In [61]:
```
%%html
<h1>Course1Task1!</h1>
```

Course1Task1!

In [62]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
```

In [63]:
```
#data is the variable name for the Pandas default data structure called
# a "Dataframe" in this case its a 2D Array
data = pd.read_csv('Demographic_Data.csv')
```

In [64]:
```
#by default, the python function head() returns the first 5 lines of a
# dataframe
data.head()
```

Out[64]:

	in-store	age	items	amount	region
0	0	37	4	281.03	2
1	0	35	2	219.51	2
2	1	45	3	1525.70	4
3	1	46	3	715.25	3
4	1	33	4	1937.50	1

In [65]:
```
#describe() is used to view some basic statistical details
#like percentile, mean, std etc.
#of a data frame or a series of numeric values.
data.describe()
```

Out[65]:

	in-store	age	items	amount	region
count	80000.000000	80000.000000	80000.000000	80000.000000	80000.000000
mean	0.500000	45.757925	4.504975	835.919670	2.675000
std	0.500003	15.715679	2.061238	721.273736	1.126672
min	0.000000	18.000000	1.000000	5.004700	1.000000
25%	0.000000	33.000000	3.000000	285.140000	2.000000
50%	0.500000	45.000000	4.000000	582.315000	3.000000
75%	1.000000	56.000000	6.000000	1233.700000	4.000000
max	1.000000	85.000000	8.000000	3000.000000	4.000000

In [66]:
```
#info() function is used to get a concise summary of the dataframe.
#It comes really handy when doing exploratory analysis of the data.
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 80000 entries, 0 to 79999
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   in-store  80000 non-null  int64
 1   age       80000 non-null  int64
 2   items     80000 non-null  int64
 3   amount    80000 non-null  float64
 4   region    80000 non-null  int64
dtypes: float64(1), int64(4)
memory usage: 3.1 MB
```

In [67]:
```
#Checking for duplicate rows - these can cause real issues
#when examining data
data = data.drop_duplicates()
```

In [68]:
```
#Checking for missing values (let's print the sum):
print(data.isnull().sum())
```

```
in-store    0
age         0
items       0
amount      0
region      0
dtype: int64
```

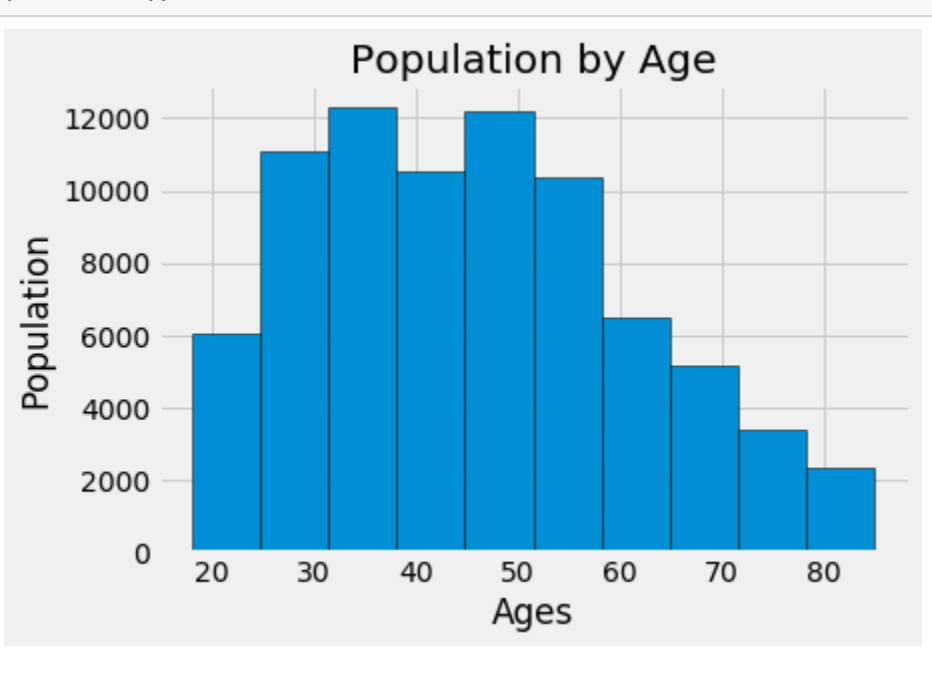
In [69]:
```
#Check that all of our datatypes are numeric:
data.dtypes
```

Out[69]:
```
in-store      int64
age           int64
items         int64
amount      float64
region        int64
dtype: object
```

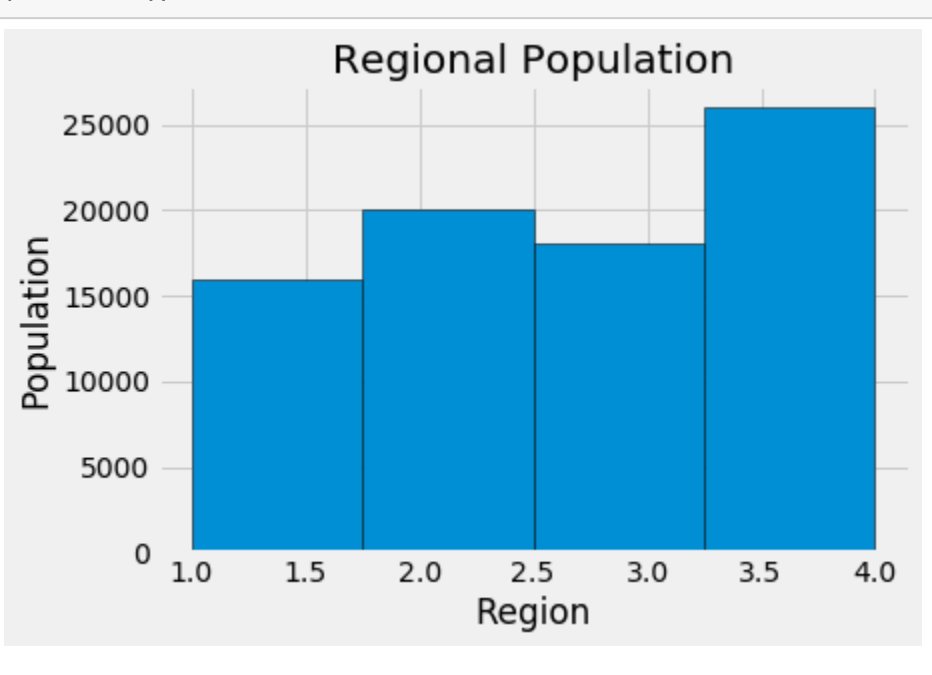
In [70]:
```
header = data.dtypes.index
print(header)
```

```
Index(['in-store', 'age', 'items', 'amount', 'region'], dtype='object')
```

In [71]:
```
plt.hist(data['age'], edgecolor='black')
plt.title('Population by Age')
plt.xlabel('Ages')
plt.ylabel('Population')
plt.show()
```

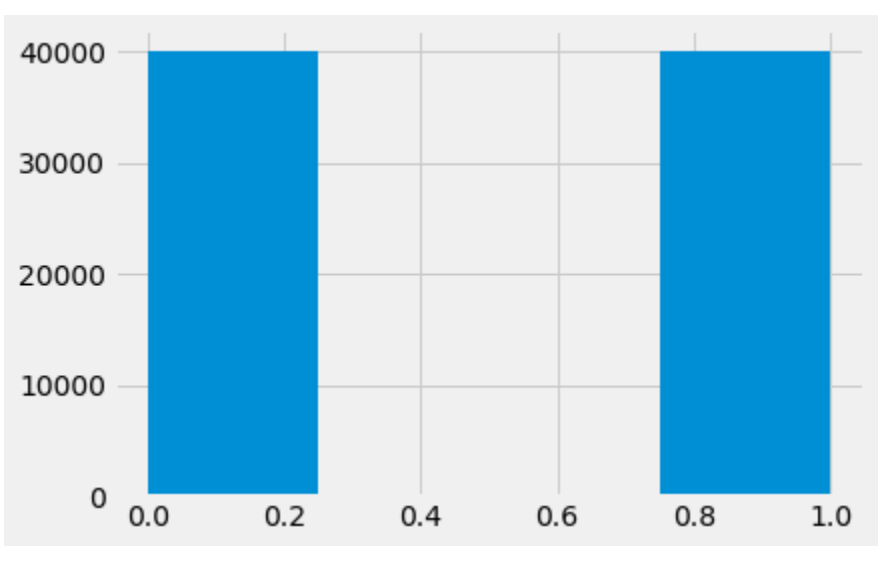


In [72]:
```
plt.hist(data['region'], bins=4, edgecolor='black')
plt.title('Regional Population')
plt.xlabel('Region')
plt.ylabel('Population')

plt.show()
```



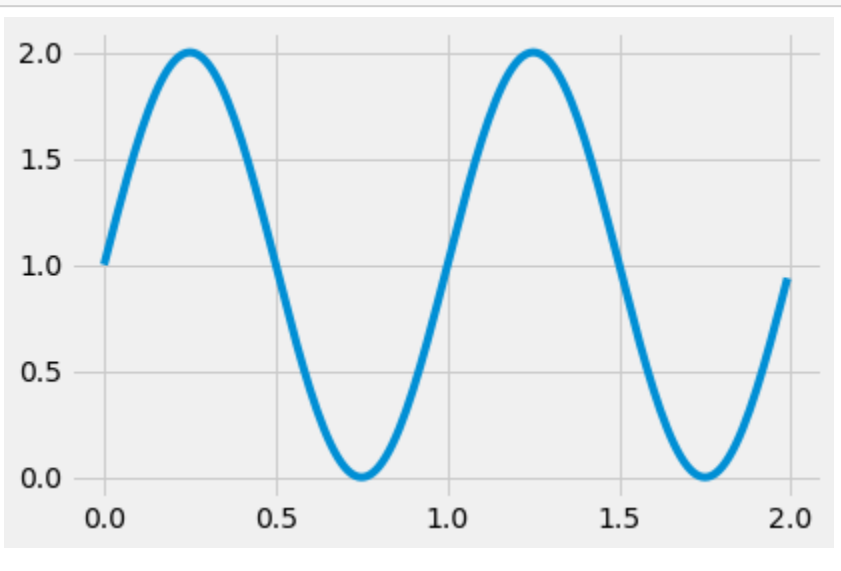
In [73]:
```
plt.hist(data['in-store'], bins=4)
```

Out[73]:
```
(array([39989.,     0.,     0., 39990.]),
 array([0.  , 0.25, 0.5 , 0.75, 1.  ]),
 <a list of 4 Patch objects>)
```



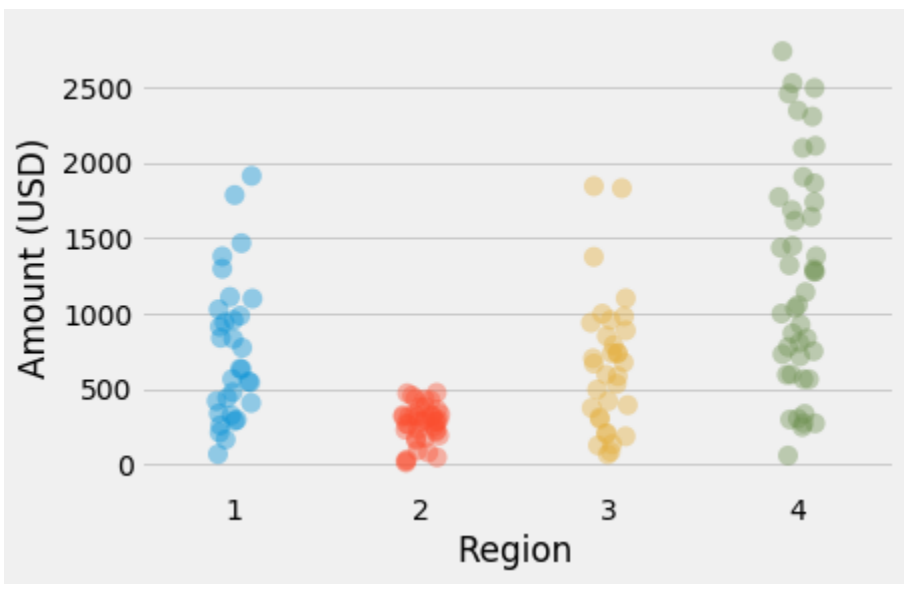
In [74]:
```
import numpy as np
```

In [75]:
```
# Data for plotting
t = np.arange(0.0, 2.0, 0.01)
s = 1 + np.sin(2 * np.pi * t)
fig, ax = plt.subplots()
ax.plot(t, s)
plt.show()
```



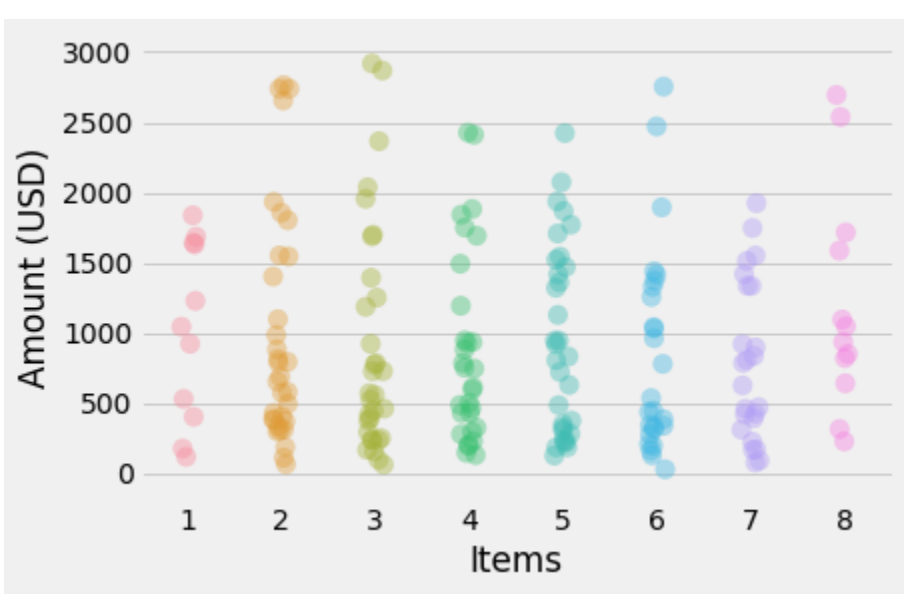
In [76]:
```
data_sample = data.sample(150)
plt.style.use('fivethirtyeight')
import seaborn as sns
x = data_sample['region']
y = data_sample['amount']

sns.stripplot(x,y,s=10,alpha=0.4,jitter=True,edgecolor='none')
sns.despine()

plt.xlabel('Region')
plt.ylabel('Amount (USD)')

#plt.scatter(x,y,s=50,edgecolor='black',linewidth=1, marker='o', alpha=0.5)
#plt.plot(x,y)
#plt.show()
```

Out[76]: Text(0, 0.5, 'Amount (USD)')



In [77]:
```
data_sample = data.sample(200)
plt.style.use('fivethirtyeight')
import seaborn as sns
x = data_sample['items']
y = data_sample['amount']

sns.stripplot(x,y,s=10,alpha=0.4,jitter=True,edgecolor='none')
sns.despine()

plt.xlabel('Items')
plt.ylabel('Amount (USD)')
```

Out[77]: Text(0, 0.5, 'Amount (USD)')



In [78]:
```
region1 = (data['region'] == 1)
region2 = (data['region'] == 2)
region3 = (data['region'] == 3)
region4 = (data['region'] == 4)

print('region 1 customers: ', region1.sum())
print('region 2 customers: ', region2.sum())
print('region 3 customers: ', region3.sum())
print('region 4 customers: ', region4.sum())
print('total customers: ', region1.sum() + region2.sum() + region3.sum() + region4.sum())
```

```
region 1 customers:  15997
region 2 customers:  19994
region 3 customers:  18000
region 4 customers:  25988
total customers:  79979
```

In [79]:
```
reg_1_age = data.loc[region1,['age']]
reg_2_age = data.loc[region2,['age']]
reg_3_age = data.loc[region3,['age']]
reg_4_age = data.loc[region4,['age']]

reg_1_items = data.loc[region1,['items']]
reg_2_items = data.loc[region2,['items']]
reg_3_items = data.loc[region3,['items']]
reg_4_items = data.loc[region4,['items']]



plt.tight_layout()

reg_1_age.describe()
```

Out[79]:

	age
count	15997.000000
mean	43.704132
std	14.085525
min	19.000000
25%	32.000000
50%	43.000000
75%	53.000000
max	74.000000

<Figure size 432x288 with 0 Axes>

In [80]:
```
reg_1_items.describe()
```

Out[80]:

	items
count	15997.000000
mean	4.510283
std	2.050402
min	1.000000
25%	3.000000
50%	4.000000
75%	6.000000
max	8.000000

In [81]:
```
reg_2_age.describe()
```

Out[81]:

	age
count	19994.000000
mean	56.609083
std	16.537368
min	28.000000
25%	42.000000
50%	57.000000
75%	71.000000
max	85.000000

In [82]:
```
reg_2_items.describe()
```

Out[82]:

	items
count	19994.000000
mean	4.512804
std	2.065467
min	1.000000
25%	3.000000
50%	5.000000
75%	6.000000
max	8.000000

In [83]:
```
reg_3_age.describe()
```

Out[83]:

	age
count	18000.000000
mean	45.646944
std	14.417935
min	18.000000
25%	34.000000
50%	45.000000
75%	57.000000
max	74.000000

In [84]:
```
reg_3_items.describe()
```

Out[84]:

	items
count	18000.000000
mean	4.494000
std	2.058095
min	1.000000
25%	3.000000
50%	4.000000
75%	6.000000
max	8.000000

In [85]:
```
reg_4_age.describe()
```

Out[85]:

	age
count	25988.000000
mean	38.752424
std	11.886239
min	18.000000
25%	29.000000
50%	39.000000
75%	49.000000
max	63.000000

In [86]:
```
reg_4_items.describe()
```

Out[86]:

	items
count	25988.000000
mean	4.503771
std	2.066920
min	1.000000
25%	3.000000
50%	5.000000
75%	6.000000
max	8.000000

In [87]:
```
data
```

Out[87]:

	in-store	age	items	amount	region
0	0	37	4	281.03	2
1	0	35	2	219.51	2
2	1	45	3	1525.70	4
3	1	46	3	715.25	3
4	1	33	4	1937.50	1
...	...	...	...	...	...
79995	1	71	3	558.82	1
79996	0	59	7	1932.00	3
79997	0	54	1	414.16	2
79998	1	49	4	335.32	1
79999	1	30	1	527.12	3

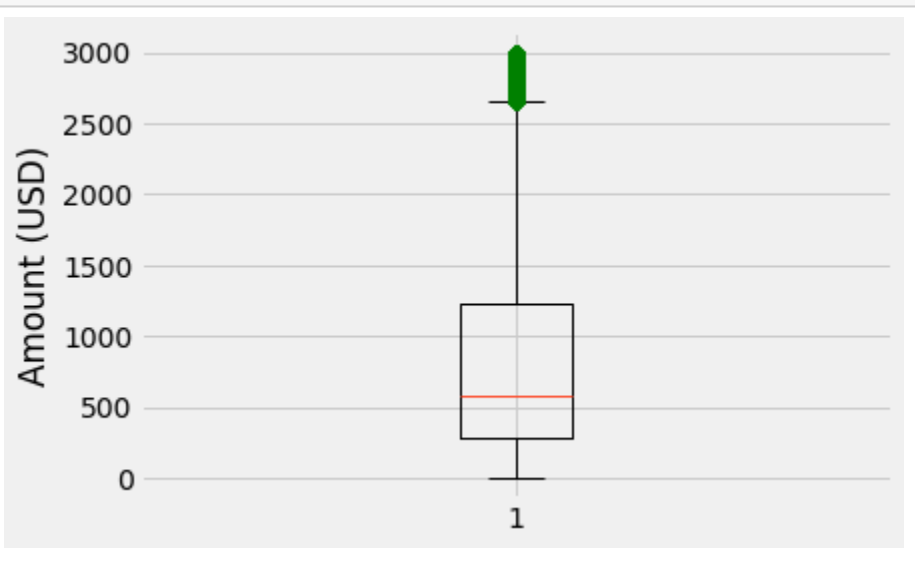
79979 rows × 5 columns

In [88]:
```
#filter out the items and amounts per region
reg_1 = data.loc[region1,['items','amount']]
reg_2 = data.loc[region2,['items','amount']]
reg_3 = data.loc[region3,['items','amount']]
reg_4 = data.loc[region4,['items','amount']]

#filter out the items per region
it_1 = data.loc[region1,['items']]
it_2 = data.loc[region2,['items']]
it_3 = data.loc[region3,['items']]
it_4 = data.loc[region4,['items']]
```

In [89]:
```
header = data.dtypes.index
print(header)
```

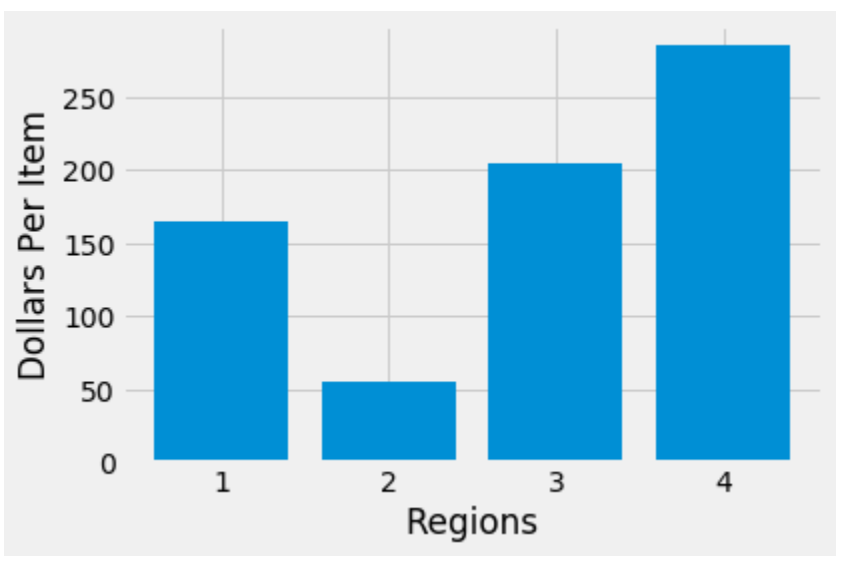
```
Index(['in-store', 'age', 'items', 'amount', 'region'], dtype='object')
```

In [90]:
```
A = data['amount']
plt.ylabel('Amount (USD)')
plt.boxplot(A,0,'gD')
plt.show()
```



In [91]:
```
reg1_iSum = reg_1['items'].sum()
reg1_aSum = reg_1['amount'].sum()
r1_per_trans = reg1_aSum / reg1_iSum

reg2_iSum = reg_2['items'].sum()
reg2_aSum = reg_2['amount'].sum()
r2_per_trans = reg2_aSum / reg2_iSum

reg3_iSum = reg_3['items'].sum()
reg3_aSum = reg_3['amount'].sum()
r3_per_trans = reg3_aSum / reg3_iSum

reg4_iSum = reg_4['items'].sum()
reg4_aSum = reg_4['amount'].sum()
r4_per_trans = reg4_aSum / reg4_iSum

print('REGIONAL RESULTS FOR $/item')
print('----------------------------')
print('region 1: ',r1_per_trans)
print('region 2: ',r2_per_trans)
print('region 3: ',r3_per_trans)
print('region 4: ',r4_per_trans)

regions = [1,2,3,4]
y_amt = [165.17, 55.86, 204.26, 285.08]
plt.bar(regions, y_amt,  label="Amount/Item")
plt.xlabel("Regions")
plt.ylabel("Dollars Per Item")
plt.tight_layout()
```

```
REGIONAL RESULTS FOR $/item
----------------------------
region 1:  165.17608977006554
region 2:  55.86277409147835
region 3:  204.26560690797606
region 4:  285.08073903831036
```

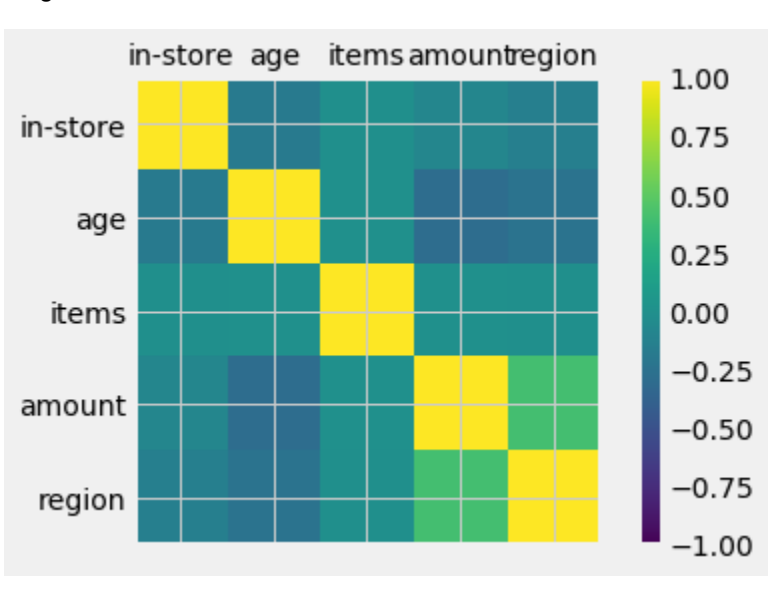


In [92]:
```
groupbySumItems = data.groupby(['region']).sum()

print(groupbySumItems)
```

```
        in-store      age   items        amount
region
1          15997   699135   72151  1.191762e+07
2              0  1131842   90229  5.040442e+06
3          10999   821645   80892  1.652345e+07
4          12994  1007098  117044  3.336699e+07
```

In [93]:
```
corr_mat = data.corr()
print(corr_mat)
fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.matshow(corr_mat, vmin=-1, vmax=1)
fig.colorbar(cax)
ticks = np.arange(0,5,1)
ax.set_xticks(ticks)
ax.set_yticks(ticks)
ax.set_xticklabels(data)
ax.set_yticklabels(data)
plt.show()
```

```
          in-store       age     items    amount    region
in-store  1.000000 -0.178180 -0.003897 -0.085573 -0.133171
age      -0.178180  1.000000  0.000657 -0.282033 -0.235370
items    -0.003897  0.000657  1.000000  0.000384 -0.001904
amount   -0.085573 -0.282033  0.000384  1.000000  0.403486
region   -0.133171 -0.235370 -0.001904  0.403486  1.000000
```



In [94]:
```
cov_mat = data.cov()
print(cov_mat)
```

```
           in-store          age     items         amount      region
in-store   0.250003    -1.400071 -0.004017     -30.860425   -0.075019
age       -1.400071   246.966189  0.021270   -3196.782841   -4.167305
items     -0.004017     0.021270  4.248751       0.570791   -0.004421
amount   -30.860425 -3196.782841  0.570791  520221.252295  327.874873
region    -0.075019    -4.167305 -0.004421     327.874873    1.269321
```