purchase price

In [40]:
```
data["Purchase Price"].max()
```

Out[40]:
```
99.99
```

In [41]:
```
data["Purchase Price"].min()
```

Out[41]:
```
0.0
```

#average purchase price

In [42]:
```
data["Purchase Price"].mean()
```

Out[42]:
```
np.float64(50.347302)
```

#people having french "fr" as their language

In [43]:
```
data.columns
```

Out[43]:
```
Index(['Address', 'Lot', 'AM or PM', 'Browser Info', 'Company',
'Credit Card',
       'CC Exp Date', 'CC Security Code', 'CC Provider', 'Email',
'Job',
       'IP Address', 'Language', 'Purchase Price'],
      dtype='object')
```

In [44]:
```
data["Language"]
```

Out[44]:

	Language
0	el
1	fr
2	de
3	es
4	es
...	...
9995	it
9996	pt
9997	el
9998	es
9999	el

10000 rows × 1 columns

dtype: object

In [47]:
```
len(data[data["Language"] == "fr"])
```

Out[47]:
```
1097
```

In [53]:
```
data[data["Language"] == "fr"].count()
```

Out[53]:

	0
Address	1097
Lot	1097
AM or PM	1097
Browser Info	1097
Company	1097
Credit Card	1097
CC Exp Date	1097
CC Security Code	1097
CC Provider	1097
Email	1097
Job	1097
IP Address	1097
Language	1097
Purchase Price	1097

dtype: int64

job title containing engineer

In [54]:
```
data.columns
```

Out[54]:
```
Index(['Address', 'Lot', 'AM or PM', 'Browser Info', 'Company',
'Credit Card',
       'CC Exp Date', 'CC Security Code', 'CC Provider', 'Email',
'Job',
       'IP Address', 'Language', 'Purchase Price'],
      dtype='object')
```


In [57]:
```
data["Job"]
```

Out[57]:

	Job
0	Scientist, product/process development
1	Drilling engineer
2	Customer service manager
3	Drilling engineer
4	Fine artist
...	...
9995	Printmaker
9996	Energy engineer
9997	Veterinary surgeon
9998	Local government officer
9999	Embryologist, clinical

10000 rows × 1 columns

dtype: object

In [63]:
```
data[data["Job"].str.contains("engineer", case = False)].head()
```

Out[63]:

	Address	Lot	AM or PM	Browser Info	Company	Cr
1	9374 Jasmine Spurs Suite 508\nSouth John, TN 8...	28 rn	PM	Opera/8.93. (Windows 98; Win 9x 4.90; en-US) Pr...	Fletcher, Richards and Whitaker	33377581
3	7780 Julia Fords\nNew Stacy, WA 45798	36 vm	PM	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_8_0 ...	Williams, Marshall and Buchanan	60115785
50	41159 Michael Centers\nAdamsfort, RI 37108-6674	46 Ce	PM	Mozilla/5.0 (Windows 98; Win 9x 4.90; sl-SI; r...	Wright, Williams and Mendez	40085864
55	27635 Maureen Bypass Apt. 883\nSandraview, SD ...	59 LJ	AM	Mozilla/5.0 (iPod; U; CPU iPhone OS 3_3 like M...	Sims-Lyons	31581136
60	7126 Katherine Squares\nPerkinsview, CO 97299-...	63 qu	AM	Opera/8.68. (X11; Linux x86_64; en-US) Presto/2...	Marshall-Fernandez	34976774

In [62]:
```
len(data[data["Job"].str.contains("engineer",case=False)])
```

Out[62]: 984

find email of the person with following ip adresses : 132.207.160.22

In [64]:
```
data.columns
```

Out[64]:
```
Index(['Address', 'Lot', 'AM or PM', 'Browser Info', 'Company',
'Credit Card',
       'CC Exp Date', 'CC Security Code', 'CC Provider', 'Email',
'Job',
       'IP Address', 'Language', 'Purchase Price'],
      dtype='object')
```

In [72]:
```
data["IP Address"]
```

Out[72]:

	IP Address
0	149.146.147.205
1	15.160.41.51
2	132.207.160.22
3	30.250.74.19
4	24.140.33.94
...	...
9995	29.73.197.114
9996	121.133.168.51
9997	156.210.0.254
9998	55.78.26.143
9999	176.119.198.199

10000 rows × 1 columns

dtype: object

In [74]:
```
data[data["IP Address"] == "132.207.160.22"]["Email"]
```

Out[74]:

	Email
2	amymiller@morales-harrison.com

dtype: object

#How many people have Mastercard as their credit card provider and made a purchase above 50

In [75]:
```
data.columns
```

Out[75]:
```
Index(['Address', 'Lot', 'AM or PM', 'Browser Info', 'Company',
'Credit Card',
       'CC Exp Date', 'CC Security Code', 'CC Provider', 'Email',
'Job',
       'IP Address', 'Language', 'Purchase Price'],
      dtype='object')
```

In [80]:
```
len(data[(data["CC Provider"] == "Mastercard") & (data["Purchase Price"] >
50)])
```

Out[80]: 405

In [81]:
```
data[(data["CC Provider"] == "Mastercard") & (data["Purchase Price"] >
50)].count()
```

Out[81]:

	0
Address	405
Lot	405
AM or PM	405
Browser Info	405
Company	405
Credit Card	405
CC Exp Date	405
CC Security Code	405
CC Provider	405
Email	405
Job	405
IP Address	405
Language	405
Purchase Price	405

dtype: int64

#email of the person with following credit card number: 4664825258997302

In [82]:
```
data.columns
```

Out[82]:
```
Index(['Address', 'Lot', 'AM or PM', 'Browser Info', 'Company',
'Credit Card',
       'CC Exp Date', 'CC Security Code', 'CC Provider', 'Email',
'Job',
       'IP Address', 'Language', 'Purchase Price'],
      dtype='object')
```

In [83]:
```
data["Credit Card"]
```

Out[83]:

	Credit Card
0	6011929061123406
1	3337758169645356
2	675957666125
3	6011578504430710
4	6011456623207998
...	...
9995	342945015358701
9996	210033169205009
9997	6011539787356311
9998	180003348082930
9999	4139972901927273

10000 rows × 1 columns

dtype: int64

In [86]:
```
data[data["Credit Card"] == 4664825258997302]["Email"]
```

Out[86]:

	Email
9992	bberry@wright.net

dtype: object

#how many people purchase during the am and how many purchase during the pm

In [87]:
```
data.columns
```

Out[87]:
```
Index(['Address', 'Lot', 'AM or PM', 'Browser Info', 'Company',
'Credit Card',
       'CC Exp Date', 'CC Security Code', 'CC Provider', 'Email',
'Job',
       'IP Address', 'Language', 'Purchase Price'],
      dtype='object')
```

In [88]:
```
data["AM or PM"]
```

Out[88]:

	AM or PM
0	PM
1	PM
2	PM
3	PM
4	AM
...	...
9995	PM
9996	AM
9997	AM
9998	PM
9999	AM

10000 rows × 1 columns

dtype: object

In [89]:
```
data[data["AM or PM"] == "AM"].count()
```

Out[89]:

	0
Address	4932
Lot	4932
AM or PM	4932
Browser Info	4932
Company	4932
Credit Card	4932
CC Exp Date	4932
CC Security Code	4932
CC Provider	4932
Email	4932
Job	4932
IP Address	4932
Language	4932
Purchase Price	4932

dtype: int64

In [90]:
```
data[data["AM or PM"] == "PM"].count()
```

Out[90]:

	0
Address	5068
Lot	5068
AM or PM	5068
Browser Info	5068
Company	5068
Credit Card	5068
CC Exp Date	5068
CC Security Code	5068
CC Provider	5068
Email	5068
Job	5068
IP Address	5068
Language	5068
Purchase Price	5068

dtype: int64

#how many people have credit card that expires in 2020

In [92]:
```
data.columns
```

Out[92]:
```
Index(['Address', 'Lot', 'AM or PM', 'Browser Info', 'Company',
'Credit Card',
       'CC Exp Date', 'CC Security Code', 'CC Provider', 'Email',
'Job',
       'IP Address', 'Language', 'Purchase Price'],
      dtype='object')
```


In [93]:
```
data["CC Exp Date"]
```

Out[93]:

	CC Exp Date
0	02/20
1	11/18
2	08/19
3	02/24
4	10/25
...	...
9995	03/22
9996	07/25
9997	05/21
9998	11/17
9999	02/19

10000 rows × 1 columns

dtype: object

In [106]:
```
data[data["CC Exp Date"].apply(lambda x : x[3:] == "20")].count()
```

Out[106]:

	0
Address	988
Lot	988
AM or PM	988
Browser Info	988
Company	988
Credit Card	988
CC Exp Date	988
CC Security Code	988
CC Provider	988
Email	988
Job	988
IP Address	988
Language	988
Purchase Price	988

dtype: int64

#top 5 most popular email providers (eg gmail.com, yahoo.com, etc)

In [107]:
```
data.columns
```

Out[107]:
```
Index(['Address', 'Lot', 'AM or PM', 'Browser Info', 'Company',
'Credit Card',
       'CC Exp Date', 'CC Security Code', 'CC Provider', 'Email',
'Job',
       'IP Address', 'Language', 'Purchase Price'],
      dtype='object')
```

In [109]:
```
list1 = []
for email in data["Email"]:
  list1.append([email.split("@")[1]])
```


In [115]:
```
data["temp"] = list1
data["Email"].head(5)
data["temp"].value_counts().head(5)
```

Out[115]:

temp	count
[hotmail.com]	1638
[yahoo.com]	1616
[gmail.com]	1605
[smith.com]	42
[williams.com]	37

dtype: int64

In [ ]:
```

```