In [1]:

```python
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:

```python
df = pd.read_excel('ANZ synthesised transaction dataset.xlsx')
```

In [3]:

```python
df.head()
```

Out[3]:

	status	card_present_flag	bpay_biller_code	account	currency	long_lat	txn_description	merchant_id	merchant_c
0	authorized	1.0	NaN	ACC-1598451071	AUD	153.41 -27.95	POS	81c48296-73be-44a7-befa-d053f48ce7cd	
1	authorized	0.0	NaN	ACC-1598451071	AUD	153.41 -27.95	SALES-POS	830a451c-316e-4a6a-bf25-e37caedca49e	
2	authorized	1.0	NaN	ACC-1222300524	AUD	151.23 -33.94	POS	835c231d-8cdf-4e96-859d-e9d571760cf0	
3	authorized	1.0	NaN	ACC-1037050564	AUD	153.10 -27.66	SALES-POS	48514682-c78a-4a88-b0da-2d6302e64673	
4	authorized	1.0	NaN	ACC-1598451071	AUD	153.41 -27.95	SALES-POS	b4e02c10-0852-4273-b8fd-7b3395e32eb0	

5 rows × 23 columns

In [4]:

```python
df.describe()
```

Out[4]:

	card_present_flag	merchant_code	balance	age	amount
count	7717.000000	883.0	12043.000000	12043.000000	12043.000000
mean	0.802644	0.0	14704.195553	30.582330	187.933588
std	0.398029	0.0	31503.722652	10.046343	592.599934
min	0.000000	0.0	0.240000	18.000000	0.100000
25%	1.000000	0.0	3158.585000	22.000000	16.000000
50%	1.000000	0.0	6432.010000	28.000000	29.000000
75%	1.000000	0.0	12465.945000	38.000000	53.655000
max	1.000000	0.0	267128.520000	78.000000	8835.980000

In [5]:

```
df.describe(include='object')
```

Out[5]:

	status	bpay_biller_code	account	currency	long_lat	txn_description	merchant_id	first_name	gender	merc
count	12043	885	12043	12043	12043	12043	7717	12043	12043	
unique	2	3	100	1	100	6	5725	80	2	
top	authorized	0	ACC-1598451071	AUD	153.41 -27.95	SALES-POS	106e1272-44ab-4dcb-a438-dd98e0071e51	Michael	M	
freq	7717	883	578	12043	578	3934	14	746	6285	

In [6]:

```
df.info()
```

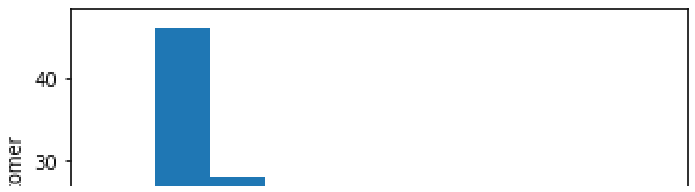
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12043 entries, 0 to 12042
Data columns (total 23 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   status             12043 non-null  object
 1   card_present_flag  7717 non-null   float64
 2   bpay_biller_code   885 non-null    object
 3   account            12043 non-null  object
 4   currency           12043 non-null  object
 5   long_lat           12043 non-null  object
 6   txn_description    12043 non-null  object
 7   merchant_id        7717 non-null   object
 8   merchant_code      883 non-null    float64
 9   first_name         12043 non-null  object
 10  balance            12043 non-null  float64
 11  date               12043 non-null  datetime64[ns]
 12  gender             12043 non-null  object
 13  age                12043 non-null  int64
 14  merchant_suburb    7717 non-null   object
 15  merchant_state     7717 non-null   object
 16  extraction         12043 non-null  object
 17  amount             12043 non-null  float64
 18  transaction_id     12043 non-null  object
 19  country            12043 non-null  object
 20  customer_id        12043 non-null  object
 21  merchant_long_lat  7717 non-null   object
 22  movement           12043 non-null  object
dtypes: datetime64[ns](1), float64(4), int64(1), object(17)
memory usage: 2.1+ MB
```

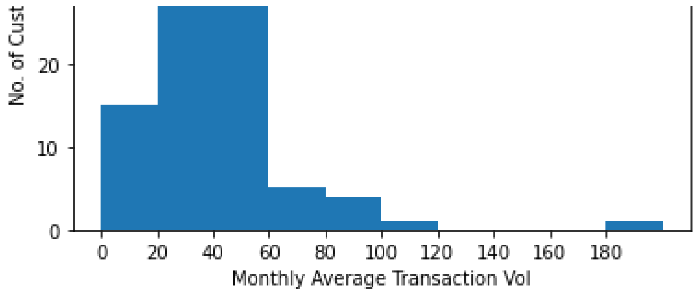
In [7]:

```
# Calculate the Monthly Average Transaction Vol

Vol = df.amount.groupby(df.account)
Ave_Vol = Vol.count()/3

plt.hist(Ave_Vol, bins = range(0,220,20))
plt.xlabel('Monthly Average Transaction Vol')
plt.ylabel('No. of Customer')
plt.xticks(np.arange(0, max(Ave_Vol)+1, 20.0))
plt.show()
```



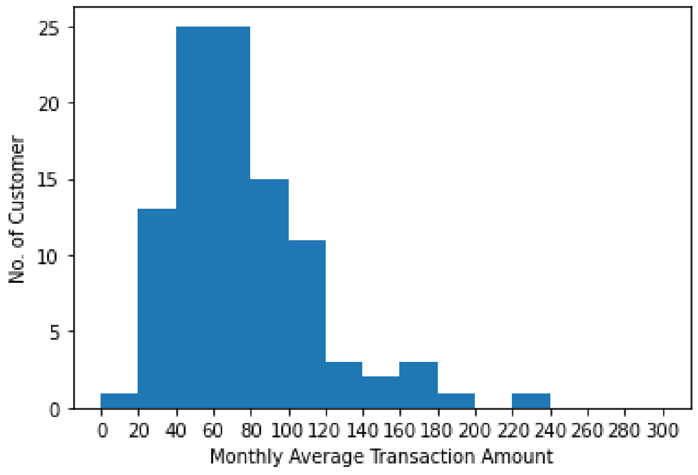


In [8]:

```
# Calculate the Monthly Average Transactiojn Amount

Ave_Amt = Vol.mean()/3

plt.hist(Ave_Amt,bins = range(0,320,20))
plt.xlabel('Monthly Average Transaction Amount')
plt.ylabel('No. of Customer')
plt.xticks(np.arange(0, 320, 20.0))
plt.show()
```



In [9]:

```
# Calculate the distance from respective customer to Merchant they have transaction with

df_L = df[['account','long_lat','merchant_long_lat']]
#M_list = df[merchant_long_lat.groupby(df.account)
df_L.dropna(inplace=True)
df_L.drop_duplicates(inplace=True)
df_L.sort_values('long_lat', inplace=True)
df_L.reset_index(drop = True, inplace = True)

# Get names of indexes for which column Account  has value 'ACC-2901672282'
indexNames = df_L[df_L['account'] == 'ACC-2901672282'].index

# Delete these row indexes from dataFrame
df_L.drop(indexNames , inplace=True)
```

```
<ipython-input-9-f9810f5193c5>:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_g
uide/indexing.html#returning-a-view-versus-a-copy
  df_L.dropna(inplace=True)
<ipython-input-9-f9810f5193c5>:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_g
uide/indexing.html#returning-a-view-versus-a-copy
  df_L.drop_duplicates(inplace=True)
<ipython-input-9-f9810f5193c5>:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_g
```

```
uide/indexing.html#returning-a-view-versus-a-copy
  df_L.sort_values('long_lat', inplace=True)
C:\Users\akell\anaconda3\lib\site-packages\pandas\core\frame.py:4163: SettingWithCopyWarn
ing:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_g
uide/indexing.html#returning-a-view-versus-a-copy
  return super().drop(
```

In [10]:

```
df_L.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 5639 entries, 0 to 5638
Data columns (total 3 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   account            5639 non-null   object
 1   long_lat           5639 non-null   object
 2   merchant_long_lat  5639 non-null   object
dtypes: object(3)
memory usage: 176.2+ KB
```

In [11]:

```
df_L['Long'] = df_L.long_lat.str.split(' ').str[0].astype(float)
df_L['Lat'] = df_L.long_lat.str.split(' ').str[1].astype(float)
df_L['Long2'] = df_L.merchant_long_lat.str.split(' ').str[0].astype(float)
df_L['Lat2'] = df_L.merchant_long_lat.str.split(' ').str[1].astype(float)
df_L.head()
```

```
<ipython-input-11-389ab3983f0a>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_g
uide/indexing.html#returning-a-view-versus-a-copy
  df_L['Long'] = df_L.long_lat.str.split(' ').str[0].astype(float)
<ipython-input-11-389ab3983f0a>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_g
uide/indexing.html#returning-a-view-versus-a-copy
  df_L['Lat'] = df_L.long_lat.str.split(' ').str[1].astype(float)
<ipython-input-11-389ab3983f0a>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_g
uide/indexing.html#returning-a-view-versus-a-copy
  df_L['Long2'] = df_L.merchant_long_lat.str.split(' ').str[0].astype(float)
<ipython-input-11-389ab3983f0a>:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_g
uide/indexing.html#returning-a-view-versus-a-copy
  df_L['Lat2'] = df_L.merchant_long_lat.str.split(' ').str[1].astype(float)
```

Out[11]:

	account	long_lat	merchant_long_lat	Long	Lat	Long2	Lat2
0	ACC-1990648130	114.62 -28.80	153.45 -28.85	114.62	-28.8	153.45	-28.85
1	ACC-1990648130	114.62 -28.80	114.63 -28.76	114.62	-28.8	114.63	-28.76
[illegible]	[illegible]	114.62 -	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

	account	long_lat	merchant_long_lat	Long	Lat	Long2	Lat2
2	ACC-1990648130	28.80	151.02 -33.88	114.62	-28.8	151.02	-33.88
3	ACC-1990648130	114.62 -28.80	114.61 -28.77	114.62	-28.8	114.61	-28.77
4	ACC-1990648130	114.62 -28.80	145.15 -37.83	114.62	-28.8	145.15	-37.83

In [12]:

```
df_L.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 5639 entries, 0 to 5638
Data columns (total 7 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   account            5639 non-null   object 
 1   long_lat           5639 non-null   object 
 2   merchant_long_lat  5639 non-null   object 
 3   Long               5639 non-null   float64
 4   Lat                5639 non-null   float64
 5   Long2              5639 non-null   float64
 6   Lat2               5639 non-null   float64
dtypes: float64(4), object(3)
memory usage: 352.4+ KB
```

In [13]:

```
from math import radians, cos, sin, asin, sqrt
def haversine(lon1, lat1, lon2, lat2):
    """
    Calculate the great circle distance between two points
    on the earth (specified in decimal degrees)
    """
    # convert decimal degrees to radians
    lon1, lat1, lon2, lat2 = map(radians, [lon1, lat1, lon2, lat2])
    # haversine formula
    dlon = lon2 - lon1
    dlat = lat2 - lat1
    a = sin(dlat/2)**2 + cos(lat1) * cos(lat2) * sin(dlon/2)**2
    c = 2 * asin(sqrt(a))
    # Radius of earth in kilometers is 6371
    km = 6371* c
    return km

df_L['distance'] = [haversine(df_L.Long[i],df_L.Lat[i],df_L.Long2[i],df_L.Lat2[i]) for i
in range(len(df_L))]
```

```
<ipython-input-13-e4f86c0568fa>:18: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_g
uide/indexing.html#returning-a-view-versus-a-copy
  df_L['distance'] = [haversine(df_L.Long[i],df_L.Lat[i],df_L.Long2[i],df_L.Lat2[i]) for
i in range(len(df_L))]
```

In [14]:

```
df_L.shape
```

Out[14]:

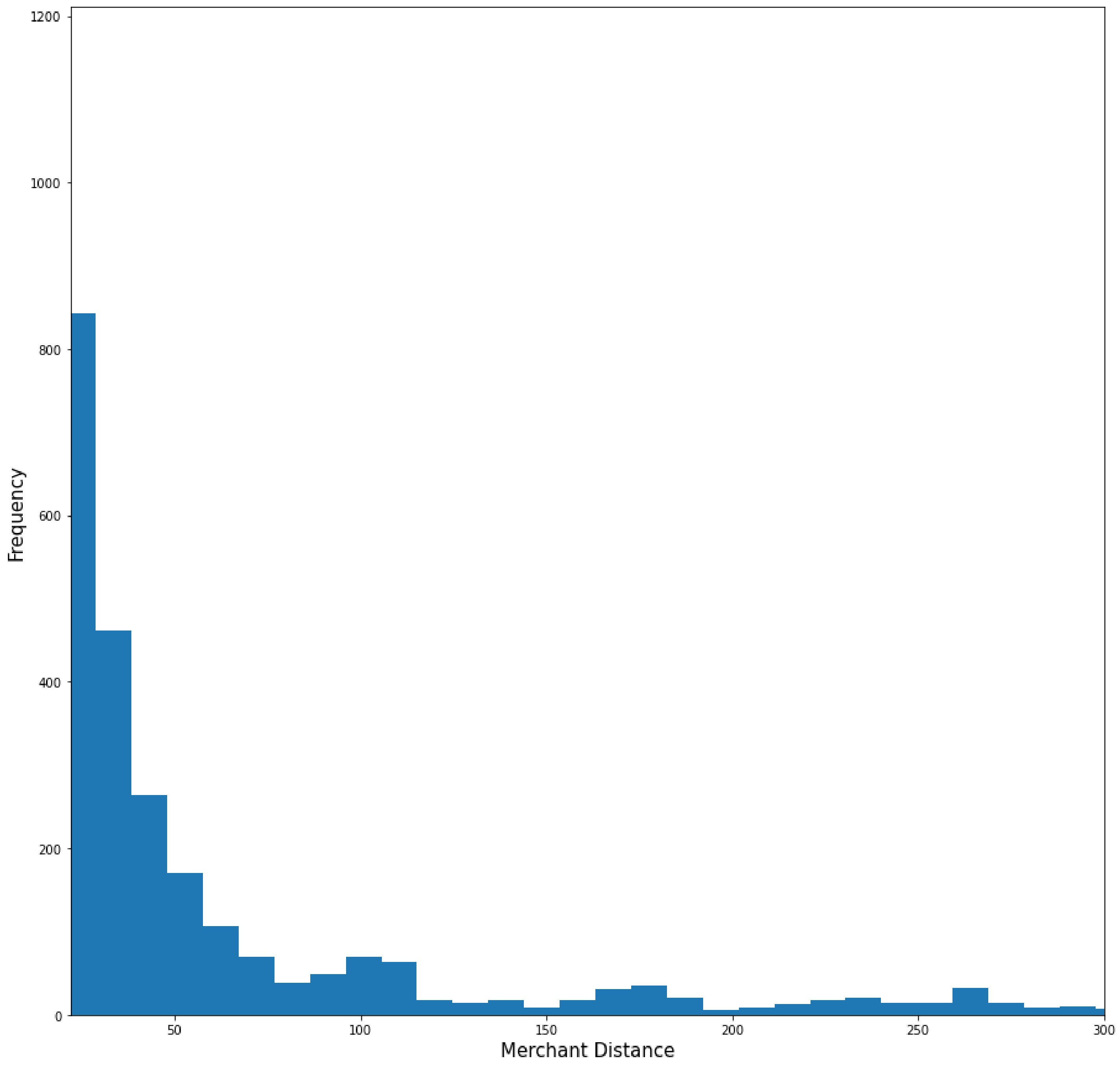
```
(5639, 8)
```

In [15]:

```
df_L.distance.hist(bins = 400, grid=False, xlabelsize=10, ylabelsize=10, figsize = (15,1
5) )
plt.xlabel('Merchant Distance', fontsize=15)
plt.ylabel('Frequency',fontsize=15)
plt.xlim([22.0,300.0])
```

Out[15]:

```
(22.0, 300.0)
```



In [ ]: