Prepared by Asif Bhat

Plotly / Plotly Express Tutorial using ANZ Dataset

In [1]:
```
import numpy as np
import pandas as pd
import plotly.express as px
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
import plotly.graph_objects as go
import plotly.express as px
import plotly.figure_factory as ff
from plotly.subplots import make_subplots
import plotly.io as pio
```

In [2]:
```
df = pd.read_excel('ANZ synthesised transaction dataset.xlsx')
df.head()
```

Out[2]:

	status	card_present_flag	bpay_biller_code	account	currency	long_lat	txn_description	merchant_id	merchant_code	first_name	...	age
0	authorized	1.0	NaN	ACC-1598451071	AUD	153.41 -27.95	POS	81c48296-73be-44a7-befa-d053f48ce7cd	NaN	Diana	...	26
1	authorized	0.0	NaN	ACC-1598451071	AUD	153.41 -27.95	SALES-POS	830a451c-316e-4a6a-bf25-e37caedca49e	NaN	Diana	...	26
2	authorized	1.0	NaN	ACC-1222300524	AUD	151.23 -33.94	POS	835c231d-8cdf-4e96-859d-e9d571760cf0	NaN	Michael	...	38
3	authorized	1.0	NaN	ACC-1037050564	AUD	153.10 -27.66	SALES-POS	48514682-c78a-4a88-b0da-2d6302e64673	NaN	Rhonda	...	40
4	authorized	1.0	NaN	ACC-1598451071	AUD	153.41 -27.95	SALES-POS	b4e02c10-0852-4273-b8fd-7b3395e32eb0	NaN	Diana	...	26

5 rows × 23 columns

In [3]:
```
df.shape
```

Out[3]:
```
(12043, 23)
```

In [4]:
```
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 [5]:
```
df.shape
```

Out[5]:
```
(12043, 23)
```

In [6]:
```
df.head()
```

Out[6]:

	status	card_present_flag	bpay_biller_code	account	currency	long_lat	txn_description	merchant_id	merchant_code	first_name	...	age
0	authorized	1.0	NaN	ACC-1598451071	AUD	153.41 -27.95	POS	81c48296-73be-44a7-befa-d053f48ce7cd	NaN	Diana	...	26
1	authorized	0.0	NaN	ACC-1598451071	AUD	153.41 -27.95	SALES-POS	830a451c-316e-4a6a-bf25-e37caedca49e	NaN	Diana	...	26
2	authorized	1.0	NaN	ACC-1222300524	AUD	151.23 -33.94	POS	835c231d-8cdf-4e96-859d-e9d571760cf0	NaN	Michael	...	38
3	authorized	1.0	NaN	ACC-1037050564	AUD	153.10 -27.66	SALES-POS	48514682-c78a-4a88-b0da-2d6302e64673	NaN	Rhonda	...	40
4	authorized	1.0	NaN	ACC-1598451071	AUD	153.41 -27.95	SALES-POS	b4e02c10-0852-4273-b8fd-7b3395e32eb0	NaN	Diana	...	26

5 rows × 23 columns

In [7]:
```
df.isnull().sum() # Drop 'bpay_biller_code' & 'merchant_code' as majority of the values are NULLS
```

Out[7]:
```
status                0
card_present_flag     4326
bpay_biller_code     11158
account               0
currency              0
long_lat              0
txn_description       0
merchant_id           4326
merchant_code        11160
first_name            0
balance               0
date                  0
gender                0
age                   0
merchant_suburb       4326
merchant_state        4326
extraction            0
amount                0
transaction_id        0
country               0
customer_id           0
merchant_long_lat     4326
movement              0
dtype: int64
```

In [8]:
```
df.country.value_counts() # This can be dropped as we are only dealing with one country
```

Out[8]:
```
Australia    12043
Name: country, dtype: int64
```

In [9]:
```
df.currency.value_counts() # This can be also dropped as we are only dealing with just one currency
```

Out[9]:
```
AUD    12043
Name: currency, dtype: int64
```

In [10]:
```
# Drop 'bpay_biller_code' ,Currency and 'merchant_code' columns.
df.drop(['bpay_biller_code','merchant_code', 'currency','country'],axis=1,inplace=True)
```

In [11]:
```
# Duplicates
df.duplicated().sum() # NO Duplicates
```

Out[11]:
```
0
```

In [12]:
```
# Create Age buckets
df['age_group']=pd.cut(df.age,[0,20,30,40,50,60,99999],labels=['<20','20-30','30-40','40-50','50-60','>60'])
```

In [13]:
```
# Change datatype of extraction to datetime
df['extraction']= pd.to_datetime(df['extraction'])
```

In [14]:
```
# Create date helper columns
df['month'] =df['date'].dt.month_name()
df['day'] = df['date'].dt.day_name()
df['hour']= df.extraction.dt.hour
df.head()
```

Out[14]:

	status	card_present_flag	account	long_lat	txn_description	merchant_id	first_name	balance	date	gender	...	extraction	amount
0	authorized	1.0	ACC-1598451071	153.41 -27.95	POS	81c48296-73be-44a7-befa-d053f48ce7cd	Diana	35.39	2018-08-01	F	...	2018-08-01 01:01:15+00:00	16.25
1	authorized	0.0	ACC-1598451071	153.41 -27.95	SALES-POS	830a451c-316e-4a6a-bf25-e37caedca49e	Diana	21.20	2018-08-01	F	...	2018-08-01 01:13:45+00:00	14.19
2	authorized	1.0	ACC-1222300524	151.23 -33.94	POS	835c231d-8cdf-4e96-859d-e9d571760cf0	Michael	5.71	2018-08-01	M	...	2018-08-01 01:26:15+00:00	6.42
3	authorized	1.0	ACC-1037050564	153.10 -27.66	SALES-POS	48514682-c78a-4a88-b0da-2d6302e64673	Rhonda	2117.22	2018-08-01	F	...	2018-08-01 01:38:45+00:00	40.90
4	authorized	1.0	ACC-1598451071	153.41 -27.95	SALES-POS	b4e02c10-0852-4273-b8fd-7b3395e32eb0	Diana	17.95	2018-08-01	F	...	2018-08-01 01:51:15+00:00	3.25

5 rows × 23 columns

In [15]:
```
df.card_present_flag = df.card_present_flag.astype('Int64')
df.head()
```

Out[15]:

	status	card_present_flag	account	long_lat	txn_description	merchant_id	first_name	balance	date	gender	...	extraction	amount
0	authorized	1	ACC-1598451071	153.41 -27.95	POS	81c48296-73be-44a7-befa-d053f48ce7cd	Diana	35.39	2018-08-01	F	...	2018-08-01 01:01:15+00:00	16.25
1	authorized	0	ACC-1598451071	153.41 -27.95	SALES-POS	830a451c-316e-4a6a-bf25-e37caedca49e	Diana	21.20	2018-08-01	F	...	2018-08-01 01:13:45+00:00	14.19
2	authorized	1	ACC-1222300524	151.23 -33.94	POS	835c231d-8cdf-4e96-859d-e9d571760cf0	Michael	5.71	2018-08-01	M	...	2018-08-01 01:26:15+00:00	6.42
3	authorized	1	ACC-1037050564	153.10 -27.66	SALES-POS	48514682-c78a-4a88-b0da-2d6302e64673	Rhonda	2117.22	2018-08-01	F	...	2018-08-01 01:38:45+00:00	40.90
4	authorized	1	ACC-1598451071	153.41 -27.95	SALES-POS	b4e02c10-0852-4273-b8fd-7b3395e32eb0	Diana	17.95	2018-08-01	F	...	2018-08-01 01:51:15+00:00	3.25

5 rows × 23 columns

In [16]:
```
cols = ['card_present_flag', 'status', 'txn_description' , 'movement' , 'gender', 'merchant_state']

#Subplot initialization
fig = make_subplots(
                    rows=3,
                    cols=2,
                    subplot_titles=('card_present_flag', 'status', 'txn_description' , 'movement','gender', 'merchant_st
                    horizontal_spacing=0.2,
                    vertical_spacing=0.2
                   )
# Adding subplots
count=0
for i in range(1,4):
    for j in range(1,3):
        fig.add_trace(go.Bar(x=df[cols[count]].value_counts().index,
                             y=df[cols[count]].value_counts(),
                             name=cols[count],
                             textposition='auto',
                             text= [str(i) + '%' for i in (df[cols[count]].value_counts(normalize=True)*100).round(2).to
                            ),
                      row=i,col=j)
        fig.update_xaxes(showgrid=False)
        fig.update_yaxes(showgrid=False)
        count+=1
fig.update_layout(
                    title=dict(text = "Analyze Categorical variables (Frequency / Percentage)",x=0.5,y=0.95),
                    title_font_size=20,
                    showlegend=False,
                    width = 980,
                    height = 920,
                    margin=dict(l=80, r=80, t=150, b=80)
                   )
fig.show()
```

Insights

- Most of the transactions (**80.26%**) have been done via cards (credit / Debit Card).
- Almost **64.08%** transactions were authorized and rest were posted.
- **92.67%** transactions are of type debit. Rest transactions are credit.
- Looks like majority of the transactions use "SALES-POS" & "POS' transaction mode.
- **Males** tend to do more transactions as compared to **females**.
- **NSW , VIC , QLD** are most busy merchant states.
- **ACT & TAS** are least busy states.

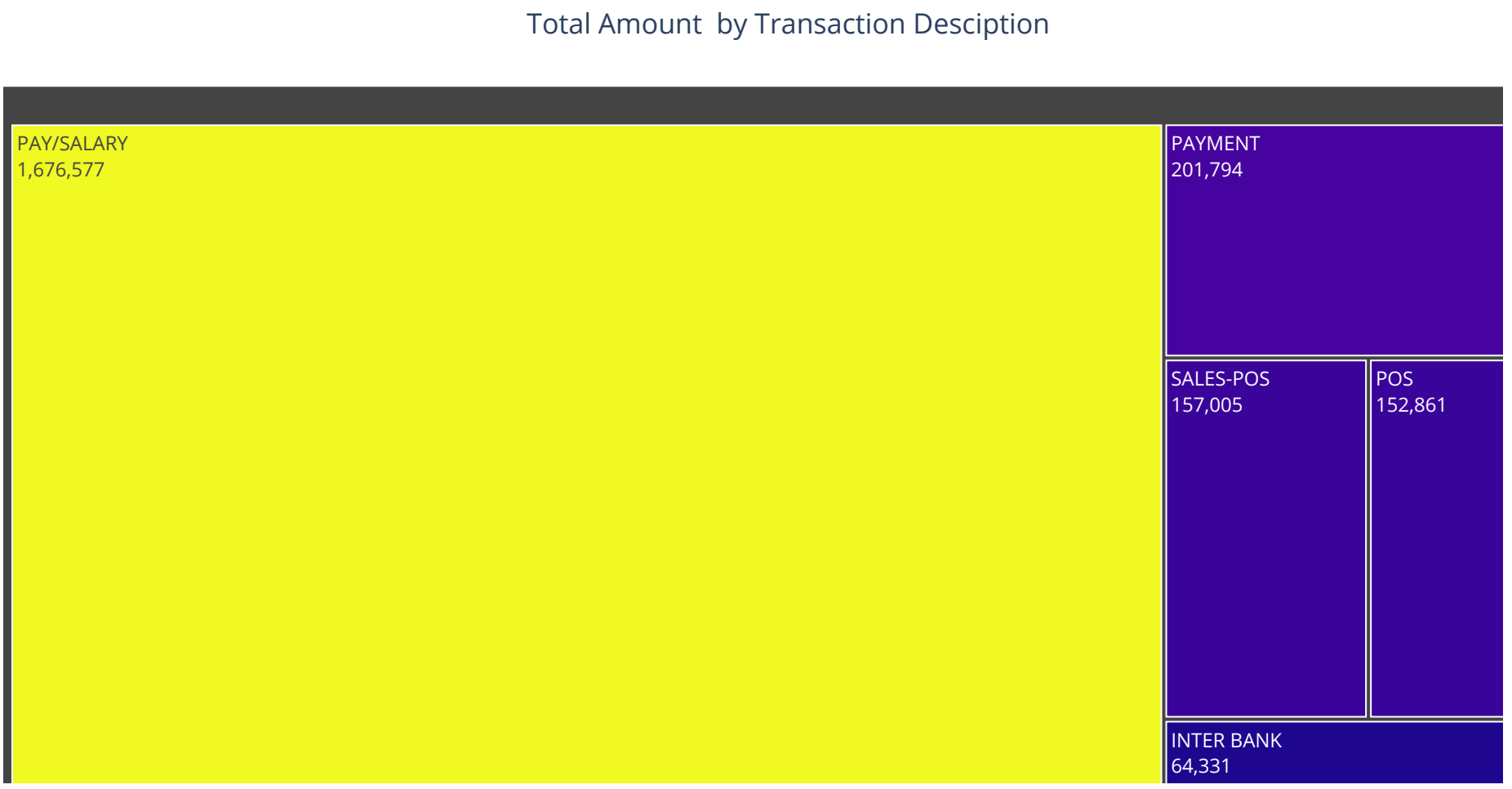
In [17]:
```
df0_grp=df.groupby(by='txn_description').sum()[['amount']].reset_index()
df0_grp.amount=df0_grp.amount.apply(lambda x : round(x))
df0_grp.head()
```

Out[17]:

	txn_description	amount
0	INTER BANK	64331
1	PAY/SALARY	1676577
2	PAYMENT	201794
3	PHONE BANK	10716
4	POS	152861

In [18]:
```
fig=px.treemap(df0_grp,
               path=['txn_description'],
               values='amount',
               color = 'amount',
              )

fig.update_layout(
                    title=dict(text = "Total Amount  by Transaction Desciption",x=0.5,y=0.95),
                    margin=dict(l=10, r=10, t=70, b=10),
                  )
fig.data[0].textinfo = 'label+value'
fig.update_traces(marker_coloraxis=None)
fig.show()
```



Insights :

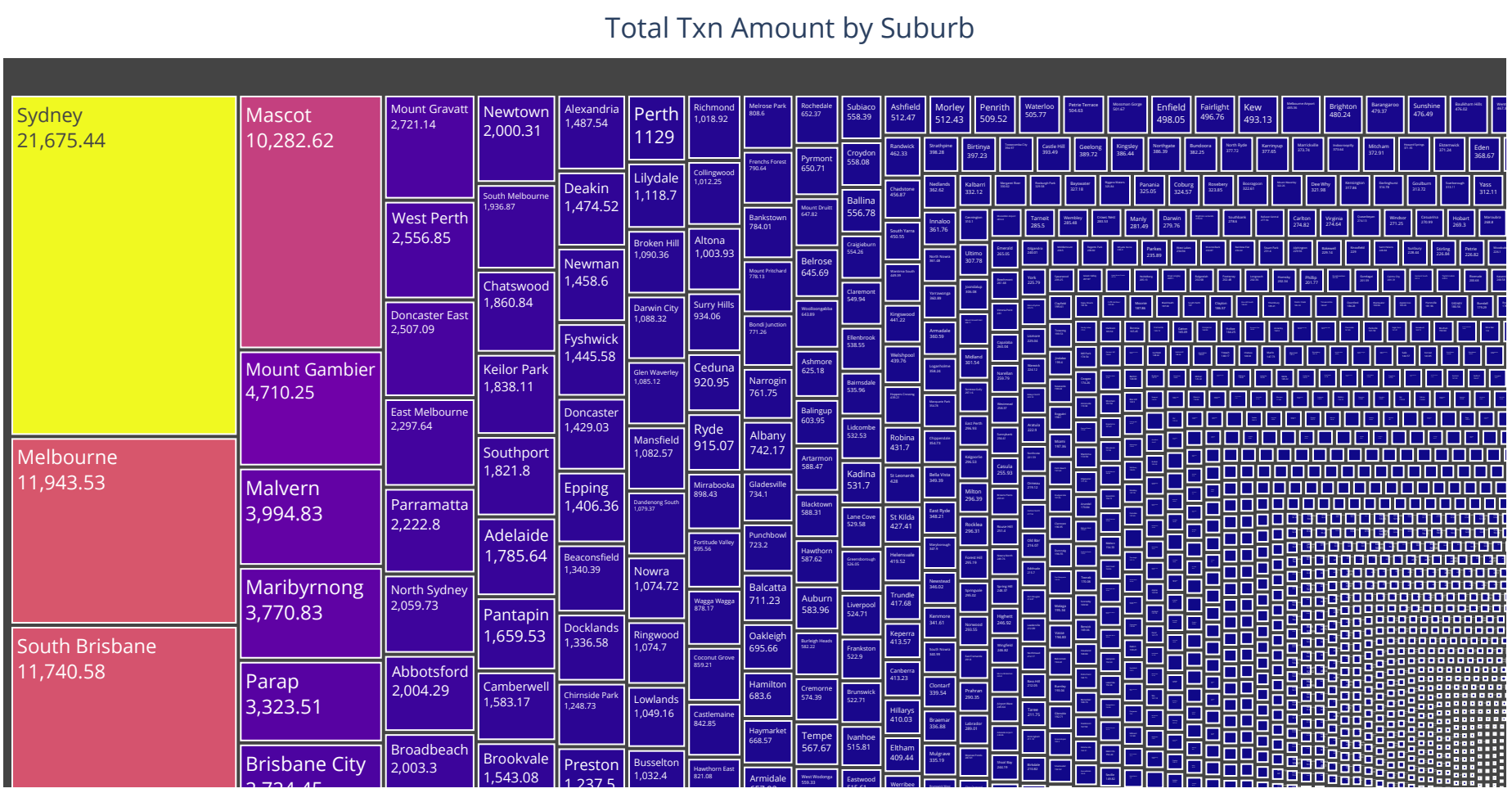
- **Pay/Salary** is the major contributor of **bank txn amount** which is expected as salary transaction amount is usually very high as compared to normal **debit** transactions.

In [19]:
```
df_grp0=df.groupby(by='merchant_suburb').sum()[['amount']].reset_index()
df_grp0.head()
```

Out[19]:

	merchant_suburb	amount
0	Abbotsford	2004.29
1	Aberdeen	52.45
2	Aberfeldie	57.77
3	Aberfoyle Park	84.92
4	Acacia Ridge	10.30

In [20]:
```
fig=px.treemap(df_grp0,
           path=['merchant_suburb'],
           values='amount',
           color = 'amount',
          )
fig.update_layout(
                  title=dict(text = "Total Txn Amount by Suburb",x=0.5,y=0.95),
                  margin=dict(l=10, r=10, t=50, b=10),
                  showlegend=False,
                 )
fig.data[0].textinfo = 'label+value'
fig.update_traces(marker_coloraxis=None)
fig.show()
```



Insights:

- **Sydney , Melbourne, South Brisbane , Mascot and Mount Gambier** are leading contributers of transaction amount.

Analyzing Debit Transactions

In [21]:
```
df1 = df[df.movement=='debit'] # Debit Transactions
df1.head()
```

Out[21]:

	status	card_present_flag	account	long_lat	txn_description	merchant_id	first_name	balance	date	gender	...	extraction	amount
0	authorized	1	ACC-1598451071	153.41 -27.95	POS	81c48296-73be-44a7-befa-d053f48ce7cd	Diana	35.39	2018-08-01	F	...	2018-08-01 01:01:15+00:00	16.25
1	authorized	0	ACC-1598451071	153.41 -27.95	SALES-POS	830a451c-316e-4a6a-bf25-e37caedca49e	Diana	21.20	2018-08-01	F	...	2018-08-01 01:13:45+00:00	14.19
2	authorized	1	ACC-1222300524	151.23 -33.94	POS	835c231d-8cdf-4e96-859d-e9d571760cf0	Michael	5.71	2018-08-01	M	...	2018-08-01 01:26:15+00:00	6.42
3	authorized	1	ACC-1037050564	153.10 -27.66	SALES-POS	48514682-c78a-4a88-b0da-2d6302e64673	Rhonda	2117.22	2018-08-01	F	...	2018-08-01 01:38:45+00:00	40.90
4	authorized	1	ACC-1598451071	153.41 -27.95	SALES-POS	b4e02c10-0852-4273-b8fd-7b3395e32eb0	Diana	17.95	2018-08-01	F	...	2018-08-01 01:51:15+00:00	3.25

5 rows × 23 columns

In [22]:
```
cols = ['card_present_flag', 'status', 'txn_description' , 'movement' , 'gender', 'merchant_state']
#Subplot initialization
fig = make_subplots(
                    rows=3,
                    cols=2,
                    subplot_titles=('card_present_flag', 'status', 'txn_description' , 'movement','gender', 'merchant_st
                    horizontal_spacing=0.2,
                    vertical_spacing=0.2
                   )
# Adding subplots
count=0
for i in range(1,4):
    for j in range(1,3):
        fig.add_trace(go.Bar(x=df1.groupby(by=cols[count]).sum()['amount'].index,
                             y=df1.groupby(by=cols[count]).sum()['amount'].values.round(2),
                             name=cols[count],
                             textposition='auto',
                             text=[str(round((i/sum(df1.groupby(by=cols[count]).sum()['amount'].values))*100,2))+'%'
                                   for i in df1.groupby(by=cols[count]).sum()['amount'].values]
                            ),
                      row=i,col=j)
        fig.update_xaxes(showgrid=False)
        fig.update_yaxes(showgrid=False)
        count+=1
fig.update_layout(
                    title=dict(text = "Analyze Categorical variables (Total Txn Amount/Percentage)",x=0.5,y=0.95),
                    title_font_size=20,
                    showlegend=False,
                    width = 980,
                    height = 980,
                    margin=dict(l=80, r=80, t=150, b=80)
                   )
fig.show()
```

Insights

- Around **80%** amount transacted via cards.
- **Payment** mode of transaction contributes most to the **txn amount**.
- **NSW & VIC** merchant states contributed more than half to overall transaction amount

Data preparation for grouped bar chart to display state & gender wise total transaction amount

In [23]:
```
df_grp=df1.groupby(by=['merchant_state','gender']).sum()[['amount']].reset_index()
df_grp # This is not sorted yet
```

Out[23]:

	merchant_state	gender	amount
0	ACT	F	1657.44
1	ACT	M	3219.24
2	NSW	F	41430.88
3	NSW	M	60590.89
4	NT	F	8741.42
5	NT	M	427.47
6	QLD	F	28611.05
7	QLD	M	24872.40
8	SA	F	11349.73
9	SA	M	5426.84
10	TAS	F	622.72
11	TAS	M	1340.21
12	VIC	F	38626.01
13	VIC	M	48957.99
14	WA	F	19908.15
15	WA	M	14083.91

In [24]:
```
# Sort the dataframe by txn amount
df1.groupby(by=['merchant_state']).sum()[['amount']].sort_values(by='amount',ascending=False).index.values
```

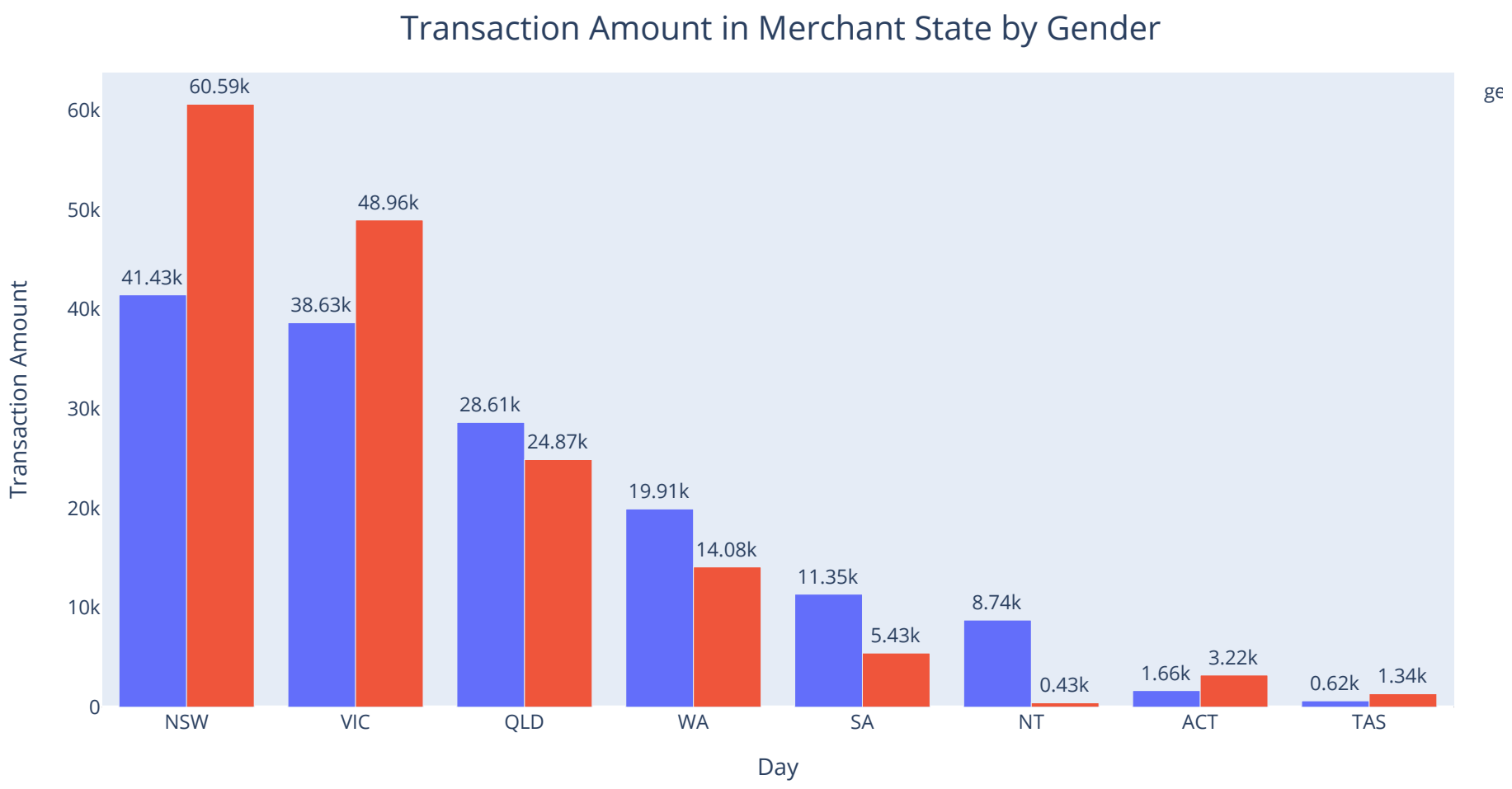
Out[24]:
```
array(['NSW', 'VIC', 'QLD', 'WA', 'SA', 'NT', 'ACT', 'TAS'], dtype=object)
```

In [25]:
```
# Perform sorting using custom order
order = df1.groupby(by=['merchant_state']).sum()[['amount']].sort_values(by='amount',ascending=False).index
df_grp['merchant_state']=pd.Categorical(df_grp['merchant_state'],order)
df_grp= df_grp.groupby(by=['merchant_state','gender']).sum().reset_index()
df_grp
```

Out[25]:

	merchant_state	gender	amount
0	NSW	F	41430.88
1	NSW	M	60590.89
2	VIC	F	38626.01
3	VIC	M	48957.99
4	QLD	F	28611.05
5	QLD	M	24872.40
6	WA	F	19908.15
7	WA	M	14083.91
8	SA	F	11349.73
9	SA	M	5426.84
10	NT	F	8741.42
11	NT	M	427.47
12	ACT	F	1657.44
13	ACT	M	3219.24
14	TAS	F	622.72
15	TAS	M	1340.21

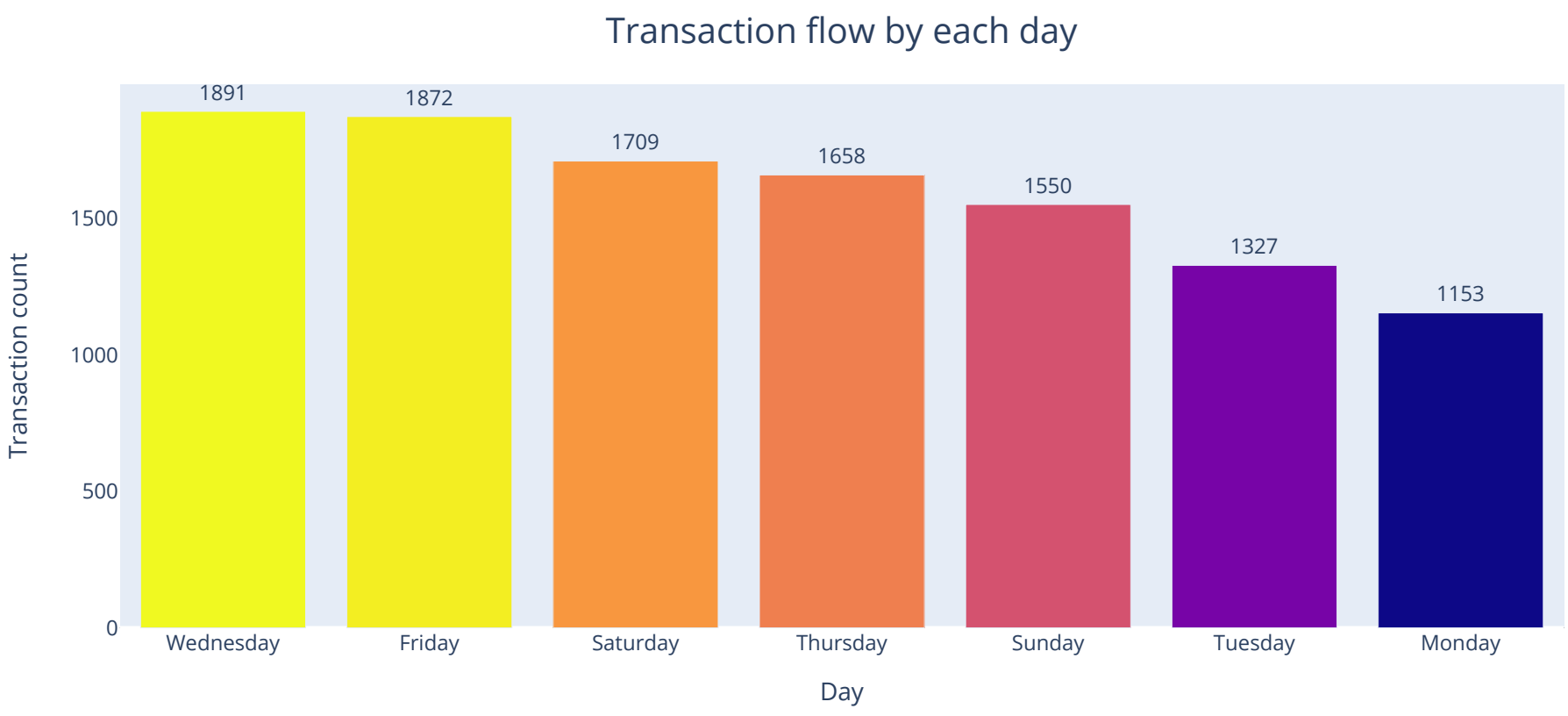
In [26]:
```
fig=px.bar(data_frame=df_grp,
       x='merchant_state',
       y='amount',color='gender',
       barmode='group',
       text=df_grp.amount.apply(lambda x : str(round(x/1000,2))+'k')
      )
fig.update_traces(textposition='outside')
fig.update_xaxes(title='Day',showgrid=False)
fig.update_yaxes(title='Transaction Amount',showgrid=False)
fig.update_layout(
                    title=dict(text = "Transaction Amount in Merchant State by Gender",x=0.5,y=0.95),
                    title_font_size=20,
                  )
fig.show()
```

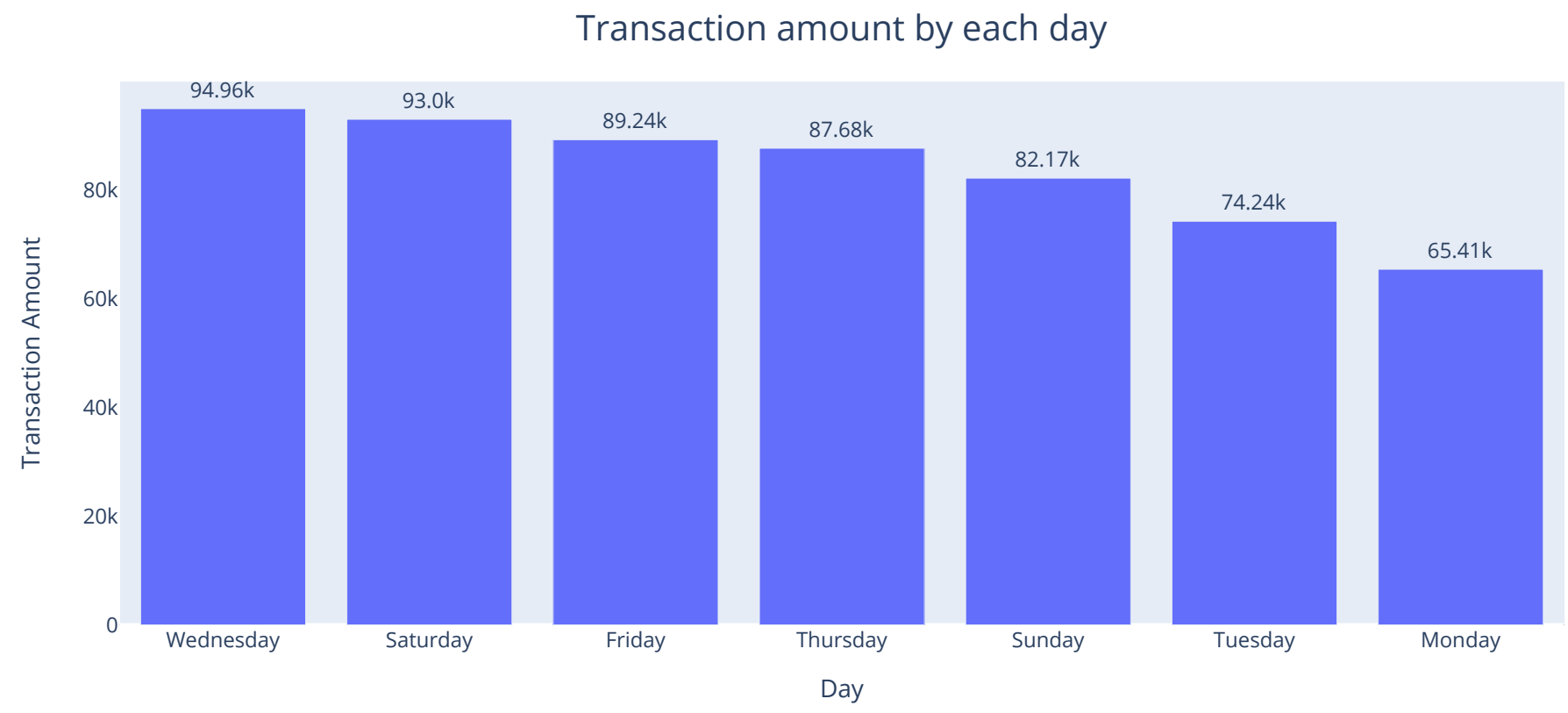


Insights : Overall males carry out more transactions as compared to females but in three states **(QLD,WA,SA)** females are leading.

In [27]:
```
fig= px.bar(data_frame=df,
       x=df1['day'].value_counts().index.tolist(),
       y=df1['day'].value_counts().tolist(),
       color=df1['day'].value_counts().tolist(),
       text=df1['day'].value_counts().tolist()
      )
fig.update_traces(textposition='outside',marker_coloraxis=None)
fig.update_xaxes(title='Day',showgrid=False)
fig.update_yaxes(title='Transaction count',showgrid=False)
fig.update_layout(
                    title=dict(text = "Transaction flow by each day",x=0.5,y=0.95),
                    title_font_size=20,
                    showlegend=False,
                    height = 450,
                  )
fig.show()


fig1= px.bar(data_frame=df1.groupby(by='day').sum()[['amount']].sort_values('amount',ascending=False),
            text=df1.groupby(by='day').sum()[['amount']].sort_values('amount',ascending=False)['amount'].apply(lambda x :
      )
fig1.update_traces(textposition='outside')
fig1.update_xaxes(title='Day',showgrid=False)
fig1.update_yaxes(title='Transaction Amount',showgrid=False)
fig1.update_layout(
                    title=dict(text = "Transaction amount by each day",x=0.5,y=0.95),
                    title_font_size=20,
                    showlegend=False,
                    height = 450,
                  )
fig1.show()
```





Insights

- The transaction count is lower during the start of the week but start to pick up on wednesday through saturday.
- Even though transaction count is comparatively less on **satuday** but it is still at **place 2** in terms of transaction amount which signifies bigger transactions on **Saturday**.

Data preparation for grouped bar chart to display day & gender wise total transaction amount

In [28]:
```
df1_grp=df1.groupby(by=['day','gender']).sum()[['amount']].reset_index()
df1_grp # This is not sorted yet
```

Out[28]:

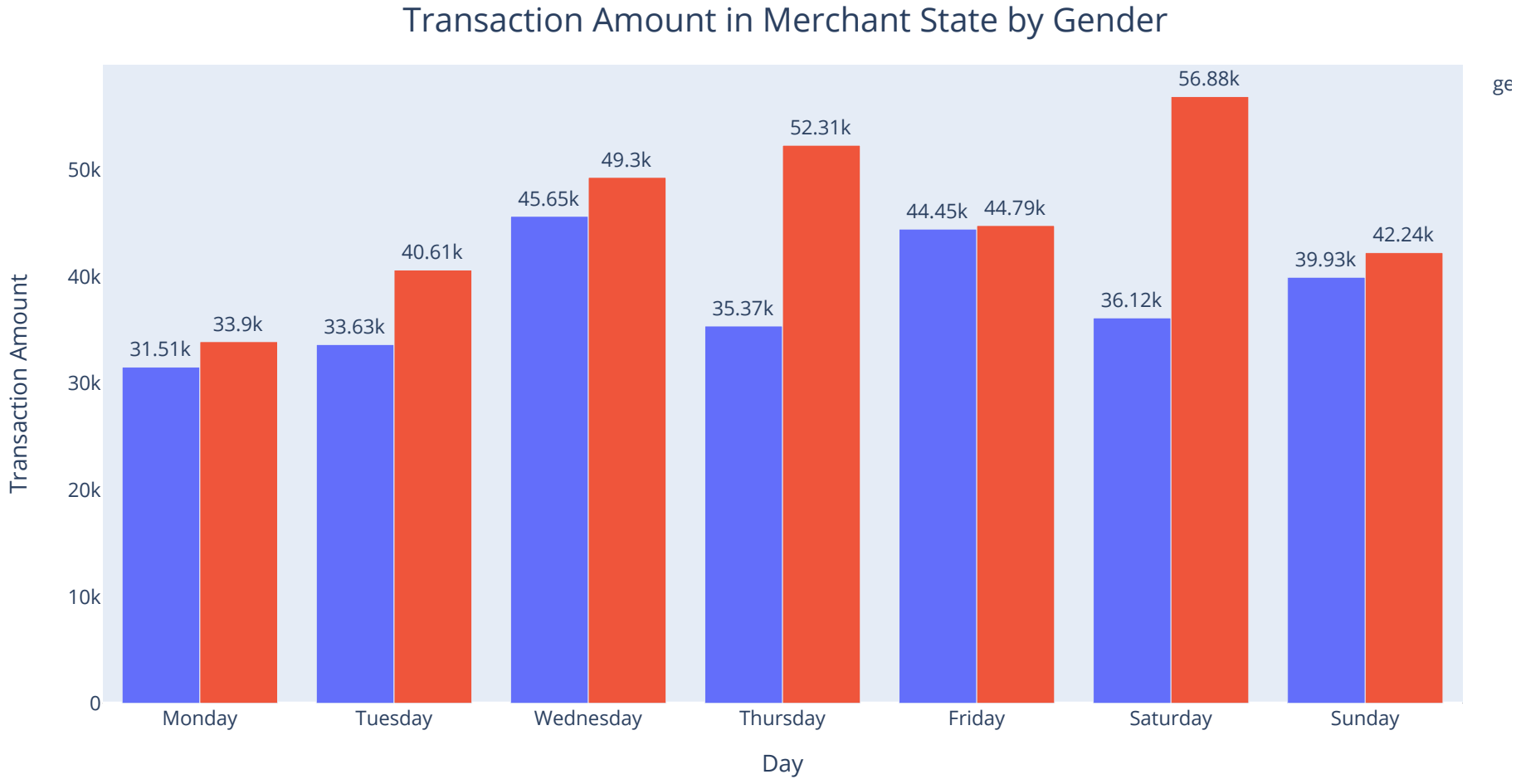
	day	gender	amount
0	Friday	F	44450.06
1	Friday	M	44789.60
2	Monday	F	31511.01
3	Monday	M	33901.90
4	Saturday	F	36124.99
5	Saturday	M	56877.57
6	Sunday	F	39932.72
7	Sunday	M	42241.84
8	Thursday	F	35366.77
9	Thursday	M	52310.59
10	Tuesday	F	33626.24
11	Tuesday	M	40614.62
12	Wednesday	F	45654.61
13	Wednesday	M	49304.83

In [29]:
```
order = ['Monday','Tuesday', 'Wednesday','Thursday','Friday','Saturday','Sunday']
df1_grp['day']=pd.Categorical(df1_grp['day'],order)
df1_grp= df1_grp.groupby(by=['day','gender']).sum().reset_index()
df1_grp
```

Out[29]:

	day	gender	amount
0	Monday	F	31511.01
1	Monday	M	33901.90
2	Tuesday	F	33626.24
3	Tuesday	M	40614.62
4	Wednesday	F	45654.61
5	Wednesday	M	49304.83
6	Thursday	F	35366.77
7	Thursday	M	52310.59
8	Friday	F	44450.06
9	Friday	M	44789.60
10	Saturday	F	36124.99
11	Saturday	M	56877.57
12	Sunday	F	39932.72
13	Sunday	M	42241.84

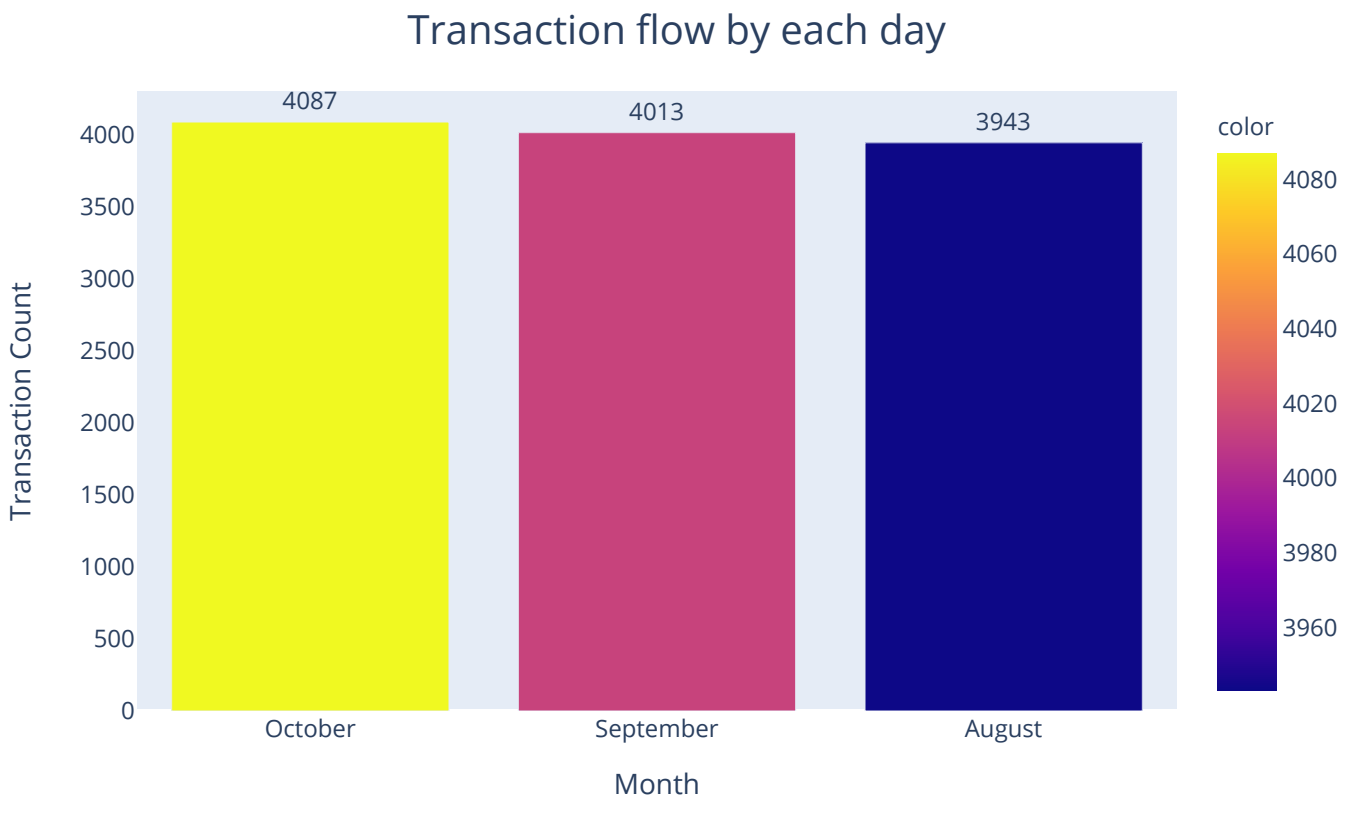
In [30]:
```
fig=px.bar(data_frame=df1_grp,
       x='day',
       y='amount',color='gender',
       barmode='group',
       text=df1_grp.amount.apply(lambda x : str(round(x/1000,2))+'k')
      )
fig.update_traces(textposition='outside')
fig.update_xaxes(title='Day',showgrid=False)
fig.update_yaxes(title='Transaction Amount',showgrid=False)
fig.update_layout(
                    title=dict(text = "Transaction Amount in Merchant State by Gender",x=0.5,y=0.95),
                    title_font_size=20,
                  )
fig.show()
```



Insights

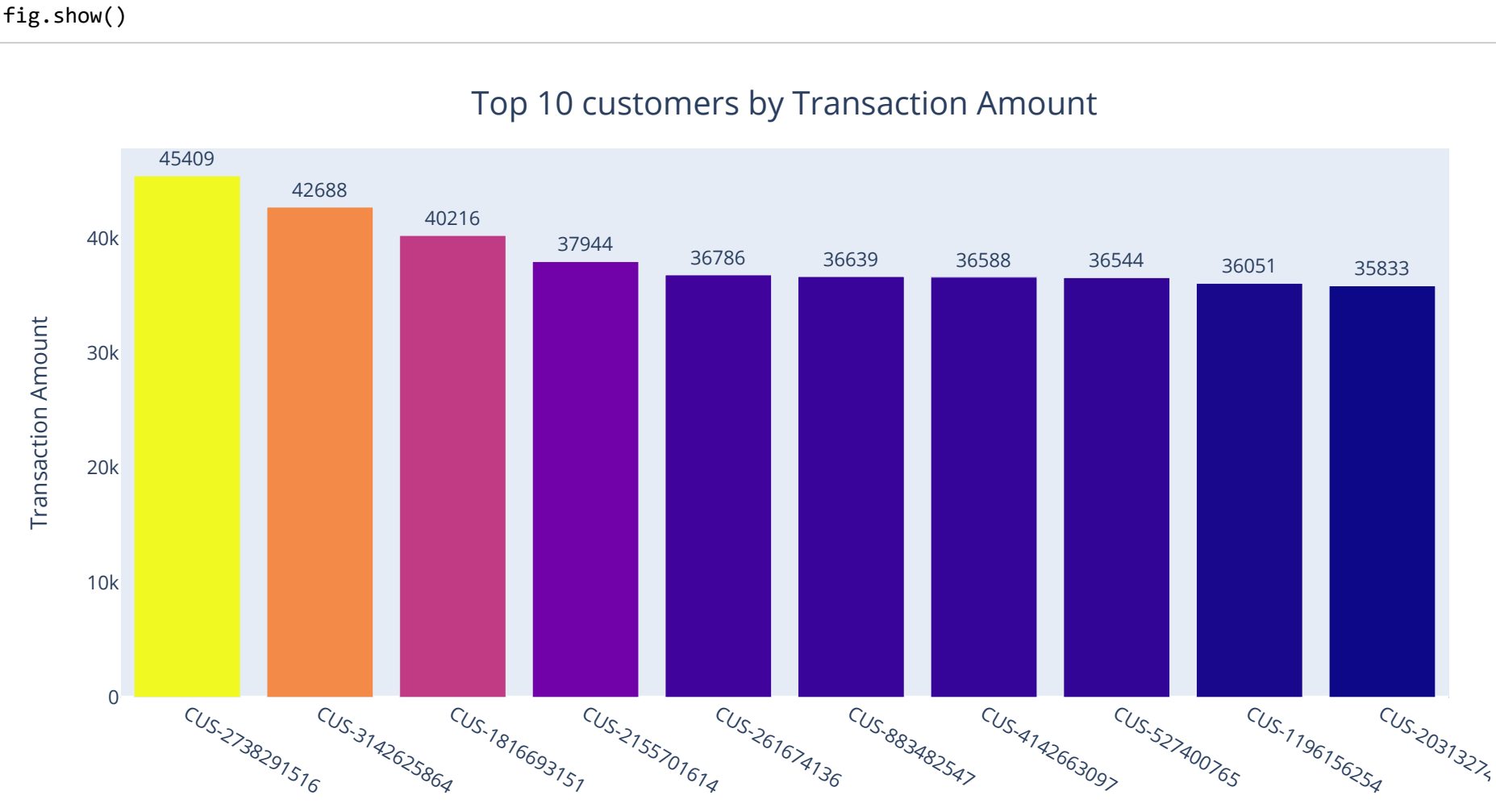
- Males spent most on **Saturday**.
- Females are spending most on **Wednesday & Friday**.

In [31]:
```
fig= px.bar(data_frame=df,
       x=df['month'].value_counts().index.tolist(),
       y=df['month'].value_counts().tolist(),
       color=df['month'].value_counts().tolist(),
       text=df['month'].value_counts().tolist()
      )
fig.update_traces(textposition='outside')
fig.update_xaxes(title='Month',showgrid=False)
fig.update_yaxes(title='Transaction Count',showgrid=False)
fig.update_layout(
                    title=dict(text = "Transaction flow by each day",x=0.5,y=0.95),
                    title_font_size=20,
                    width = 700,
                    height = 450,
                  )
fig.show()
```



- **Insights** : As per the above bar graph there is a steady increase in the number of transaction by each passing Month which is a good sign

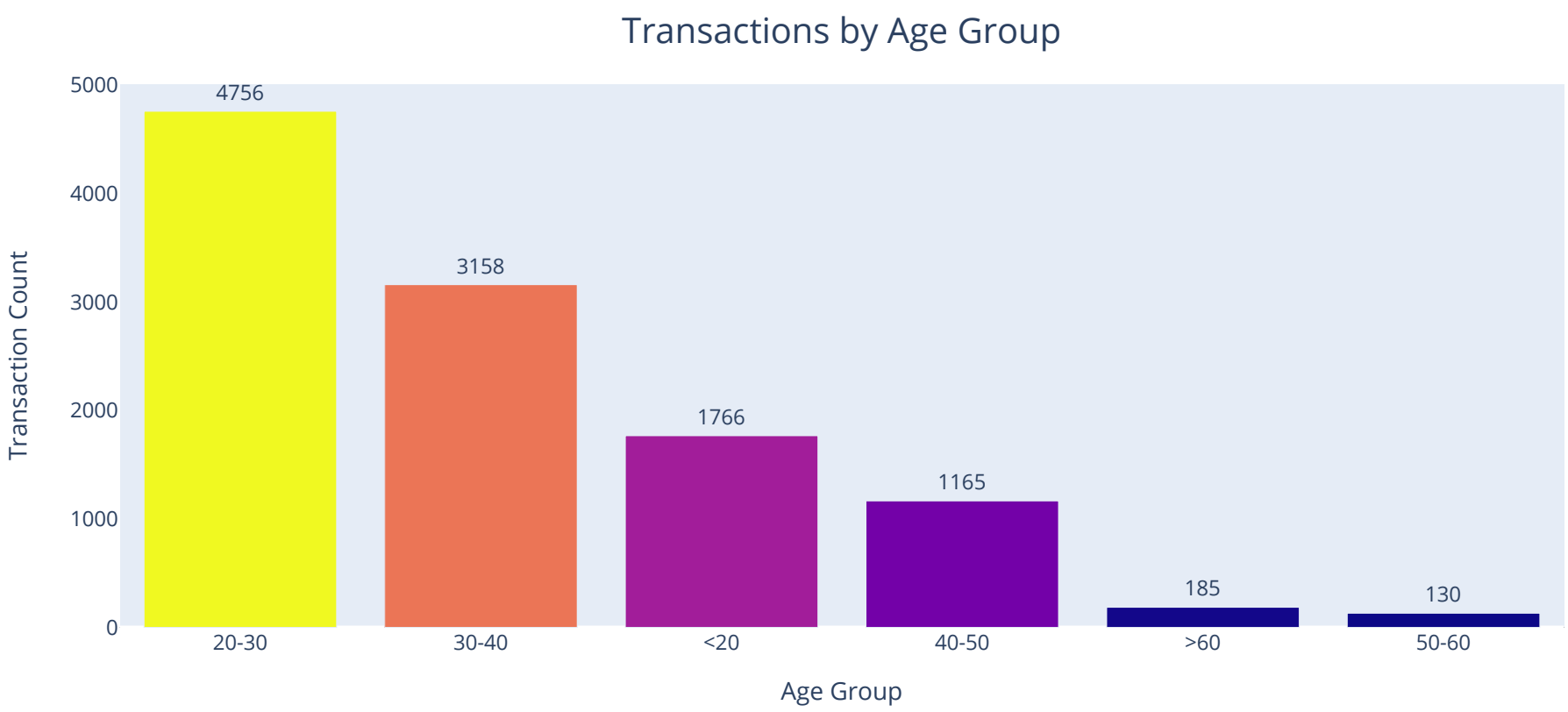
In [32]:
```
fig=px.bar(df.groupby(by='customer_id').sum()['amount'].sort_values(ascending=False).head(10),
       color=df.groupby(by='customer_id').sum()['amount'].sort_values(ascending=False).head(10),
       text=df.groupby(by='customer_id').sum()['amount'].sort_values(ascending=False).head(10).round(),
      )
fig.update_traces(textposition='outside',marker_coloraxis=None)
fig.update_xaxes(title='Customer ID',showgrid=False)
fig.update_yaxes(title='Transaction Amount',showgrid=False)
fig.update_layout(
                    title=dict(text = "Top 10 customers by Transaction Amount",x=0.5,y=0.95),
                    title_font_size=20,
                    showlegend=False,
                    height = 500,
                  )
fig.show()
```



In [33]:
```
df.age_group.value_counts()
```

Out[33]:
```
20-30    5071
30-40    3405
<20      1900
40-50    1293
>60       224
50-60     150
Name: age_group, dtype: int64
```

In [34]:
```
fig=px.bar(df1.age_group.value_counts(),
       color=df1.age_group.value_counts(),
       text=df1.age_group.value_counts().tolist(),
      )
fig.update_traces(textposition='outside',marker_coloraxis=None)
fig.update_xaxes(title='Age Group',showgrid=False)
fig.update_yaxes(title='Transaction Count',showgrid=False)
fig.update_layout(
                    title=dict(text = "Transactions by Age Group",x=0.5,y=0.95),
                    title_font_size=20,
                    showlegend=False,
                    height = 450,
                  )
fig.show()
```



Insights

- Most transactions have been been carried out by Age Groups - **"20-30" & "30-40".**
- Company should think of providing some attractive offers for **"50-60" & ">60"** age groups considering the transaction volume of these groups.

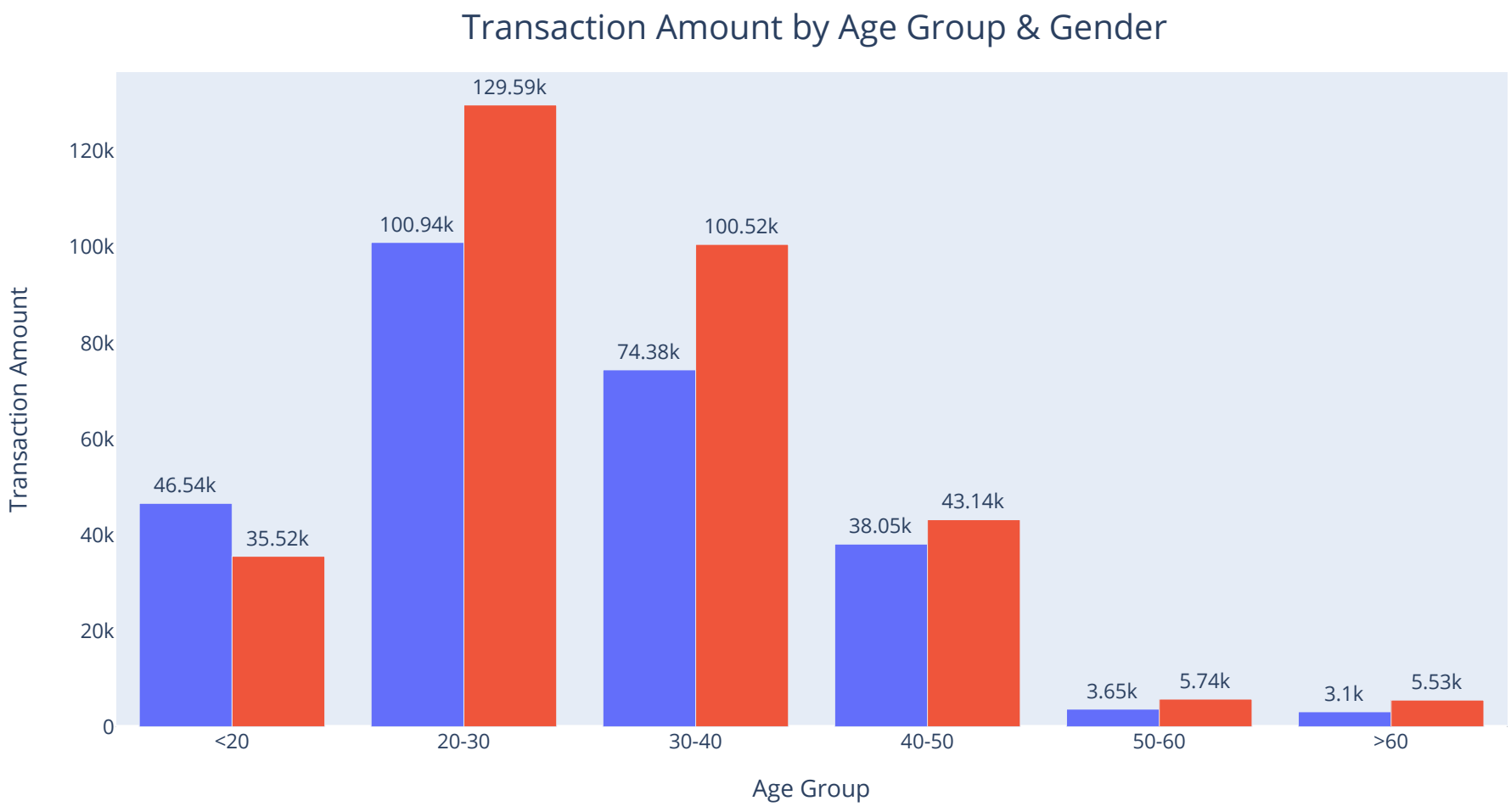
Data preparation for grouped bar chart to display Age Group & gender wise total transaction amount

In [35]:
```
df2_grp=df1.groupby(by=['age_group','gender']).sum()['amount'].reset_index()
df2_grp
```

Out[35]:

	age_group	gender	amount
0	<20	F	46543.04
1	<20	M	35515.86
2	20-30	F	100941.94
3	20-30	M	129592.17
4	30-40	F	74379.11
5	30-40	M	100520.30
6	40-50	F	38050.03
7	40-50	M	43137.16
8	50-60	F	3652.86
9	50-60	M	5742.71
10	>60	F	3099.42
11	>60	M	5532.75

In [36]:
```
fig=px.bar(data_frame=df2_grp,
       x = 'age_group',
       y = 'amount',
       color='gender',
       barmode='group',
       text=df2_grp.amount.apply(lambda x : str(round(x/1000,2))+'k')
      )
fig.update_traces(textposition='outside')
fig.update_xaxes(title='Age Group',showgrid=False)
fig.update_yaxes(title='Transaction Amount',showgrid=False)
fig.update_layout(
                    title=dict(text = "Transaction Amount by Age Group & Gender",x=0.5,y=0.95),
                    title_font_size=20,
                  )
fig.show()
```



Insights :

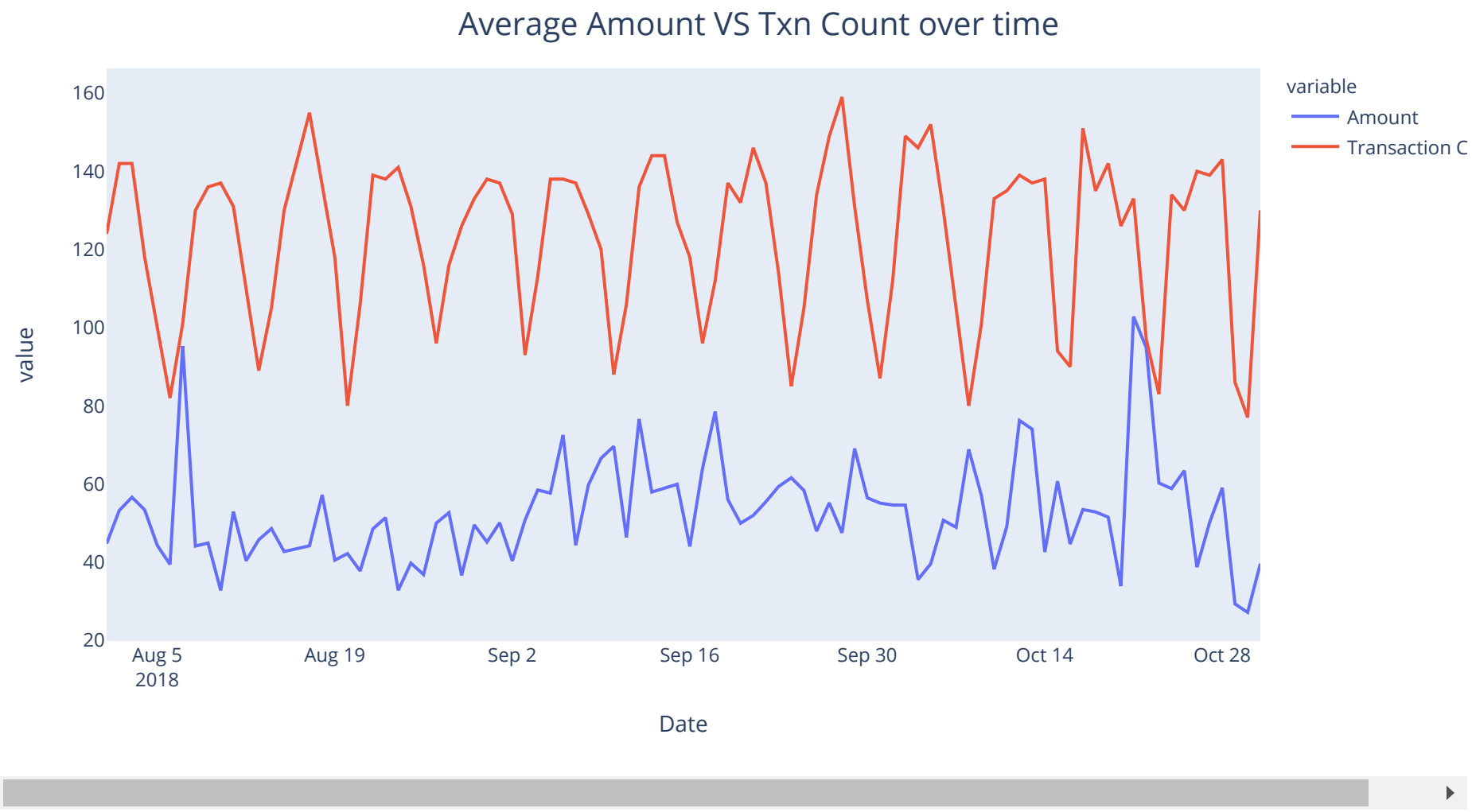
- Males in the age group of **20-30** are contributing most to the Total Txn amount.
- In Age group **'<20'**, Females are ahead of males in terms of Total txn amount

In [37]:
```
df3_grp=df1.groupby(by='date').mean()[['amount']].merge(df1.groupby(by='date').count()[['transaction_id']],on='date')
df3_grp.columns= ['Amount','Transaction Count']
df3_grp.head()
```

Out[37]:

	Amount	Transaction Count
date		
2018-08-01	44.729355	124
2018-08-02	53.225986	142
2018-08-03	56.590845	142
2018-08-04	53.356356	118
2018-08-05	44.265000	100

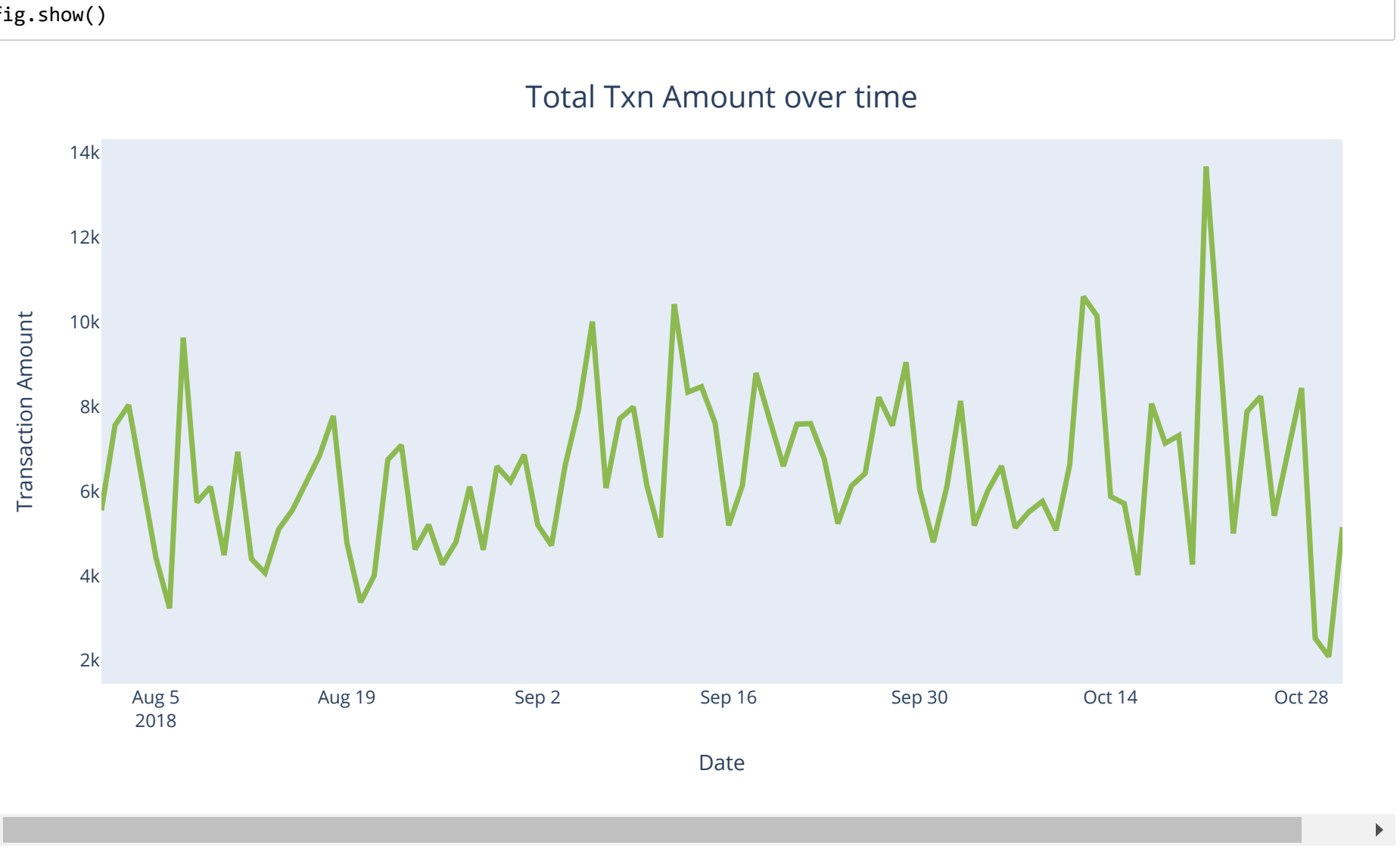
In [38]:
```
fig=px.line(df3_grp)
fig.update_xaxes(title='Date',showgrid=False)
fig.update_yaxes(showgrid=False)
fig.update_layout(
                    title=dict(text = "Average Amount VS Txn Count over time",x=0.5,y=0.95),
                    title_font_size=20,
                    width = 980,
                    height = 500,
                  )
fig.show()
```



Insights

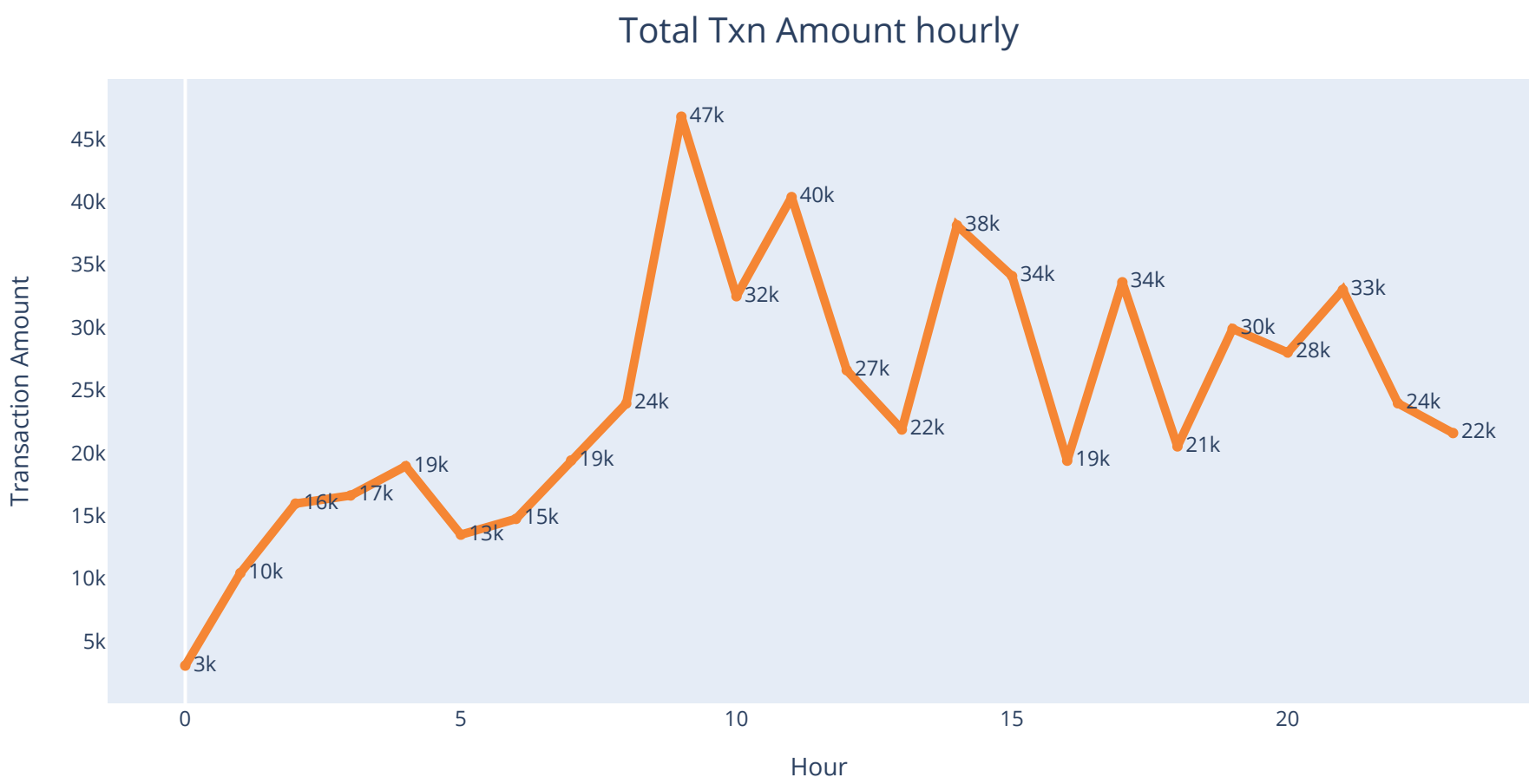
- The average transaction amount on **7th August & Oct 21st** was very high approx **100 AUD**.
- Large number of transactions took place on **17th August & 28th September.**

In [39]:
```
fig=px.line(df1.groupby(by='date').sum()[['amount']])
fig.update_traces(line=dict(color="#8cba51", width=3.5))
fig.update_xaxes(title='Date',showgrid=False)
fig.update_yaxes(title='Transaction Amount',showgrid=False)
fig.update_layout(
                    title=dict(text = "Total Txn Amount over time",x=0.5,y=0.95),
                    title_font_size=20,
                    showlegend=False,
                    width = 980,
                    height = 500
                  )
fig.show()
```



Insights: Total Transaction amount almost touched **14k AUD** on **21st Oct**. Looks like some big transaction were done on that day as the transaction count is not that high on 21st Oct.

In [40]:
```
fig=px.line(df1.groupby(by='hour').sum()[['amount']],
            text=df1.groupby(by='hour').sum()['amount'].apply(lambda x : str(round(x/1000))+'k').values
            )
fig.update_traces(line=dict(color="#f58634", width=5))
fig.update_xaxes(title='Hour',showgrid=False)
fig.update_yaxes(title='Transaction Amount',showgrid=False)
fig.update_layout(
                    title=dict(text = "Total Txn Amount hourly",x=0.5,y=0.95),
                    title_font_size=20,
                    showlegend=False,
                    width = 980,
                    height = 500
                  )
fig.update_traces(textposition='middle right',fillcolor='red')
fig.show()
```



Insights:

- Total transaction amount generated at **9:00 AM** is approx **47k** which is highest throughout the day.
- Between **12:00 AM - 7:00 AM** we have least transaction amount because of off hours.

In [41]:
```
df4_grp= df1.groupby(by=['hour','month','gender']).agg(['count','sum'])[['amount']].reset_index()
df4_grp.columns = ['hour', 'month' ,'gender','Transaction Count', 'Total Txn Amount']
df4_grp
```

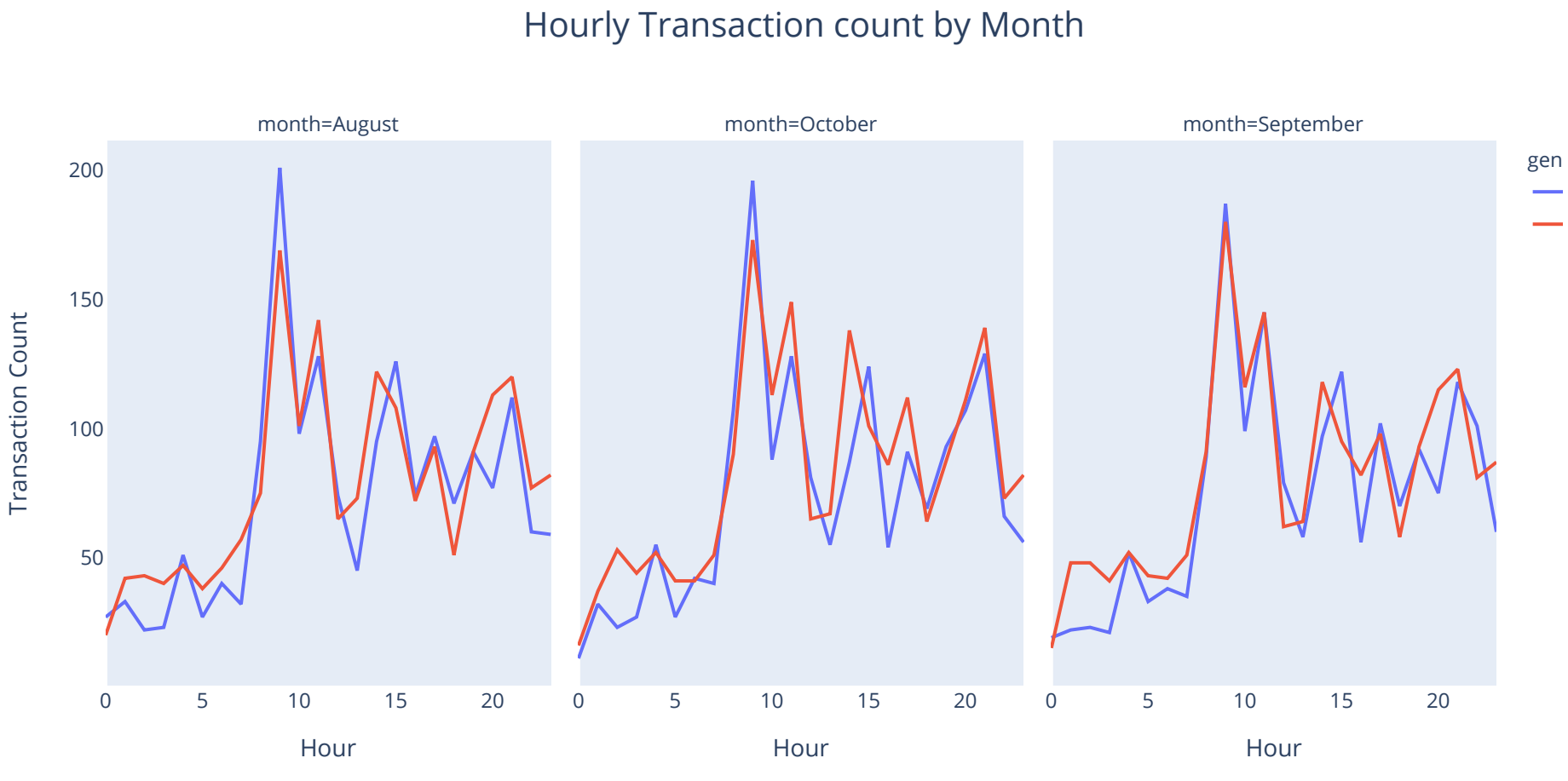
Out[41]:

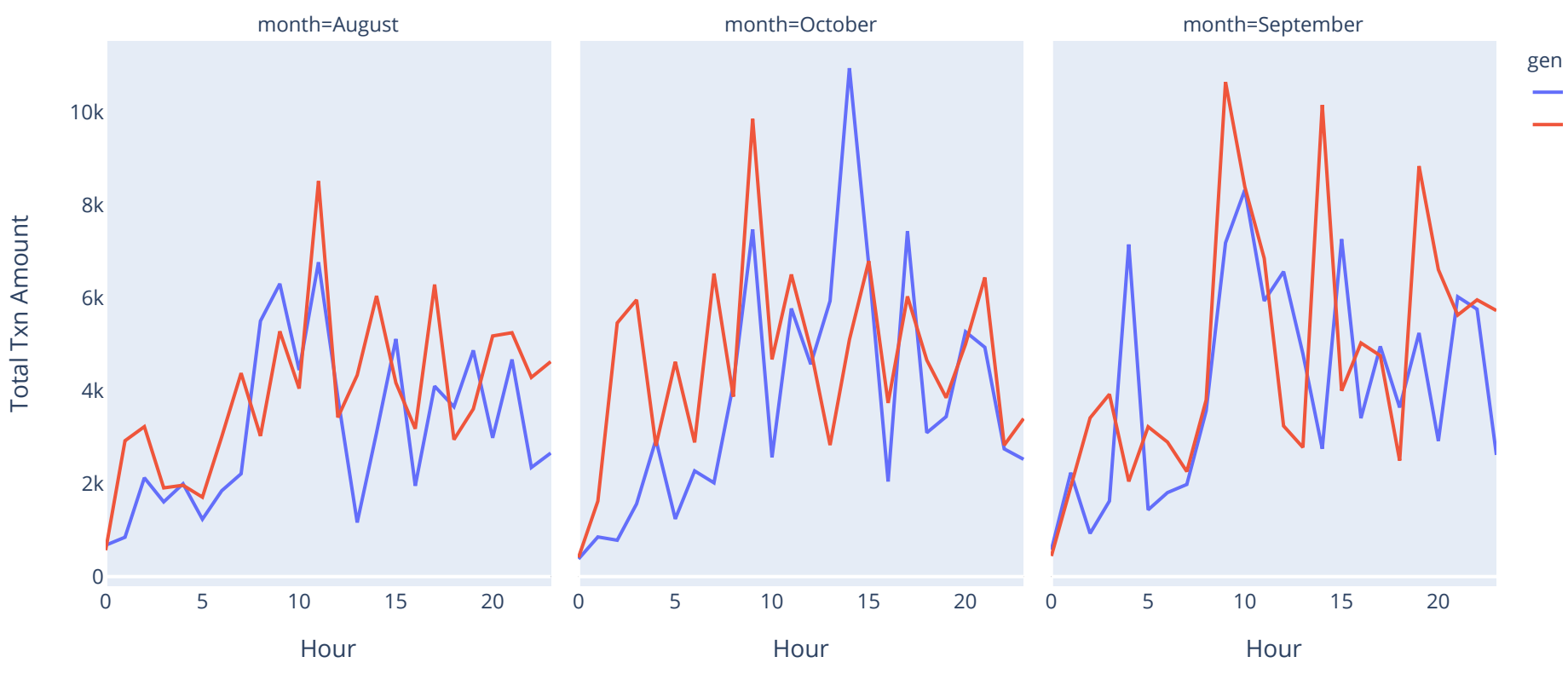
	hour	month	gender	Transaction Count	Total Txn Amount
0	0	August	F	27	676.43
1	0	August	M	20	568.77
2	0	October	F	11	379.63
3	0	October	M	16	411.30
4	0	September	F	19	574.91
...	...	...	...	...	...
139	23	August	M	82	4630.99
140	23	October	F	56	2527.53
141	23	October	M	82	3404.24
142	23	September	F	60	2621.67
143	23	September	M	87	5732.32

144 rows × 5 columns

In [42]:
```
fig1=px.line(data_frame=df4_grp,
             x=df4_grp.hour,
             y=df4_grp['Transaction Count'],
             color=df4_grp.gender,
             facet_col= df4_grp.month
            )
fig1.update_xaxes(title='Hour',showgrid=False)
fig1.update_yaxes(showgrid=False)
fig1.update_layout(
                    title=dict(text = "Hourly Transaction count by Month ",x=0.5,y=0.95),
                    title_font_size=20,
                    width = 980,
                    height = 500,
                    margin=dict(l=80, r=80, t=100, b=80)
                   )
fig1.show()


fig2=px.line(data_frame=df4_grp,
             x=df4_grp.hour,
             y=df4_grp['Total Txn Amount'],
             color=df4_grp.gender,
             facet_col= df4_grp.month
            )
fig2.update_xaxes(title='Hour',showgrid=False)
fig2.update_yaxes(showgrid=False)
fig2.update_layout(
                    title=dict(text = "Hourly Transaction count by Month ",x=0.5,y=0.95),
                    title_font_size=20,
                    width = 980,
                    height = 500,
                    margin=dict(l=80, r=80, t=100, b=80)
                   )
fig2.show()
```





Insights:

- In the month of **September & October** even though transaction count by females are more at **9:00 AM** but **TXN amount** is still less. Seems like comparatively small transactions done by females during the start of the day.
- In **October** at **2:00 PM** transaction amount by **females** is almost double as compared to males.

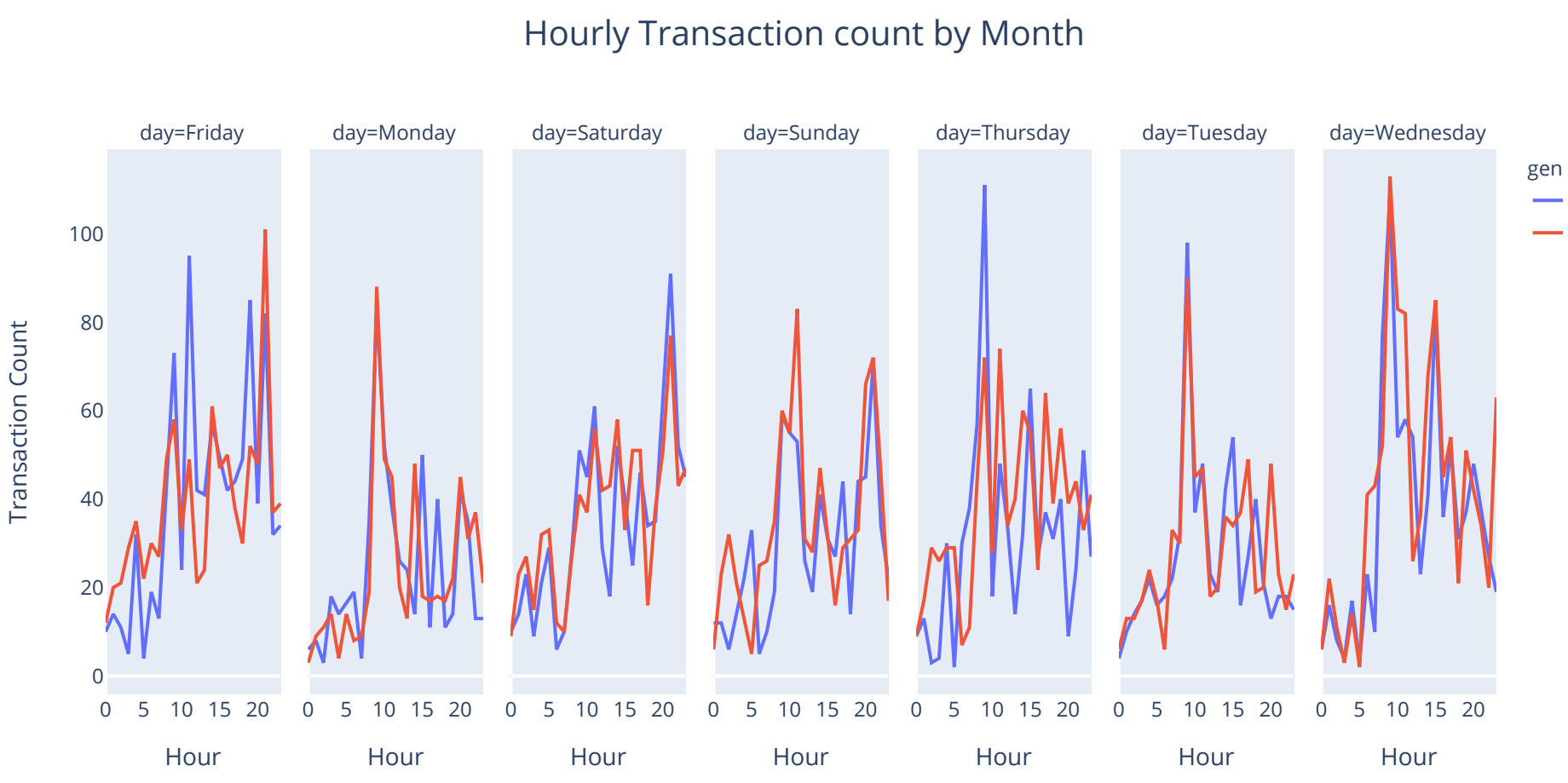
In [43]:
```
df4_grp= df1.groupby(by=['hour','day','gender']).agg(['count','sum'])[['amount']].reset_index()
df4_grp.columns = ['hour', 'day' ,'gender','Transaction Count', 'Total Txn Amount']
df4_grp.head()
```

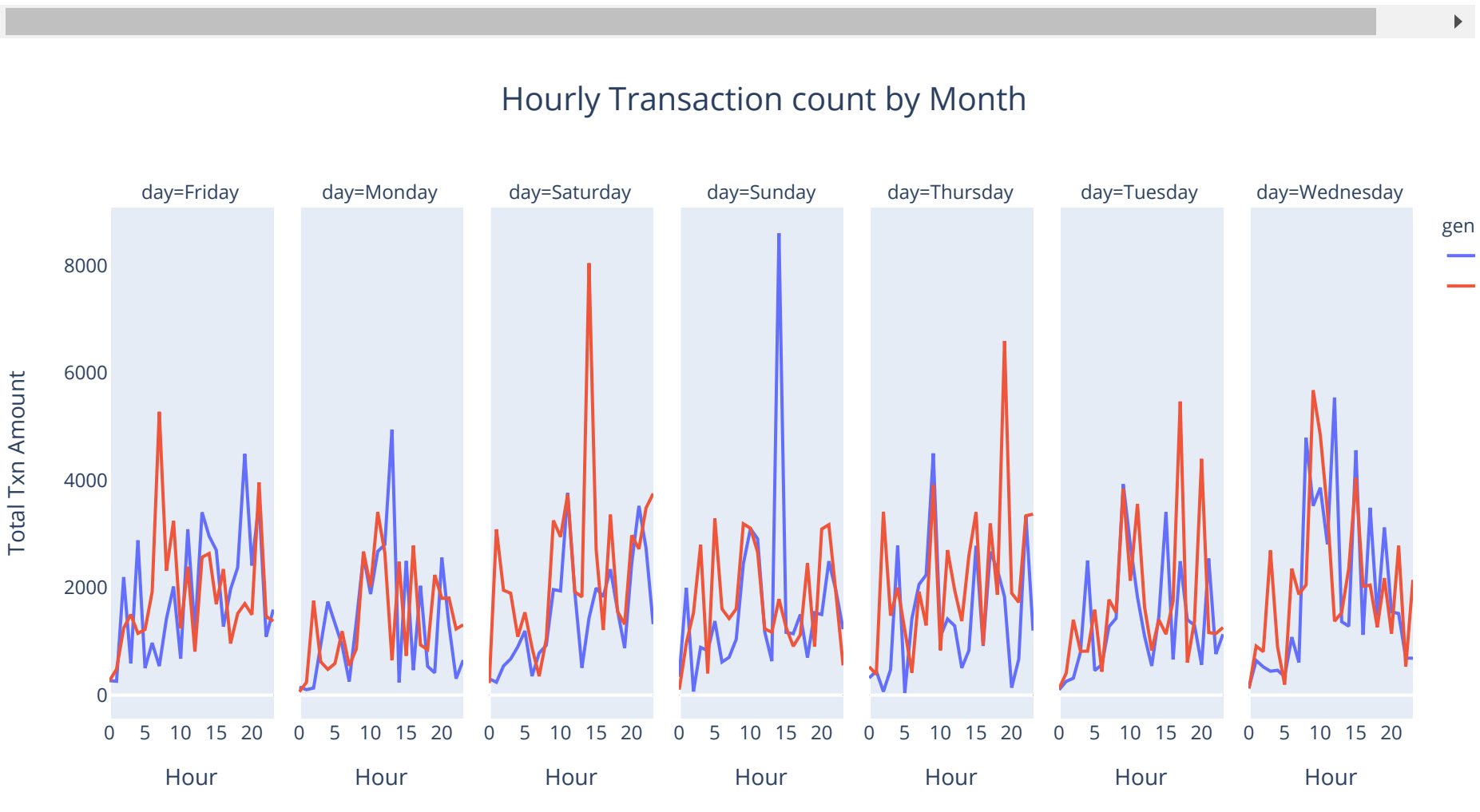
Out[43]:

	hour	day	gender	Transaction Count	Total Txn Amount
0	0	Friday	F	10	268.61
1	0	Friday	M	12	265.03
2	0	Monday	F	6	149.13
3	0	Monday	M	3	61.40
4	0	Saturday	F	10	303.74

In [44]:
```
fig1=px.line(data_frame=df4_grp,
             x=df4_grp.hour,
             y=df4_grp['Transaction Count'],
             color=df4_grp.gender,
             facet_col= df4_grp.day
            )
fig1.update_xaxes(title='Hour',showgrid=False)
fig1.update_yaxes(showgrid=False)
fig1.update_layout(
                    title=dict(text = "Hourly Transaction count by Month ",x=0.5,y=0.95),
                    title_font_size=20,
                    width = 980,
                    height = 500,
                    margin=dict(l=80, r=80, t=100, b=80)
                   )
fig1.show()


fig2=px.line(data_frame=df4_grp,
             x=df4_grp.hour,
             y=df4_grp['Total Txn Amount'],
             color=df4_grp.gender,
             facet_col= df4_grp.day
            )
fig2.update_xaxes(title='Hour',showgrid=False)
fig2.update_yaxes(showgrid=False)
fig2.update_layout(
                    title=dict(text = "Hourly Transaction count by Month ",x=0.5,y=0.95),
                    title_font_size=20,
                    width = 980,
                    height = 500,
                    margin=dict(l=80, r=80, t=100, b=80)
                   )
fig2.show()
```





Insights:

- On **Saturday** at **lunch time (2:00 PM)** transaction amount by **males** is almost **6 times** higher than **females**. However on **Sunday** at the same time the trend is completely in the opposite direction.

Analysing Credit Transactions

In [45]:
```
df2 = df[df.movement=='credit']
df2.head()
```

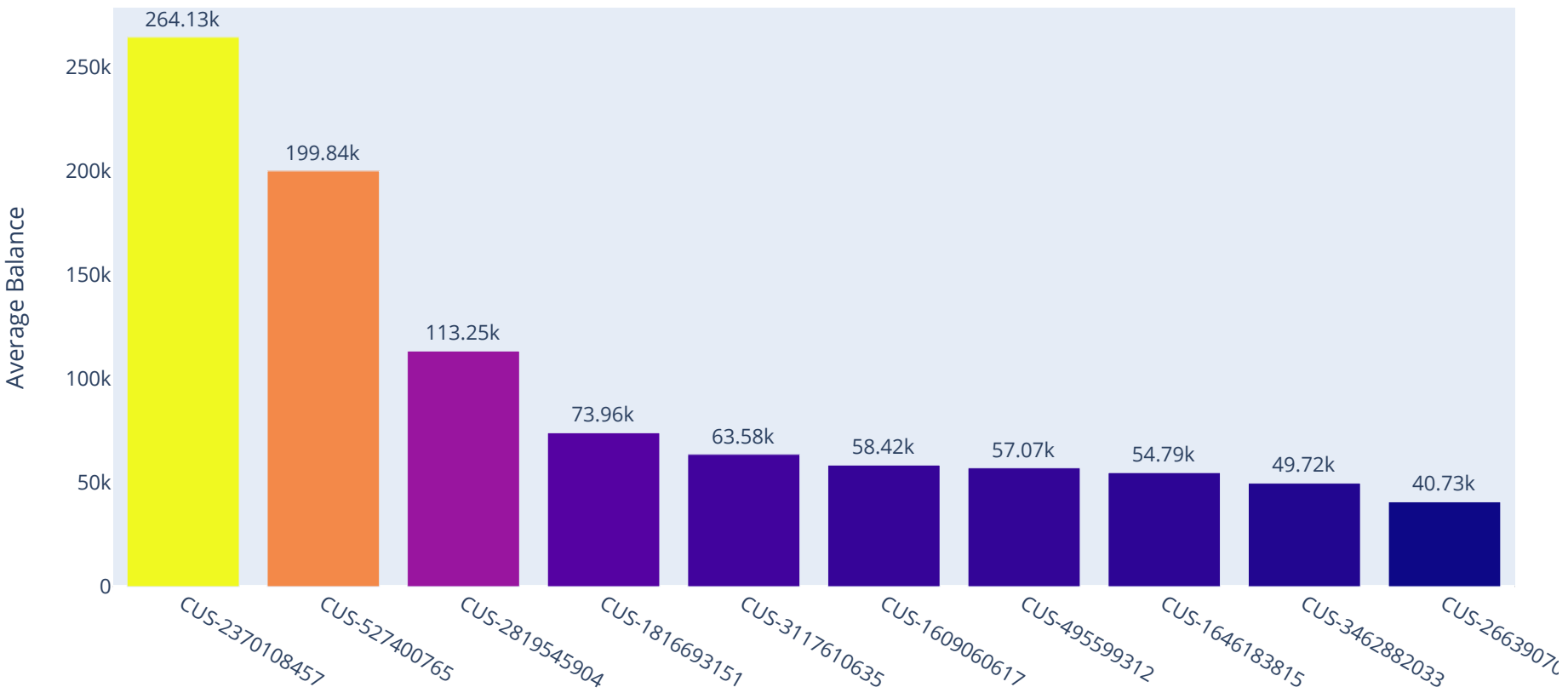
Out[45]:

	status	card_present_flag	account	long_lat	txn_description	merchant_id	first_name	balance	date	gender	...	extraction	amount	
50	posted	<NA>	ACC-588564840	151.27 -33.76	PAY/SALARY	NaN	Isaiah	8342.11	2018-08-01	M	...	2018-08-01 11:00:00+00:00	3903.95	9c
61	posted	<NA>	ACC-1650504218	145.01 -37.93	PAY/SALARY	NaN	Marissa	2040.58	2018-08-01	F	...	2018-08-01 12:00:00+00:00	1626.48	18
64	posted	<NA>	ACC-3326339947	151.18 -33.80	PAY/SALARY	NaN	Eric	3158.51	2018-08-01	M	...	2018-08-01 12:00:00+00:00	983.36	b
68	posted	<NA>	ACC-3541460373	145.00 -37.83	PAY/SALARY	NaN	Jeffrey	2517.66	2018-08-01	M	...	2018-08-01 13:00:00+00:00	1408.08	0d
70	posted	<NA>	ACC-2776252858	144.95 -37.76	PAY/SALARY	NaN	Kristin	2271.79	2018-08-01	F	...	2018-08-01 13:00:00+00:00	1068.04	f

5 rows × 23 columns

In [46]:
```
fig=px.bar(
            df2.groupby(by='customer_id').mean()['balance'].sort_values(ascending=False).head(10),
            text = df2.groupby(by='customer_id').mean()['balance'].sort_values(ascending=False).head(10).apply(
                lambda x : str(round(x/1000,2))+'k' ),
            color = df2.groupby(by='customer_id').mean()['balance'].sort_values(ascending=False).head(10)
          )
fig.update_traces(textposition='outside')
fig.update_xaxes(title='Customer ID',showgrid=False)
fig.update_yaxes(title='Average Balance',showgrid=False)
fig.update_layout(
                    title=dict(text = "Top Valuable Customers by AVG Balance",x=0.5,y=0.95),
                    title_font_size=20,
                    showlegend=False,
                    height = 500

                  )
fig.update_traces(marker_coloraxis=None)
fig.show()
```



In [47]:
```
order = ['August','September','October']
df['month']=pd.Categorical(df['month'],order)
```

In [48]:
```
g1 = df.groupby(by='month').agg(['mean','sum'])['amount']
g1.columns=['Avg Amount', 'Total Amount']
g1[['Avg Amount','Total Amount']]=g1[['Avg Amount','Total Amount']].round().astype(int)
g1.reset_index(inplace=True)
g1
```

Out[48]:

	month	Avg Amount	Total Amount
0	August	185	729936
1	September	182	730550
2	October	196	802798

In [49]:
```
g2=df.groupby(by='month').agg(['mean','sum'])['balance']
g2.columns=['Avg Balance', 'Total Balance']
g2[['Avg Balance','Total Balance']]=g2[['Avg Balance','Total Balance']].round().astype(int)
g2.reset_index(inplace=True)
g2
```

Out[49]:

	month	Avg Balance	Total Balance
0	August	10794	42561328
1	September	14730	59112097
2	October	18451	75409203

In [50]:
```
month = g1.merge(g2,on='month')
month
```

Out[50]:

	month	Avg Amount	Total Amount	Avg Balance	Total Balance
0	August	185	729936	10794	42561328
1	September	182	730550	14730	59112097
2	October	196	802798	18451	75409203

In [51]:
```
pio.templates.default = "plotly_white"
fig = ff.create_table(month)
for i in range(len(fig.layout.annotations)):
    fig.layout.annotations[i].font.size = 13
fig.show()
```

month	Avg Amount	Total Amount	Avg Balance	Total Balance
August	185	729936	10794	42561328
September	182	730550	14730	59112097
October	196	802798	18451	75409203

Insights:

- There is a 7% increase in Avg transaction amount from August to October.
- 71% increase in AVG balance maintained by the customers. Looks like customers have deep trust in the ANZ bank.
- 77% increase in total balance over these 3 months.

End