

Cohort Analysis for Assessing Customer Retention in the E-commerce Industry

Business Overview/Problem

E-Shop Pro, an e-commerce company, is facing declining customer retention rates despite successful customer acquisition efforts. The company collects customer data but hasn't fully utilized it to gain actionable insights for improving retention.

Rationale for the Project

Cohort analysis is used to track customer behavior over time, helping identify trends, pinpoint churn patterns, and devise strategies to improve retention and loyalty.

Aim of the Project

The project aims to utilize cohort analysis to segment customers based on their purchase behavior and understand key retention opportunities. By analyzing customer cohorts, the business can enhance engagement and marketing strategies.

Data Description

The dataset includes the following features:

- InvoiceNo: Transaction ID
- StockCode: Product code
- Description: Product details
- Quantity: Number of items purchased
- InvoiceDate: Transaction date
- UnitPrice: Price per unit
- CustomerID: Unique customer identifier
- Country: Customer location

Tech Stack

- Programming Language: Python
- Libraries: Numpy, Pandas, Matplotlib, Seaborn, Scikit-learn

```
In [1]: import pandas as pd
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt
import datetime as dt
```

Load Dataset

```
In [2]: data = pd.read_csv('Dataset-commerce.csv')
```

```
In [3]: data.head()
```

```
Out[3]:
```

	InvoiceNo	InvoiceDate	CustomerID	StockCode	Description	Quantity	UnitPrice	Country
0	536365	2010-12-01 08:26:00	17850.0	SC1734	Electronics	65	10.23	Egypt
1	536365	2010-12-01 08:26:00	17850.0	SC2088	Furniture	95	19.61	Mali
2	536365	2010-12-01 08:26:00	17850.0	SC3463	Books	78	61.49	Mali
3	536365	2010-12-01 08:26:00	17850.0	SC6228	Toys	15	24.73	South Africa
4	536365	2010-12-01 08:26:00	17850.0	SC2149	Toys	50	38.83	Rwanda

```
In [4]: ## descriptive stats
data.describe(include='all')
```

```
Out[4]:
```

	InvoiceNo	InvoiceDate	CustomerID	StockCode	Description	Quantity	Unit
count	541909	541909	406829.000000	541909	541909	541909.000000	541909.0
unique	25900	23260	NaN	9000	10	NaN	
top	573585	2011-10-31 14:41:00	NaN	SC2014	Sports Equipment	NaN	
freq	1114	1114	NaN	96	54765	NaN	
mean	NaN	NaN	15287.690570	NaN	NaN	50.534748	50.4
std	NaN	NaN	1713.600303	NaN	NaN	28.849367	28.5
min	NaN	NaN	12346.000000	NaN	NaN	1.000000	1.0
25%	NaN	NaN	13953.000000	NaN	NaN	26.000000	25.7
50%	NaN	NaN	15152.000000	NaN	NaN	51.000000	50.4
75%	NaN	NaN	16791.000000	NaN	NaN	76.000000	75.1
max	NaN	NaN	18287.000000	NaN	NaN	100.000000	100.0

In [5]: *#Check for mising values*

```
data.isnull().sum()
```

Out[5]: InvoiceNo 0
InvoiceDate 0
CustomerID 135080
StockCode 0
Description 0
Quantity 0
UnitPrice 0
Country 0
dtype: int64

In [6]: *#Dropping missing values*

```
data.dropna(inplace=True)
```

In [7]:

```
data.isnull().sum()
```

Out[7]: InvoiceNo 0
InvoiceDate 0
CustomerID 0
StockCode 0
Description 0
Quantity 0
UnitPrice 0
Country 0
dtype: int64

In [8]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
Int64Index: 406829 entries, 0 to 541908  
Data columns (total 8 columns):  
#   Column          Non-Null Count  Dtype  
---  ---  
0   InvoiceNo       406829 non-null object  
1   InvoiceDate     406829 non-null object  
2   CustomerID     406829 non-null float64  
3   StockCode      406829 non-null object  
4   Description     406829 non-null object  
5   Quantity       406829 non-null int64  
6   UnitPrice      406829 non-null float64  
7   Country        406829 non-null object  
dtypes: float64(2), int64(1), object(5)  
memory usage: 27.9+ MB
```

In [9]: *# Convert to datetime format*

```
data['InvoiceDate'] = data['InvoiceDate'].astype('datetime64[ns]')
```

In [10]: `data.info()`

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 406829 entries, 0 to 541908
Data columns (total 8 columns):
 #   Column          Non-Null Count  Dtype  
---  -
 0   InvoiceNo       406829 non-null object  
 1   InvoiceDate     406829 non-null datetime64[ns]
 2   CustomerID     406829 non-null float64 
 3   StockCode      406829 non-null object  
 4   Description    406829 non-null object  
 5   Quantity       406829 non-null int64   
 6   UnitPrice      406829 non-null float64 
 7   Country        406829 non-null object  
dtypes: datetime64[ns](1), float64(2), int64(1), object(4)
memory usage: 27.9+ MB
```

Exploratory Data Analysis

Analysis by Country

In [11]: `quantity_by_country = data.groupby(["Country"])["Quantity"].sum().reset_index()`

In [12]: `quantity_by_country.head()`

Out[12]:

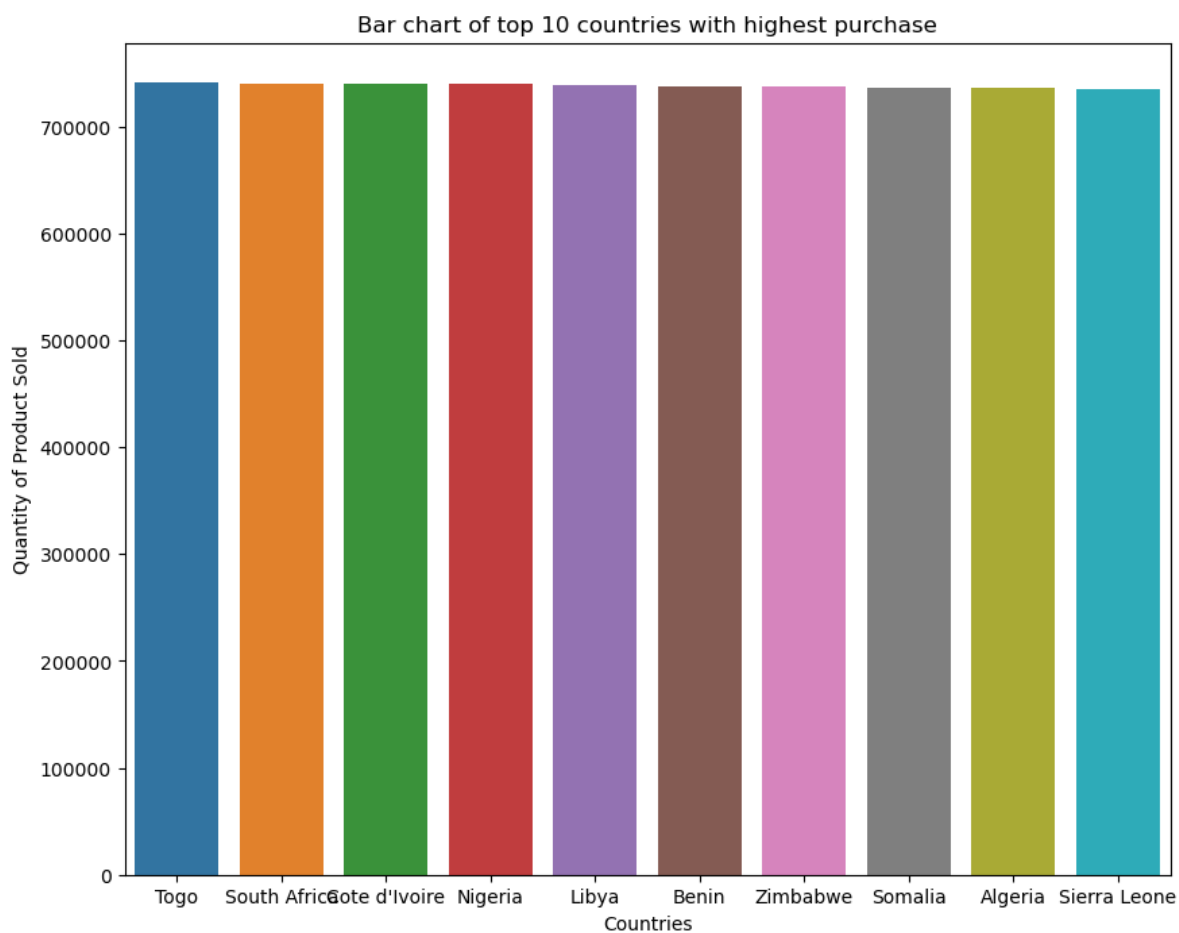
	index	Country	Quantity
0	23	Togo	741223
1	20	South Africa	740589
2	4	Cote d'Ivoire	740229
3	15	Nigeria	739708
4	9	Libya	739206

```
In [13]: # Corrected figure size and code
top_10_countries = quantity_by_country.head(10)

plt.figure(figsize=(10, 8))
ax = sns.barplot(x= "Country", y = "Quantity", data = top_10_countries)

plt.xlabel("Countries")
plt.ylabel("Quantity of Product Sold")
plt.title("Bar chart of top 10 countries with highest purchase")

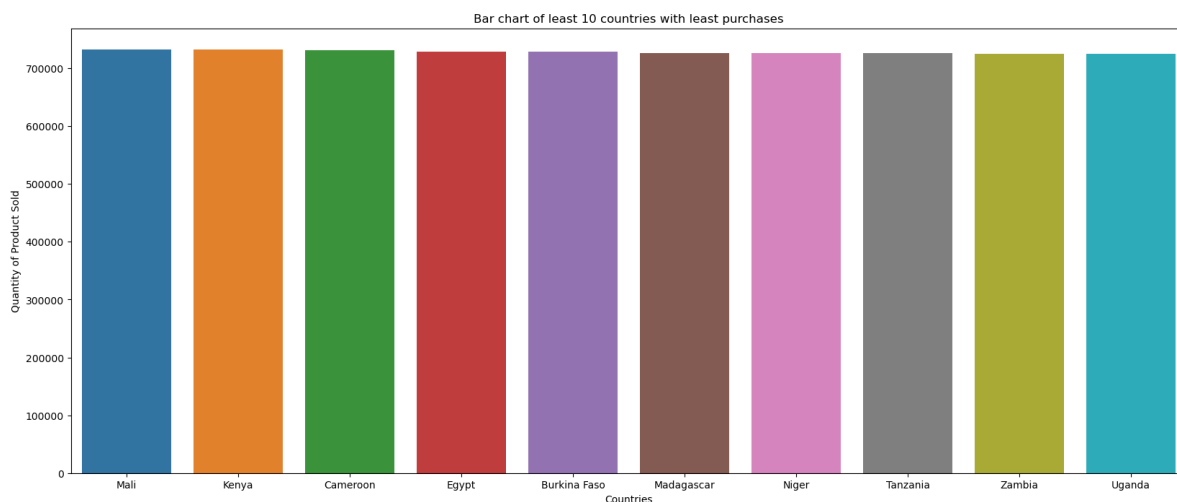
plt.show()
```



```
In [14]: # Plot the bottom 10 countries
least_10_countries = quantity_by_country.tail(10)

plt.figure(figsize=(20, 8))
ax = sns.barplot(x= "Country", y = "Quantity", data = least_10_countries)

plt.xlabel("Countries")
plt.ylabel("Quantity of Product Sold")
plt.title("Bar chart of least 10 countries with least purchases")
plt.show()
```



Analysis of countries and number of customers

```
In [15]: country_to_customer = data.groupby(['Country'])['CustomerID'].nunique().reset_

# Rename the column for clarity
country_to_customer = country_to_customer.sort_values("CustomerID", ascending
```

```
In [16]: # Display the result
country_to_customer.head()
```

Out[16]:

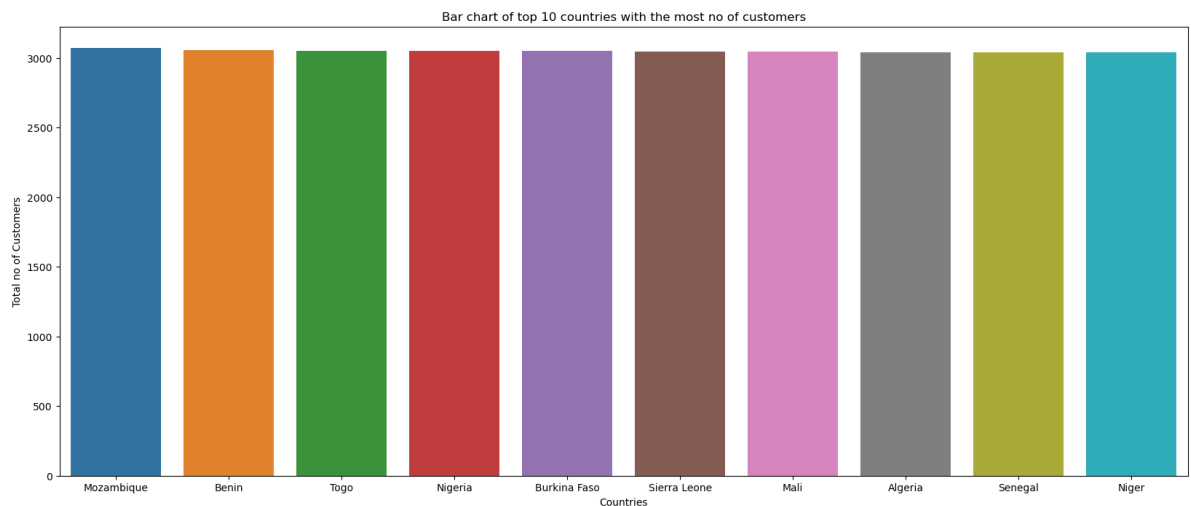
	index	Country	CustomerID
0	13	Mozambique	3072
1	1	Benin	3054
2	23	Togo	3053
3	15	Nigeria	3051
4	2	Burkina Faso	3049

```
In [17]: # Draw a plot of top 10 countries with ,most numner of customers

top_countries_to_customer = country_to_customer.head(10)

plt.figure(figsize = (20,8))
sns.barplot(x = 'Country', y = 'CustomerID', data = top_countries_to_customer)

plt.xlabel('Countries')
plt.ylabel('Total no of Customers')
plt.title('Bar chart of top 10 countries with the most no of customers')
plt.show()
```



In []:

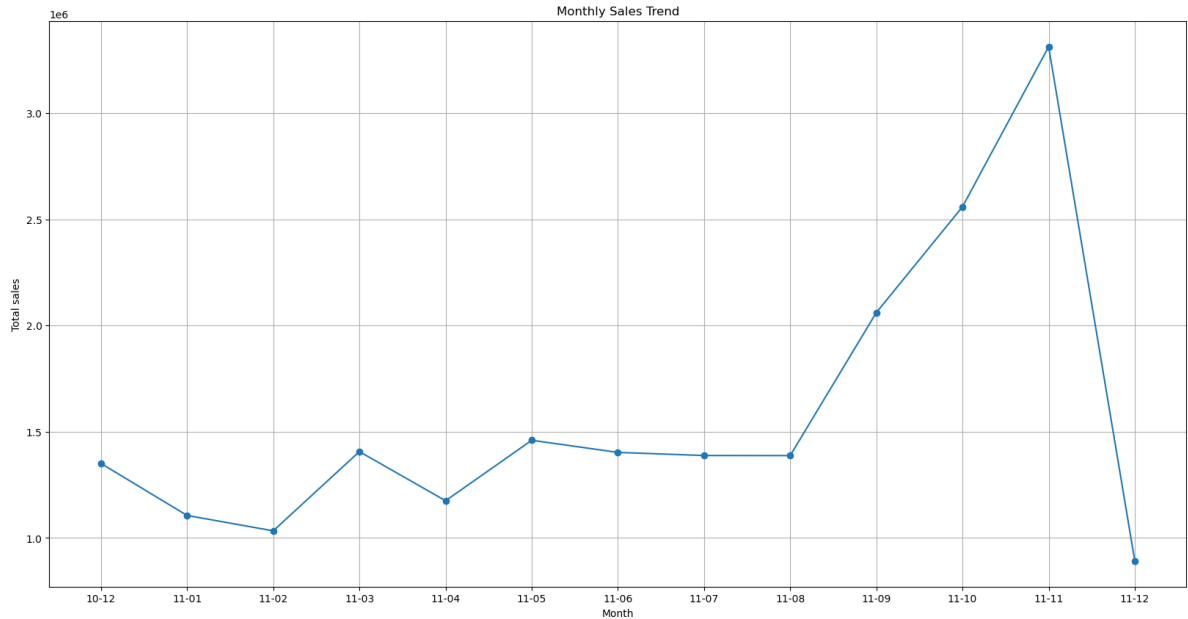
Sales trend Visualization

```
In [18]: monthly_sales = data.groupby(data['InvoiceDate']).dt.to_period('M')['Quantity']
```

```
In [19]: monthly_sales
```

```
Out[19]: InvoiceDate
2010-12      1350834
2011-01      1105886
2011-02      1033468
2011-03      1406617
2011-04      1175052
2011-05      1459981
2011-06      1402857
2011-07      1388119
2011-08      1387695
2011-09      2061181
2011-10      2559609
2011-11      3313177
2011-12       891630
Freq: M, Name: Quantity, dtype: int64
```

```
In [23]: # Create a line plot
plt.figure(figsize=(20,10))
plt.plot(monthly_sales.index.strftime('%y-%m'), monthly_sales.values, marker =
plt.xlabel('Month')
plt.ylabel('Total sales')
plt.title("Monthly Sales Trend")
plt.grid(True)
plt.show()
```



Cohort Analysis

```
In [24]: data.head()
```

Out[24]:

	InvoiceNo	InvoiceDate	CustomerID	StockCode	Description	Quantity	UnitPrice	Country
0	536365	2010-12-01 08:26:00	17850.0	SC1734	Electronics	65	10.23	Egypt
1	536365	2010-12-01 08:26:00	17850.0	SC2088	Furniture	95	19.61	Mali
2	536365	2010-12-01 08:26:00	17850.0	SC3463	Books	78	61.49	Mali
3	536365	2010-12-01 08:26:00	17850.0	SC6228	Toys	15	24.73	South Africa
4	536365	2010-12-01 08:26:00	17850.0	SC2149	Toys	50	38.83	Rwanda

```
In [27]: def get_month(x):
return dt.datetime(x.year, x.month, 1)
```

```
In [28]: data['InvoiceDate'] = data['InvoiceDate'].apply(get_month)
```


In [29]: `data.head()`

Out[29]:

	InvoiceNo	InvoiceDate	CustomerID	StockCode	Description	Quantity	UnitPrice	Country
0	536365	2010-12-01	17850.0	SC1734	Electronics	65	10.23	Egypt
1	536365	2010-12-01	17850.0	SC2088	Furniture	95	19.61	Mali
2	536365	2010-12-01	17850.0	SC3463	Books	78	61.49	Mali
3	536365	2010-12-01	17850.0	SC6228	Toys	15	24.73	South Africa
4	536365	2010-12-01	17850.0	SC2149	Toys	50	38.83	Rwanda

```
In [31]: def get_cohort_date(data):
         data["cohort_date"] = data.groupby("CustomerID")["InvoiceDate"].transform(
         return data["cohort_date"]
```

In [32]: `data["cohort_date"] = get_cohort_date(data)`

In [33]: `data`

Out[33]:

	InvoiceNo	InvoiceDate	CustomerID	StockCode	Description	Quantity	UnitPrice	Co
0	536365	2010-12-01	17850.0	SC1734	Electronics	65	10.23	
1	536365	2010-12-01	17850.0	SC2088	Furniture	95	19.61	
2	536365	2010-12-01	17850.0	SC3463	Books	78	61.49	
3	536365	2010-12-01	17850.0	SC6228	Toys	15	24.73	South
4	536365	2010-12-01	17850.0	SC2149	Toys	50	38.83	Ri
...	
541904	581587	2011-12-01	12680.0	SC5440	Furniture	25	36.53	Mozan
541905	581587	2011-12-01	12680.0	SC3576	Cloths	94	90.65	T
541906	581587	2011-12-01	12680.0	SC5283	Furniture	14	40.07	Sc
541907	581587	2011-12-01	12680.0	SC1099	Cosmetics	45	79.83	Se
541908	581587	2011-12-01	12680.0	SC9276	Toys	6	46.62	Et

406829 rows × 9 columns

Cohort Index

```
In [39]: def get_year_and_month(data, col):
         month = data[col].dt.month
         year = data[col].dt.year
         return month, year
```

```
In [40]: first_month, first_year = get_year_and_month(data, "cohort_date")
```

```
In [41]: first_month
```

```
Out[41]: 0      12
         1      12
         2      12
         3      12
         4      12
         ..
         541904    8
         541905    8
         541906    8
         541907    8
         541908    8
         Name: cohort date, Length: 406829, dtype: int64
```

```
In [42]: first_year
```

```
Out[42]: 0      2010
         1      2010
         2      2010
         3      2010
         4      2010
         ...
         541904    2011
         541905    2011
         541906    2011
         541907    2011
         541908    2011
         Name: cohort date, Length: 406829, dtype: int64
```

```
In [43]: latest_month, latest_year = get_year_and_month(data, "InvoiceDate")
```

```
In [44]: latest_month
```

```
Out[44]: 0      12
         1      12
         2      12
         3      12
         4      12
         ..
         541904    12
         541905    12
         541906    12
         541907    12
         541908    12
         Name: InvoiceDate, Length: 406829, dtype: int64
```

In [45]: latest_year

```
Out[45]: 0      2010
1      2010
2      2010
3      2010
4      2010
...
541904 2011
541905 2011
541906 2011
541907 2011
541908 2011
Name: InvoiceDate, Length: 406829, dtype: int64
```

Cohort index

```
In [46]: def create_cohort_index(first_month, first_year, latest_month, latest_year):
        year_diff = latest_year - first_year
        month_diff = latest_month - first_month
        index = year_diff*12 + month_diff + 1
        return index
```

```
In [48]: data["cohort_index"] = create_cohort_index(first_month, first_year, latest_mon
```

```
In [49]: data
```

```
Out[49]:
```

	InvoiceNo	InvoiceDate	CustomerID	StockCode	Description	Quantity	UnitPrice	Cc
0	536365	2010-12-01	17850.0	SC1734	Electronics	65	10.23	
1	536365	2010-12-01	17850.0	SC2088	Furniture	95	19.61	
2	536365	2010-12-01	17850.0	SC3463	Books	78	61.49	
3	536365	2010-12-01	17850.0	SC6228	Toys	15	24.73	South
4	536365	2010-12-01	17850.0	SC2149	Toys	50	38.83	R
...	
541904	581587	2011-12-01	12680.0	SC5440	Furniture	25	36.53	Mozar
541905	581587	2011-12-01	12680.0	SC3576	Cloths	94	90.65	T
541906	581587	2011-12-01	12680.0	SC5283	Furniture	14	40.07	Sc
541907	581587	2011-12-01	12680.0	SC1099	Cosmetics	45	79.83	Se
541908	581587	2011-12-01	12680.0	SC9276	Toys	6	46.62	Et

406829 rows × 10 columns

Cohort Table

```
In [51]: cohort_info = data.groupby(["cohort_date", "cohort_index"])["CustomerID"].nunique()
```

```
In [55]: cohort_info.rename(columns={"CustomerID": "Number of Customers"}, inplace=True)
```

```
In [57]: cohort_info
```

Out[57]:

	cohort date	cohort_index	Number of Customers
0	2010-12-01	1	948
1	2010-12-01	2	362
2	2010-12-01	3	317
3	2010-12-01	4	367
4	2010-12-01	5	341
...
86	2011-10-01	2	93
87	2011-10-01	3	46
88	2011-11-01	1	321
89	2011-11-01	2	43
90	2011-12-01	1	41

91 rows × 3 columns

```
In [60]: cohort_table = cohort_info.pivot(index = 'cohort_date', columns = ["cohort_index", "Number of Customers"], values = "Number of Customers")
```

In [61]:

```
cohort_table
```

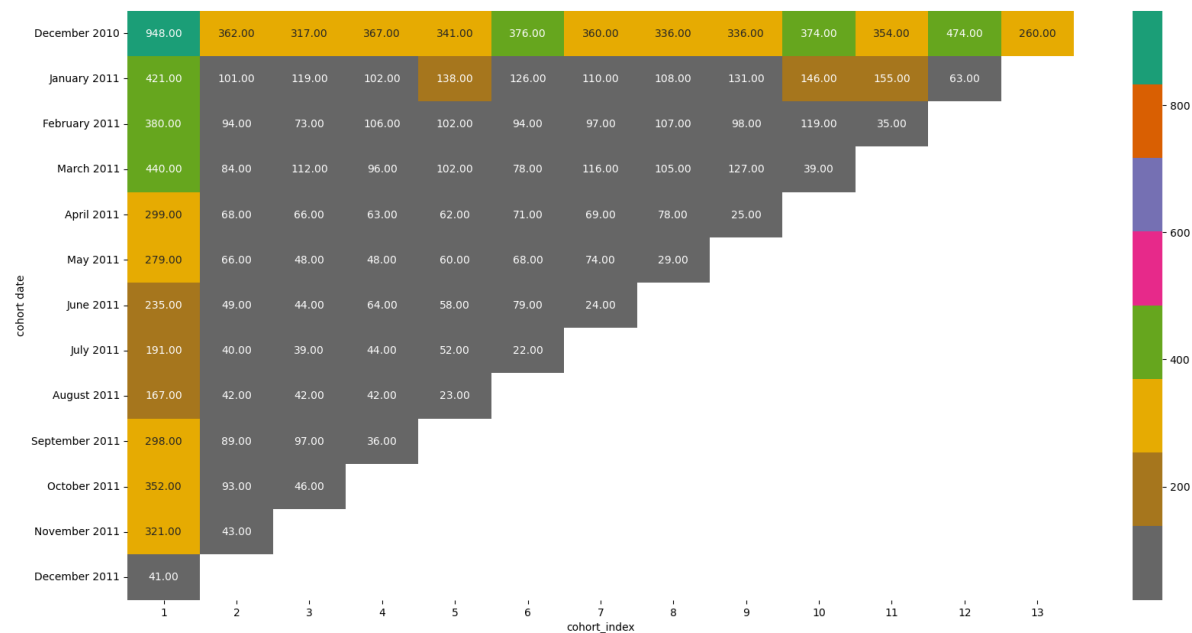
Out[61]:

cohort_index	1	2	3	4	5	6	7	8	9	10	11	12
cohort date												
December 2010	948.0	362.0	317.0	367.0	341.0	376.0	360.0	336.0	336.0	374.0	354.0	474.0
January 2011	421.0	101.0	119.0	102.0	138.0	126.0	110.0	108.0	131.0	146.0	155.0	63.0
February 2011	380.0	94.0	73.0	106.0	102.0	94.0	97.0	107.0	98.0	119.0	35.0	NaN
March 2011	440.0	84.0	112.0	96.0	102.0	78.0	116.0	105.0	127.0	39.0	NaN	NaN
April 2011	299.0	68.0	66.0	63.0	62.0	71.0	69.0	78.0	25.0	NaN	NaN	NaN
May 2011	279.0	66.0	48.0	48.0	60.0	68.0	74.0	29.0	NaN	NaN	NaN	NaN
June 2011	235.0	49.0	44.0	64.0	58.0	79.0	24.0	NaN	NaN	NaN	NaN	NaN
July 2011	191.0	40.0	39.0	44.0	52.0	22.0	NaN	NaN	NaN	NaN	NaN	NaN
August 2011	167.0	42.0	42.0	42.0	23.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
September 2011	298.0	89.0	97.0	36.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
October 2011	352.0	93.0	46.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
November 2011	321.0	43.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
December 2011	41.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

In [63]:

```
plt.figure(figsize = (20, 10))  
sns.heatmap(cohort_table, annot = True, cmap = "Dark2_r", fmt = ".2f")
```

Out[63]:



In [64]: `new_cohort_table = cohort_table.divide(cohort_table.iloc[:, 0], axis=0)`

In [65]: `new_cohort_table`

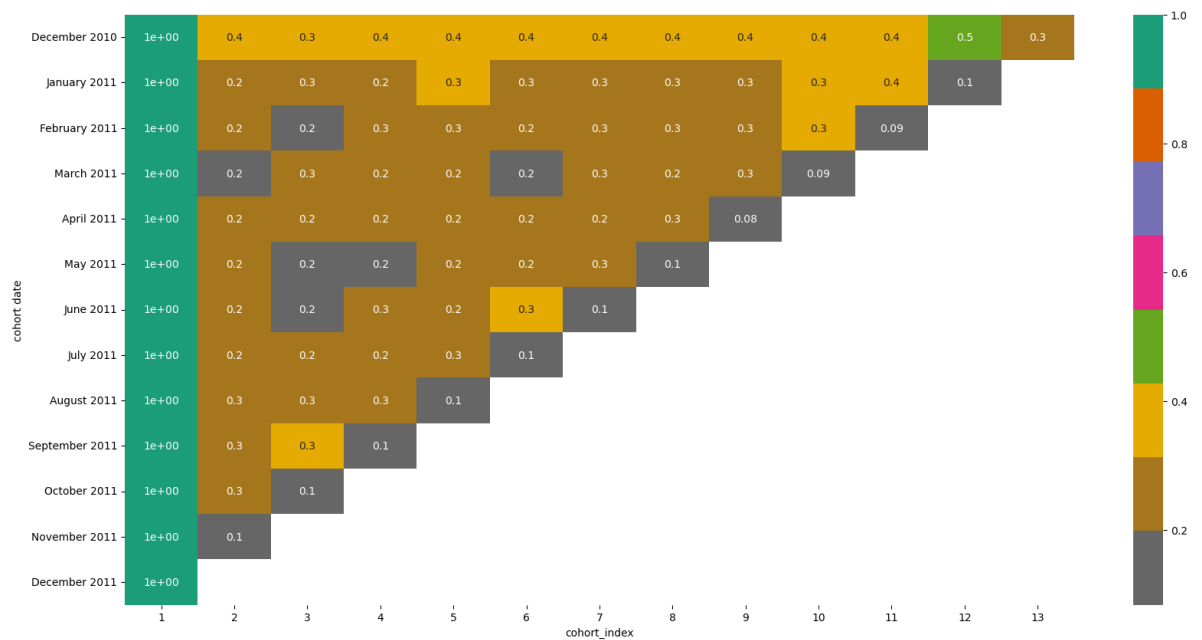
Out[65]:

cohort_index	1	2	3	4	5	6	7	8	9
cohort date									
December 2010	1.0	0.381857	0.334388	0.387131	0.359705	0.396624	0.379747	0.354430	0.354430
January 2011	1.0	0.239905	0.282660	0.242280	0.327791	0.299287	0.261283	0.256532	0.311160
February 2011	1.0	0.247368	0.192105	0.278947	0.268421	0.247368	0.255263	0.281579	0.257890
March 2011	1.0	0.190909	0.254545	0.218182	0.231818	0.177273	0.263636	0.238636	0.288636
April 2011	1.0	0.227425	0.220736	0.210702	0.207358	0.237458	0.230769	0.260870	0.083610
May 2011	1.0	0.236559	0.172043	0.172043	0.215054	0.243728	0.265233	0.103943	NaN
June 2011	1.0	0.208511	0.187234	0.272340	0.246809	0.336170	0.102128	NaN	NaN
July 2011	1.0	0.209424	0.204188	0.230366	0.272251	0.115183	NaN	NaN	NaN
August 2011	1.0	0.251497	0.251497	0.251497	0.137725	NaN	NaN	NaN	NaN
September 2011	1.0	0.298658	0.325503	0.120805	NaN	NaN	NaN	NaN	NaN
October 2011	1.0	0.264205	0.130682	NaN	NaN	NaN	NaN	NaN	NaN
November 2011	1.0	0.133956	NaN	NaN	NaN	NaN	NaN	NaN	NaN
December 2011	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

```
In [66]: plt.figure(figsize = (20, 10))
```

```
sns.heatmap(cow_cohort_table, annot = True, cmap = "Dark2_r", fmt = ".0")
```

```
Out[66]: <AxesSubplot: xlabel='cohort_index', ylabel='cohort date'>
```



Quantity Bought

```
In [70]: quantity_bought = data.groupby(["cohort_date", "cohort_index"])["Quantity"].sum()
```

```
In [71]: quantity_bought
```

```
Out[71]:
```

	cohort_date	cohort_index	Quantity
0	2010-12-01	1	1350834
1	2010-12-01	2	546873
2	2010-12-01	3	462374
3	2010-12-01	4	613244
4	2010-12-01	5	510943
...
86	2011-10-01	2	181888
87	2011-10-01	3	44776
88	2011-11-01	1	574330
89	2011-11-01	2	56021
90	2011-12-01	1	50955

91 rows × 3 columns

```
In [74]: quantity_table = quantity_bought.pivot(index = "cohort date", columns = ["cohort index", "quantity"])
```

```
In [75]: quantity_table
```

Out[75]:

cohort_index	1	2	3	4	5	6	7	8
cohort date								
December 2010	1350834.0	546873.0	462374.0	613244.0	510943.0	636443.0	597676.0	604504.0
January 2011	559013.0	121074.0	155931.0	120023.0	194675.0	166585.0	144835.0	149848.0
February 2011	450020.0	74008.0	98850.0	122918.0	111386.0	104214.0	108442.0	135276.0
March 2011	563434.0	74215.0	125519.0	119480.0	122078.0	107554.0	139852.0	182259.0
April 2011	371021.0	71541.0	66410.0	65115.0	70639.0	105065.0	97319.0	110572.0
May 2011	308885.0	55453.0	44637.0	45690.0	76473.0	118534.0	97156.0	41183.0
June 2011	285867.0	47847.0	35516.0	76483.0	88504.0	127400.0	31264.0	NaN
July 2011	254889.0	25983.0	68721.0	69466.0	91529.0	26831.0	NaN	NaN
August 2011	243542.0	80156.0	144849.0	141888.0	45405.0	NaN	NaN	NaN
September 2011	420688.0	124681.0	135347.0	44631.0	NaN	NaN	NaN	NaN
October 2011	598521.0	181888.0	44776.0	NaN	NaN	NaN	NaN	NaN
November 2011	574330.0	56021.0	NaN	NaN	NaN	NaN	NaN	NaN
December 2011	50955.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN


```
In [76]: plt.figure(figsize = (20, 10))
```

```
sns.heatmap(quantity_table, annot = True, cmap = "Dark2_r", fmt = ".2f")
```

```
Out[76]: <AxesSubplot: xlabel='cohort_index', ylabel='cohort date'>
```

