In [6]: Out[6]:	<pre>import pandas as pd</pre>
	data = pd.read_csv('Ecommerce Purchases') Address Lot Address Lot PM Opera/9.56.(X11; Linux x86_64; sl-Sl) Presto/2 Martinez-Herman 6011929061123406 02/20 900 JCB 16 digit pdunlap@yahoo.com Scientist, product/process development 149.146.147.205 el 98.14
	1 9374 Jasmine Spurs Suite 508\nSouth John, TN 8 28 rm PM Opera/8.93.(Windows 98; Win 9x 4.90; en-US) Pr Fletcher, Richards and Whitaker 3337758169645356 11/18 561 Mastercard anthony41@reed.com Drilling engineer 15.160.41.51 fr 70.73 2 Unit 0065 Box 5052\nDPO AP 27450 94 vE PM Mozilla/5.0 (compatible; MSIE 9.0; Windows NT Simpson, Williams and Pham Pham 675957666125 08/19 699 JCB 16 digit amymiller@morales-harrison.com Customer service manager 132.207.160.22 de 0.95 3 7780 Julia Fords\nNew Stacy, WA 45798 36 vm PM Mozilla/5.0 (Macintosh; Intel Mac OS X 10_8 m.) Williams, Marshall and Buchanan Buchanan 6011578504430710 02/24 384 Discover brent16@olson-robinson.info Drilling engineer 30.250.74.19 es 78.04
	4 23012 Munoz Drive Suite 337\nNew Cynthia, TX 5 20 IE AM Opera/9.58.(X11; Linux x86_64; it-IT) Presto/2 Brown, Watson and Andrews 6011456623207998 10/25 678 Diners Club / Carte Blanche christopherwright@gmail.com Fine artist 24.140.33.94 es 77.82 <
	9997 Unit 4434 Box 6343\nDPO AE 28026-0283
In [8]:	10000 rows × 14 columns data.head(10)
Out[8]:	The second of th
	2 Unit 0065 Box 5052\nDPO AP 27450
	5 7502 Powell Mission Apt. 768\nTravisland, VA 3 21 XT PM Mozilla/5.0 (Macintosh; U; PPC Mac OS X 10_8_5 Silva-Anderson 30246185196287 07/25 7169 Discover ynguyen@gmail.com Fish farm manager 55.96.152.147 ru 25.15 6 93971 Conway Causeway\nAndersonburgh, AZ 75107 96 Xt AM Mozilla/5.0 (compatible; MSIE 7.0; Windows NT Gibson and Sons 6011398782655569 07/24 714 VISA 16 digit olivia04@yahoo.com Dancer 127.252.144.18 de 88.56 7 260 Rachel Plains Suite 366\nCastroberg, WV 24 96 pG PM Mozilla/5.0 (X11; Linux i686) AppleWebKit/5350 Marshall-Collins 561252141909 06/25 256 VISA 13 digit phillip48@parks.info Event organiser 224.247.97.150 pt 44.25
In [9]:	8 2129 Dylan Burg\nNew Michelle, ME 28650 \$\frac{45}{JN}\$ PM \$\frac{Mozilla/5.0 \text{ (Macintosh; U; Intel Mac OS X}}{10_7}\$ Galloway and Sons \$180041795790001\$ 04/24 899 JCB 16 digit kdavis@rasmussen.com Financial manager \$146.234.201.229\$ ru 59.54 9 \$\frac{3795 \text{ Dawson Extensions\nLake Tinafort, ID}}{88739}\$ \frac{15}{Ug}\$ AM \$\frac{Mozilla/5.0 \text{ (X11; Linux i686; rv:1.9.7.20)}}{Gec}\$ \frac{Rivera, Buchanan and}{Ramirez}\$ \$4396283918371\$ 01/17\$ 931 American Express \$qcoleman@hunt-huerta.com\$ Forensic scientist \$236.198.199.8\$ zh 95.63 \$\frac{data.tail(10)}{}\$ \$\frac{1}{3} \text{ data} \text{ tail}(10)\$ \$\frac{1}{3} mozing in the mozing of the mozing in the mozing i
Out[9]:	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 9990 75731 Molly Springs\nWest Danielle, VT 96934-5102 93 ty PM Mozilla/5.0 (Macintosh; Intel Mac OS X 10_7_4; Pace, Vazquez and Richards Richards Richards Richards Richards Snyder Inc 4221582137197481 02/24 969 Voyager kking@wise-liu.com IT sales professional 254.25.31.156 el 25.93
	9992 885 Allen Mountains Apt. 230\nWallhaven, LA 16995 VH PM Mozilla/5.0 (Macintosh; PPC Mac OS X 10_6_5) A Wells Ltd 466482528997302 10/20 431 Discover bberry@wright.net Set designer 174.173.51.32 de 67.96 9993 7555 Larson Locks Suite 229\nEllisburgh, MA 34 72 jg PM Mozilla/5.0 (Macintosh; U; Intel Mac OS X 10_8 Colon and Sons 30025560104631 10/25 629 Maestro chelseawilliams@lopez.biz exhibition/display 177.46.82.128 el 65.61 9994 6276 Rojas Hollow\nLake Louis, WY 56410-7837 Ex PM Opera/9.68.(X11; Linux x86_64; sl-Sl) Presto/2 Ritter-Smith 3112186784121077 01/25 1823 Maestro iroberts@gmail.com Education officer, museum 242.44.112.18 zh 31.85
	999 999 999 999 999 999 999 999 999 99
In [10]:	9998 0096 English Rest\nRoystad, IA 12457 74 PM Mozilla/5.0 (Macintosh; Intel Mac OS X 10_8_8; Cook Inc 180003348082930 11/17 987 American Express elizabethmoore@reid.net Local government officer 55.78.26.143 es 38.84 9999 40674 Barrett Stravenue\nGrimesville, WI 79682 Hr AM Mozilla/5.0 (X11; Linux i686; rv:1.9.5.20) Gec Greene Inc 4139972901927273 02/19 302 JCB 15 digit rachelford@vaughn.com Embryologist, clinical 176.119.198.199 el 67.59 data.dtypes
Out[10]:	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
In [12]: Out[12]:	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
In [14]:	Purchase Price 0 dtype: int64 1. How many rows and columns are there in our Dataset? len(data.columns)
Out[14]:	len(data)
Out[17]: In [19]:	<pre>data.info() <class 'pandas.core.frame.dataframe'=""> RangeIndex: 10000 entries, 0 to 9999 Data columns (total 14 columns): # Column Non-Null Count Dtype</class></pre>
	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)
In [20]: Out[20]:	memory usage: 1.1+ MB 1. Highest and Lowest Purchase Prices. data.columns
In [22]:	<pre>Index(['Address', 'Lot', 'AM or PM', 'Browser Info', 'Company', 'Credit Card',</pre>
Out[22]: In [23]: Out[23]:	<pre>data['Purchase Price'].min() 0.0</pre>
In [24]: Out[24]:	1. Average Purchase Price data['Purchase Price'].mean() 50.34730200000025
In [26]: Out[26]:	uata.uesci ibe()
	mean 2.341374e+15 907.217800 50.347302 std 2.256103e+15 1589.693035 29.015836 min 6.040186e+10 0.000000 0.000000 25% 3.056322e+13 280.000000 25.150000 50% 8.699942e+14 548.00000 50.505000 75% 4.492298e+15 816.000000 75.770000
In [27]:	max 6.012000e+15 9993.00000 99.990000 1. How many people have French 'fr' as their Language?
	<pre>Index(['Address', 'Lot', 'AM or PM', 'Browser Info', 'Company', 'Credit Card',</pre>
Out[35]: In [37]:	<pre>data[data['Language'] == 'fr'].count()</pre>
Out[37]:	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
In [38]: Out[38]:	Index(['Address', 'Lot', 'AM or PM', 'Browser Info', 'Company', 'Credit Card',
In [54]: Out[54]:	<pre>dtype='object') len(data[data['Job'].str.contains('engineer', case =False)]) 984</pre>
In [55]: Out[55]:	Index(['Address', 'Lot', 'AM or PM', 'Browser Info', 'Company', 'Credit Card',
In [62]: Out[62]:	<pre>dtype='object') data[data['IP Address'] == '132.207.160.22']['Email']</pre>
In [63]: Out[63]:	Index(['Address', 'Lot', 'AM or PM', 'Browser Info', 'Company', 'Credit Card',
In [71]: Out[71]:	'IP Address', 'Language', 'Purchase Price'], dtype='object') len(data[(data['CC Provider'] == 'Mastercard') & (data['Purchase Price'] > 50)]) 405
In [74]: Out[74]:	Address
	Company 405 Credit Card 405 CC Exp Date 405 CC Security Code 405 CC Provider 405 Email 405 Job 405 IP Address 405
In [75]:	Purchase Price 405 dtype: int64 1. Find the email of the person with the following Credit Card Number: 4664825258997302
Out[75]: In [80]:	<pre>Index(['Address', 'Lot', 'AM or PM', 'Browser Info', 'Company', 'Credit Card',</pre>
	Name: Email, dtype: object 1. How many people purchase during the AM and how many people purchase during PM?
Out[80]: In [81]:	Index(['Address', 'Lot', 'AM or PM', 'Browser Info', 'Company', 'Credit Card',
In [81]:	'IP Address', 'Language', 'Purchase Price'], dtype='object') len(data[data['AM or PM'] == 'AM'])
In [81]: Out[81]:	<pre>dtype='object') len(data['AM or PM'] == 'AM']) 4932 len(data[data['AM or PM'] == 'PM'])</pre>
In [81]: Out[81]: In [89]: Out[89]: In [90]:	<pre>dtype='object') len(data[data['AM or PM'] == 'AM']) 4932 len(data[data['AM or PM'] == 'PM']) 5068 data['AM or PM'].value_counts()</pre>
<pre>In [81]: Out[81]: In [89]: Out[89]: In [90]: Out[90]: In [92]: Out[92]:</pre>	len(data[data['AM or PM'] == 'AM']) 4932 len(data[data['AM or PM'] == 'PM']) 5068 data['AM or PM'].value_counts() PM
<pre>In [81]: Out[81]: In [89]: Out[89]: In [90]: Out[90]: In [92]: In [92]:</pre>	Inc(data data 'AM or PN' == 'AN') Inc(data data 'AM or PN' == 'PN') Inc(data data 'AM or PN' == 'PN') See
In [81]: Out[81]: In [89]: Out[89]: In [90]: Out[90]: In [92]: Out[92]: In [93]: In [93]:	Including [And or PM'] == 'AM']) Including [And or PM'] == 'AM']) Including [And or PM'] == 'PM']) Including [And or PM'] == 'AM']) Including [And
In [81]: Out[81]: In [89]: Out[89]: In [90]: Out[90]: In [92]: Out[92]: In [93]: In [93]:	Int(Sara[Asan] (An or PR] = (AN])
In [81]: Out[81]: In [89]: Out[89]: In [90]: Out[92]: Out[92]: In [93]: Out[93]: In [94]: In [96]:	August A
In [81]: Out[81]: In [89]: Out[89]: In [90]: Out[90]: In [92]: Out[92]: In [93]: Out[93]: In [94]: Out[94]: Out[94]:	Bigger day, earl
In [81]: Out[81]: In [89]: Out[89]: In [90]: In [92]: Out[92]: In [93]: In [94]: Out[94]: In [98]: Out[98]: Out[102 Out[102	######################################
In [81]: Out[81]: Out[89]: Out[90]: Out[90]: In [92]: Out[92]: In [93]: Out[94]: In [96]: In [97]: In [98]: In [98]:	######################################
In [81]: Out [81]: Out [89]: Out [90]: In [92]: Out [92]: In [93]: Out [93]: In [94]: Out [94]: Out [98]: In [102 Out [102 In [103	Reference Part Pa
In [81]: Out [81]: Out [89]: Out [90]: In [92]: Out [92]: In [93]: Out [93]: In [94]: Out [94]: Out [98]: In [102 In [102 In [103 In [105	Part
In [81]: Out [81]: In [89]: Out [89]: Out [90]: In [92]: Out [92]: In [93]: Out [93]: In [94]: Out [94]: In [102 Out [102 In [103 In [107 In [107 In [107	Part
In [81]: Out [81]: In [89]: Out [89]: Out [90]: In [92]: Out [92]: In [93]: Out [93]: In [94]: In [96]: In [102 Out [105 Out [105 Out [107 Out [107	Margin Part
In [81]: Out [81]: Out [89]: Out [89]: Out [90]: In [92]: Out [92]: In [93]: Out [94]: In [96]: In [102 Out [107 Out [107 Out [107 Out [107 Out [107 Out [109 Out [109 Out [109	Margin Paris Par
In [81]: Out [81]: Out [89]: Out [90]: In [90]: Out [92]: Out [92]: In [93]: Out [93]: In [94]: Out [94]: Out [102 Out [105 Out [107 Out [107 Out [107 Out [109 In [109 Out [109 Out [109 Out [109 Out [109	Margin Part
In [81]: Out [81]: Out [89]: Out [90]: In [90]: Out [92]: Out [92]: In [93]: Out [93]: In [94]: Out [94]: Out [102 Out [105 Out [107 Out [107 Out [109 Out [109 Out [109 Out [109 Out [109 Out [109	Margin Part
In [81]: Out [81]: Out [89]: Out [90]: In [90]: Out [92]: Out [92]: In [93]: Out [93]: In [94]: Out [94]: Out [102 Out [105 Out [107 Out [107 Out [109 Out [109 Out [109 Out [109 Out [109	Margin Part