**import** pandas **as** pd

*#Concatenating objects***"""  
pd.concat(  
 objs,  
 axis=0,  
 join="outer",  
 ignore\_index=False,  
 keys=None,  
 levels=None,  
 names=None,  
 verify\_integrity=False,  
 copy=True,  
)  
"""**df1 = pd.DataFrame(  
 {  
 **"A"**: [**"A0"**, **"A1"**, **"A2"**, **"A3"**],  
 **"B"**: [**"B0"**, **"B1"**, **"B2"**, **"B3"**],  
 **"C"**: [**"C0"**, **"C1"**, **"C2"**, **"C3"**],  
 **"D"**: [**"D0"**, **"D1"**, **"D2"**, **"D3"**],  
 },  
 index=[0, 1, 2, 3],  
)  
  
  
df2 = pd.DataFrame(  
 {  
 **"A"**: [**"A4"**, **"A5"**, **"A6"**, **"A7"**],  
 **"B"**: [**"B4"**, **"B5"**, **"B6"**, **"B7"**],  
 **"C"**: [**"C4"**, **"C5"**, **"C6"**, **"C7"**],  
 **"D"**: [**"D4"**, **"D5"**, **"D6"**, **"D7"**],  
 },  
 index=[4, 5, 6, 7],  
)  
  
  
df3 = pd.DataFrame(  
 {  
 **"A"**: [**"A8"**, **"A9"**, **"A10"**, **"A11"**],  
 **"B"**: [**"B8"**, **"B9"**, **"B10"**, **"B11"**],  
 **"C"**: [**"C8"**, **"C9"**, **"C10"**, **"C11"**],  
 **"D"**: [**"D8"**, **"D9"**, **"D10"**, **"D11"**],  
 },  
 index=[8, 9, 10, 11],  
)  
  
  
frames = [df1, df2, df3]  
  
*# result = pd.concat(frames)  
# print(result)  
  
# result = pd.concat(frames, keys=["x", "y", "z"])  
# print(result)  
#  
# print(result.loc["y"])  
  
# join = inner/outer example*df4 = pd.DataFrame(  
 {  
 **"B"**: [**"B2"**, **"B3"**, **"B6"**, **"B7"**],  
 **"D"**: [**"D2"**, **"D3"**, **"D6"**, **"D7"**],  
 **"F"**: [**"F2"**, **"F3"**, **"F6"**, **"F7"**],  
 },  
 index=[2, 3, 6, 7],  
)  
print(df1)  
print()  
print(df4)  
print()  
  
result = pd.concat([df1, df4], axis=1)  
print(result)  
print()  
  
result = pd.concat([df1, df4], axis=1, join=**"inner"**)  
print(result)  
print()  
  
result = pd.concat([df1, df4], axis=1).reindex(df1.index)  
print(result)  
print()  
  
result = pd.concat([df1, df4], ignore\_index=**True**, sort=**False**)  
print(result)  
print()  
  
*#Appending row to a dataframe*s2 = pd.Series([**"X0"**, **"X1"**, **"X2"**, **"X3"**], index=[**"A"**, **"B"**, **"C"**, **"D"**])  
print(s2)  
print()  
result = pd.concat([df1, s2.to\_frame().T], ignore\_index=**True**)  
print(result)  
print()  
  
  
*# Merge***"""  
pd.merge(  
 left,  
 right,  
 how="inner",  
 on=None,  
 left\_on=None,  
 right\_on=None,  
 left\_index=False,  
 right\_index=False,  
 sort=True,  
 suffixes=("\_x", "\_y"),  
 copy=True,  
 indicator=False,  
 validate=None,  
)  
  
Pandas merge(): Combining Data on Common Columns or Indices  
Used to combine data objects based on one or more keys in a similar way   
to a relational database  
  
Some important parameters  
how: This defines what kind of merge to make.  
 It defaults to 'inner', other possible options:'outer', 'left', ,'right'  
  
on :it isn’t specified, columns from the two DataFrames   
that share names will be used as join keys.  
  
left\_on and right\_on : Use either of these to specify a column or index that is present   
only in the left or right objects that you are merging. Both default to None.  
  
left\_index and right\_index: Set these to True to use the index of the left or right objects   
to be merged. Both default to False.  
  
suffixes: This is a tuple of strings  
"""**df1 = pd.DataFrame({**'user\_id'**: [**'id001'**, **'id002'**, **'id003'**, **'id004'**, **'id005'**, **'id006'**, **'id007'**],  
 **'first\_name'**: [**'Rivi'**, **'Wynnie'**, **'Kristos'**, **'Madalyn'**, **'Tobe'**, **'Regan'**, **'Kristin'**],  
 **'last\_name'**: [**'Valti'**, **'McMurty'**, **'Ivanets'**, **'Max'**, **'Riddich'**, **'Huyghe'**, **'Illis'**],  
 **'email'**: [**'rvalti0@example.com'**, **'wmcmurty1@example.com'**, **'kivanets2@example.com'**,  
 **'mmax3@example.com'**, **'triddich4@example.com'**, **'rhuyghe@example.com'**, **'killis4@example.com'**]  
 })  
print(df1)  
print()  
  
df2 = pd.DataFrame({**'user\_id'**: [**'id001'**, **'id002'**, **'id003'**, **'id004'**, **'id005'**,**'id008'**],  
 **'image\_url'**: [**'http://img/id001.png'**, **'http://img/id002.jpg'**,  
 **'http://img/id003.bmp'**, **'http://img/id004.jpg'**,  
 **'http://img/id005.png'**,**'http://img/id008.png'**]  
 })  
print(df2)  
print()  
  
df3\_merged = pd.merge(df1, df2)  
print(df3\_merged)  
print()  
  
**"""  
Since df1 and df2 have the column user\_id with the same name,   
the merge() function automatically joins on that key.   
If we had two columns with different names,   
we could use left\_on= and right\_on= to specify keys on both DataFrames explicitly.  
"""**df3\_merged = pd.merge(df1, df2,how=**'inner'**)  
print(df3\_merged)  
print()  
  
df\_left\_merge = pd.merge(df1, df2, how=**'left'**)  
print(df\_left\_merge)  
print()  
  
df\_right\_merge = pd.merge(df1, df2, how=**'right'**)  
print(df\_right\_merge)  
print()  
  
df\_outer = pd.merge(df2, df1, how=**'outer'**, indicator=**True**)  
print(df\_outer)  
print()  
  
  
  
*# ## Another example*sales=pd.read\_csv(**"sales.csv"**) *# reading from csv file*print(sales)  
  
product=pd.read\_csv(**"c:\\pandas\\products.csv"**) *# reading from csv file*print(product)  
  
customer=pd.read\_csv(**"c:\\pandas\\customer.csv"**) *# reading from csv file*print(customer)  
  
  
*# ### List of quantity sold against each products*print(sales.groupby([**'product'**,**'p\_id'**])[**'qty'**].sum())  
  
print(type(sales.groupby([**'product'**,**'p\_id'**])[**'qty'**]))  
  
print(type(sales.groupby([**'product'**,**'p\_id'**])[**'qty'**].sum()))  
  
print(type(sales.groupby([**'product'**,**'p\_id'**])[[**'qty'**]].sum()))  
  
  
print(sales.groupby([**'product'**,**'p\_id'**])[[**'qty'**]].sum())  
  
  
*# List of quantity and total sales against each product  
# using groupby get the list of products and its sum sold*my\_sale=sales.groupby([**'product'**,**'p\_id'**])[[**'qty'**]].sum()  
print(my\_sale)  
print()  
  
my\_sale.reset\_index(inplace=**True**)  
print(my\_sale)  
print()  
  
  
my\_sum=pd.merge(my\_sale,product,on=**'p\_id'**)  
print(my\_sum)  
print()  
  
  
my\_sum=pd.merge(my\_sale,product)  
print(my\_sum)  
print()  
  
*# Joining my\_sale and product to get the price of each product*my\_sum=pd.merge(my\_sale,product,how=**'inner'**,on=**'p\_id'**)  
print(my\_sum)  
print()  
  
  
*# left join and inner join will give the same result. Why?\_\_*my\_sum=pd.merge(my\_sale,product,how=**'left'**,on=**'p\_id'**)  
print(my\_sum)  
print()  
  
  
*# adding column total\_sales by multiplying qnty with price.*my\_sum[**'total\_sale'**]=my\_sum[**'qty'**]\*my\_sum[**'price'**]  
print(my\_sum)  
print()  
  
  
*# List of quantity sold against each Store with total price of the store*my\_sale=sales.groupby([**'product'**,**'p\_id'**, **'store'**])[[**'qty'**]].sum()  
print(my\_sale)  
print()  
  
  
my\_sale=sales.groupby([**'product'**,**'p\_id'**, **'store'**])[[**'qty'**]].sum().reset\_index()  
print(my\_sale)  
print()  
  
my\_sum=pd.merge(my\_sale,product,how=**'left'**)  
print(my\_sum)  
print()  
  
  
my\_sum[**'sales\_total'**]=my\_sum[**'qty'**]\*my\_sum[**'price'**]  
print(my\_sum)  
print()  
  
  
print(my\_sum.groupby([**'store'**])[[**'qty'**,**'sales\_total'**]].sum())  
  
  
*# List of products which are not sold*my\_data=pd.merge(product,sales,how=**'left'**)  
print(my\_data)  
print()  
  
*#print(my\_data['sale\_id'].isna())*my\_data=my\_data[my\_data[**'sale\_id'**].isnull()] *# products which are not sold*print(my\_data)  
print()  
  
*#my\_data.loc[:,'product\_y'] # to display only produts column*print(my\_data[**'product'**].unique())  
  
  
*# ### List of customers who have not purchased any product*my\_data=pd.merge(sales,customer,on=**'c\_id'**,how=**'right'**)  
my\_data=my\_data[my\_data[**'sale\_id'**].isnull()] *# products which are not sold  
#print(my\_data)*print(my\_data.loc[:,**'Customer'**]) *# to display customers who has not purchased*