In [1]:

```
import networkx as nx
```

In [2]:

```
G = nx.Graph()
G.add_edge('A','B', weight= 6, relation = 'family')
G.add_edge('B','C', weight= 13, relation = 'friend')
```

Edge Attributes in NetworkX

In [3]:

```
G.edges()           # list of all edges
```

Out[3]:

```
[('A', 'B'), ('B', 'C')]
```

In [4]:

```
G.edges(data= True) # list of all edges with attributes
```

Out[4]:

```
[('A', 'B', {'relation': 'family', 'weight': 6}),
 ('B', 'C', {'relation': 'friend', 'weight': 13})]
```

In [5]:

```
G.edges(data= 'relation') #list of all edges with attribute ‘relation’
```

Out[5]:

```
[('A', 'B', 'family'), ('B', 'C', 'friend')]
```

Accessing attributes of a specific edge:

In [6]:

```
G.edge['A']['B'] # dictionary of attributes of edge (A, B)
```

Out[6]:

```
{'relation': 'family', 'weight': 6}
```

In [7]:

```
G.edge['B']['C']['weight']
```

Out[7]:

```
13
```

In [8]:

```
G.edge['C']['B']['weight'] # undirected graph, order does not matter
```

Out[8]:

```
13
```

In [9]:

```
G=nx.DiGraph()
G.add_edge('A','B', weight= 6, relation = 'family')
G.add_edge('C', 'B', weight= 13, relation = 'friend')
```

In [10]:

```
G.edge['C']['B']['weight']
```

Out[10]:

```
13
```

In [11]:

```
G.edge['B']['C']['weight'] # directed graph, order matters
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-11-764a9ce3ec40> in <module>()
----> 1 G.edge['B']['C']['weight'] # directed graph, order matters

KeyError: 'C'
```

In [12]:

```
G=nx.MultiGraph()
G.add_edge('A','B', weight= 6, relation = 'family')
G.add_edge('A','B', weight= 18, relation = 'friend')
G.add_edge('C','B', weight= 13, relation = 'friend')
```

In [13]:

```
G.edge['A']['B'] # One dictionary of attributes per (A,B) edge
```

Out[13]:

```
{0: {'relation': 'family', 'weight': 6},
 1: {'relation': 'friend', 'weight': 18}}
```

In [14]:

```
G.edge['A']['B'][0]['weight'] # undirected graph, order does not matter
```

Out[14]:

```
6
```

In [15]:

```
G=nx.MultiDiGraph()
G.add_edge('A','B', weight= 6, relation = 'family')
G.add_edge('A','B', weight= 18, relation = 'friend')
G.add_edge('C','B', weight= 13, relation = 'friend')
```

In [16]:

```
G.edge['A']['B'][0]['weight']
```

Out[16]:

```
6
```

In [17]:

```
G.edge['B']['A'][0]['weight'] # directed graph, order matters
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-17-f0b626672b4c> in <module>()
----> 1 G.edge['B']['A'][0]['weight'] # directed graph, order matters

KeyError: 'A'
```

Node Attributes in NetworkX

In [18]:

```
G=nx.Graph()
G.add_edge('A','B', weight= 6, relation = 'family')
G.add_edge('B','C', weight= 13, relation = 'friend')
```

Adding node attributes:

In [19]:

```
G.add_node('A', role = 'trader')
G.add_node('B', role = 'trader')
G.add_node('C', role = 'manager')
```

Accessing node attributes:

In [20]:

```
G.nodes() # list of all nodes
```

Out[20]:

```
['A', 'B', 'C']
```