In [ ]:
```
#*******************************************************************************************

#                      Project 1 : Assignment


#*******************************************************************************************
#importing the packages and setting the alias

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
% matplotlib inline


# reading the csv data into a dataframe and printing 2 records of each
print('-'*80,"\n The Data wrangling Data-text file \n",'-'*80)
df = pd.read_csv('https://raw.githubusercontent.com/jackiekazil/data-wrangling/master/data/chp3/data-text.cs
v')
print(df.head(2))

print('-'*80,"\n The Data wrangling berlin-weather file \n",'-'*80)

df1 = pd.read_csv('https://raw.githubusercontent.com/kjam/data-wrangling-pycon/master/data/berlin_weather_old
est.csv')
print(df1.head(2))
```

In [ ]:
```
#*********************************************************************************************
#                       task1 : Getting the MetaData for files
#*********************************************************************************************


#importing the packages and setting the alias

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
% matplotlib inline


# reading the csv data into a dataframe and printing 2 records of each
#print('-'*80,"\n The Data wrangling Data-text file \n",'-'*80)
df = pd.read_csv('https://raw.githubusercontent.com/jackiekazil/data-wrangling/master/data/chp3/data-text.cs
v')
#print(df.head(2))

#print('-'*80,"\n The Data wrangling berlin-weather file \n",'-'*80)

df1 = pd.read_csv('https://raw.githubusercontent.com/kjam/data-wrangling-pycon/master/data/berlin_weather_old
est.csv')
#print(df1.head(2))



#1. Getting the Metadata for the above two files

print('-'*80,"\n The Data wrangling Data-text metaData \n",'-'*80)
df.info()

print('-'*80,"\n The Data wrangling berlin-weather metaData \n",'-'*80)
df1.info()
```

In [ ]:
```
#*********************************************************************************
#                       task 2 : Getting the row names  for files
#*********************************************************************************

#importing the packages and setting the alias

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
% matplotlib inline


# reading the csv data into a dataframe and printing 2 records of each
#print('-'*80,"\n The Data wrangling Data-text file \n",'-'*80)
df = pd.read_csv('https://raw.githubusercontent.com/jackiekazil/data-wrangling/master/data/chp3/data-text.csv')
#print(df.head(2))

#print('-'*80,"\n The Data wrangling berlin-weather file \n",'-'*80)

df1 = pd.read_csv('https://raw.githubusercontent.com/kjam/data-wrangling-pycon/master/data/berlin_weather_oldest.csv')
#print(df1.head(2))


# 2. To get the row names for both the Data Frames

print('-'*80,"\n The Data wrangling Data-text row names are : \n",'-'*80)

print(df.index.values)

print('-'*80,"\n The Data wrangling berlin-weather row names are : \n",'-'*80)

df1.index.values
```

In [ ]:
```
#*********************************************************************************************
#                       task3 : Changing temporary Column Name Data wrangling Data Text files
#*********************************************************************************************


#importing the packages and setting the alias

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
% matplotlib inline


# reading the csv data into a dataframe and printing 2 records of each
#print('-'*80,"\n The Data wrangling Data-text file \n",'-'*80)
df = pd.read_csv('https://raw.githubusercontent.com/jackiekazil/data-wrangling/master/data/chp3/data-text.cs
v')
#print(df.head(2))

#print('-'*80,"\n The Data wrangling berlin-weather file \n",'-'*80)

df1 = pd.read_csv('https://raw.githubusercontent.com/kjam/data-wrangling-pycon/master/data/berlin_weather_old
est.csv')
#print(df1.head(2))

# 3. Changing Column name of Data wrangling Data text file Column Indicator to Indicator_id

print('-'*80,"\n The Data wrangling Data Text column names after changing Indicator to Indicator_Id  : \n",'-
'*80)
df.rename(index=str,columns={"Indicator":"Indicator_id"}).head(2)
```

In [ ]:
```
#*********************************************************************************************
#                       task4  : Changing Column Name permanenently Data wrangling Data Text files
#*********************************************************************************************


#importing the packages and setting the alias

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
% matplotlib inline


# reading the csv data into a dataframe and printing 2 records of each
#print('-'*80,"\n The Data wrangling Data-text file \n",'-'*80)
df = pd.read_csv('https://raw.githubusercontent.com/jackiekazil/data-wrangling/master/data/chp3/data-text.cs
v')
#print(df.head(2))

#print('-'*80,"\n The Data wrangling berlin-weather file \n",'-'*80)

df1 = pd.read_csv('https://raw.githubusercontent.com/kjam/data-wrangling-pycon/master/data/berlin_weather_old
est.csv')
#print(df1.head(2))



# 4. Changing Column name of Data wrangling Data text file Column Indicator to Indicator_id and Store it perm
anently
print(df.head(2))
print('-'*80,"\n The Data wrangling Data Text column names after changing Indicator to Indicator_Id  : \n",'-
'*80)
df.rename(inplace=True, columns={"Indicator":"Indicator_id"})
df.head(2)
```

In [ ]:
```
#*********************************************************************************************
#                       task5 : Changing multiple Column Names Data wrangling Data Text files
#*********************************************************************************************
#importing the packages and setting the alias

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
% matplotlib inline


# reading the csv data into a dataframe and printing 2 records of each
#print('-'*80,"\n The Data wrangling Data-text file \n",'-'*80)
df = pd.read_csv('https://raw.githubusercontent.com/jackiekazil/data-wrangling/master/data/chp3/data-text.csv')
#print(df.head(2))

#print('-'*80,"\n The Data wrangling berlin-weather file \n",'-'*80)

df1 = pd.read_csv('https://raw.githubusercontent.com/kjam/data-wrangling-pycon/master/data/berlin_weather_oldest.csv')
#print(df1.head(2))

print('-'*80,"\n The Data wrangling Data Text 2 column names changed Indicator and PUBLISH STATES  : \n",'-'*80)
df.rename(inplace=True, columns={"Indicator":"Indicator_id","PUBLISH STATES":"PUBLICATION STATUS"})
df.head(2)
```

In [ ]:
```
#*********************************************************************************
#                      task6 : Arranging Column Year in ascending order for  Data wrangling Data Text files
#*********************************************************************************
#importing the packages and setting the alias

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
% matplotlib inline


# reading the csv data into a dataframe and printing 2 records of each
#print('-'*80,"\n The Data wrangling Data-text file \n",'-'*80)
df = pd.read_csv('https://raw.githubusercontent.com/jackiekazil/data-wrangling/master/data/chp3/data-text.csv')
#print(df.head(2))

#print('-'*80,"\n The Data wrangling berlin-weather file \n",'-'*80)

df1 = pd.read_csv('https://raw.githubusercontent.com/kjam/data-wrangling-pycon/master/data/berlin_weather_oldest.csv')
#print(df1.head(2))

df.rename(inplace=True, columns={"Indicator":"Indicator_id","PUBLISH STATES":"PUBLICATION STATUS"})




print('-'*80,"\n The Data wrangling Data Text arranging year in ascending order  : \n",'-'*80)

df.sort_values('Year', ascending = True )
df.head(2)
```

In [ ]:
```
#*********************************************************************************
#                      task7  : Arranging multiple  Columns (Year , Country) in ascending order for  Data wran
gling Data Text files
#*********************************************************************************
#importing the packages and setting the alias

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
% matplotlib inline


# reading the csv data into a dataframe and printing 2 records of each
#print('-'*80,"\n The Data wrangling Data-text file \n",'-'*80)
df = pd.read_csv('https://raw.githubusercontent.com/jackiekazil/data-wrangling/master/data/chp3/data-text.cs
v')
#print(df.head(2))

#print('-'*80,"\n The Data wrangling berlin-weather file \n",'-'*80)

df1 = pd.read_csv('https://raw.githubusercontent.com/kjam/data-wrangling-pycon/master/data/berlin_weather_old
est.csv')
#print(df1.head(2))

df.rename(inplace=True, columns={"Indicator":"Indicator_id","PUBLISH STATES":"PUBLICATION STATUS"})




print('-'*80,"\n The Data wrangling Data Text arranging (year,Country) in ascending order  : \n",'-'*80)

df.sort_values(['Year','Country'], ascending = True )
df.head(2)
```

In [ ]:
```
#*********************************************************************************************
#                       task8  : Making Country first column  for  Data wrangling Data Text files
#*********************************************************************************************
#importing the packages and setting the alias

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
% matplotlib inline


# reading the csv data into a dataframe and printing 2 records of each
#print('-'*80,"\n The Data wrangling Data-text file \n",'-'*80)
df = pd.read_csv('https://raw.githubusercontent.com/jackiekazil/data-wrangling/master/data/chp3/data-text.cs
v')
#print(df.head(2))

#print('-'*80,"\n The Data wrangling berlin-weather file \n",'-'*80)

df1 = pd.read_csv('https://raw.githubusercontent.com/kjam/data-wrangling-pycon/master/data/berlin_weather_old
est.csv')
#print(df1.head(2))

df.rename(inplace=True, columns={"Indicator":"Indicator_id","PUBLISH STATES":"PUBLICATION STATUS"})

df.sort_values(['Year','Country'], ascending = True )




# get the list of columns and get the index of country in this list

cols = df.columns.tolist()
lenCols = len(df.columns.tolist())
inCntry = cols.index('Country')

# making a list of column list with Country as first column and followed by others
newCols = []
newCols.append(cols[inCntry])
for i in range(inCntry):
    newCols.append(cols[i])
for i in range((inCntry+1),lenCols):
```

```
    newCols.append(cols[i])

print('-'*80,"\n The Data wrangling Data Text new list of Columns by looping and the Dataframe  : \n",'-'*80)
#print(newCols)
df = df[newCols]
print(df.head(2))

print('-'*80,"\n In The Data wrangling Data Text arranging (Country) as first column  : \n",'-'*80)
# The above can also be achieved by
df[pd.unique(['Country']+df.columns.tolist())].head(2)
```

In [ ]:
```
#*********************************************************************************************
#                       task9  : Getting Column Array using Variable  for  Data wrangling Data Text files
#*********************************************************************************************
#importing the packages and setting the alias

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
% matplotlib inline


# reading the csv data into a dataframe and printing 2 records of each
#print('-'*80,"\n The Data wrangling Data-text file \n",'-'*80)
df = pd.read_csv('https://raw.githubusercontent.com/jackiekazil/data-wrangling/master/data/chp3/data-text.cs
v')
#print(df.head(2))

#print('-'*80,"\n The Data wrangling berlin-weather file \n",'-'*80)

df1 = pd.read_csv('https://raw.githubusercontent.com/kjam/data-wrangling-pycon/master/data/berlin_weather_old
est.csv')
#print(df1.head(2))

df.rename(inplace=True, columns={"Indicator":"Indicator_id","PUBLISH STATES":"PUBLICATION STATUS"})


# Using Col as a variable for Who region string and getting the column WHO region array

print('-'*80,"\n Getting the Column Array WHO region by using the variable\n",'-'*80)
#df.columns.values
col1= 'WHO region'
df[col1].values
```

In [ ]:
```
#*********************************************************************************************
#                       task10  : Getting Subset of rows 11,24,37  for  Data wrangling Data Text files
#*********************************************************************************************
#importing the packages and setting the alias

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
% matplotlib inline


# reading the csv data into a dataframe and printing 2 records of each
#print('-'*80,"\n The Data wrangling Data-text file \n",'-'*80)
df = pd.read_csv('https://raw.githubusercontent.com/jackiekazil/data-wrangling/master/data/chp3/data-text.csv')
#print(df.head(2))

#print('-'*80,"\n The Data wrangling berlin-weather file \n",'-'*80)

df1 = pd.read_csv('https://raw.githubusercontent.com/kjam/data-wrangling-pycon/master/data/berlin_weather_oldest.csv')
#print(df1.head(2))

df.rename(inplace=True, columns={"Indicator":"Indicator_id","PUBLISH STATES":"PUBLICATION STATUS"})

df.sort_values(['Year','Country'], ascending = True )


print('-'*80,"\n Getting the Subset of DataFrame of Rows 11,24,37 for DataWrangling Data text file\n",'-'*80)

df.iloc[[11,24,37]]
```

In [ ]:
```
#*********************************************************************************
#                      task11  : Getting Subset of rows Excluding 5,12,23,56  for  Data wrangling Data Text fi
les
#*********************************************************************************
#importing the packages and setting the alias

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
% matplotlib inline


# reading the csv data into a dataframe and printing 2 records of each
#print('-'*80,"\n The Data wrangling Data-text file \n",'-'*80)
df = pd.read_csv('https://raw.githubusercontent.com/jackiekazil/data-wrangling/master/data/chp3/data-text.cs
v')
#print(df.head(2))

#print('-'*80,"\n The Data wrangling berlin-weather file \n",'-'*80)

df1 = pd.read_csv('https://raw.githubusercontent.com/kjam/data-wrangling-pycon/master/data/berlin_weather_old
est.csv')
#print(df1.head(2))

df.rename(inplace=True, columns={"Indicator":"Indicator_id","PUBLISH STATES":"PUBLICATION STATUS"})

df.sort_values(['Year','Country'], ascending = True )



print('-'*80,"\n Getting the Subset of DataFrame of Excluding Rows 5,12,23,56 for DataWrangling Data text fil
e\n",'-'*80)

df.drop([5,12,23,56])
```

In [ ]:
```
#*********************************************************************************************
#                       task12  : Loading Data Sets from CSV File and Show Trnsaxtions and matching USers Data
#*********************************************************************************************
#importing the packages and setting the alias

import pandas as pd



users = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/users.csv' )


sessions = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/sessions.csv' )


products = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/products.csv' )


transactions = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/transactions.c
sv')

# print("\n Printing first 2 rows of Users Data\n",'-'*80)
# print(users.head(2))
# print("-"*80)

# print("\n Printing first 5 rows of Products Data\n",'-'*80)
# print(products.head())
# print("-"*80)

# print("\n Printing first 5 rows of Sessions Data\n",'-'*80)
# print(sessions.head())
# print("-"*80)

# print("\n Printing first 2 rows of transactions Data\n",'-'*80)
# print(transactions.head(2))
# print("-"*80)

#Converting all the Datetime fields not having values to NaT

users['Registered'] = pd.to_datetime(users.Registered,errors='coerce')
users['Cancelled'] = pd.to_datetime(users.Cancelled,errors='coerce')
transactions['TransactionDate'] = pd.to_datetime(transactions.TransactionDate,errors='coerce')
```

```
print("\n Joining Users to transactions with all transactions and matching users :\n ",'-'*80)

transactions.merge(users, left_on="UserID",right_on="UserID",how="left")
```

In [ ]:
```
#*********************************************************************************
#                      task13  : Transactions having USers not in Users
#*********************************************************************************
#importing the packages and setting the alias

import pandas as pd



users = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/users.csv' )


sessions = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/sessions.csv' )


products = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/products.csv' )


transactions = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/transactions.c
sv')

# Printing transactions having Users not in UserID

print('-'*80,"\n Trasnactions having Users that are not in the Users Data Set \n",'-'*80)
transactions[~transactions['UserID'].isin(users['UserID'])]
```

In [ ]:
```
#*********************************************************************************
#                      task 14  : Transactions having Users matching in Users
#*********************************************************************************
#importing the packages and setting the alias

import pandas as pd



users = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/users.csv' )


sessions = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/sessions.csv' )


products = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/products.csv' )


transactions = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/transactions.c
sv')

# Printing transactions having Users not in UserID

print('-'*80,"\n Trasnactions having Users that are in the Users Data Set \n",'-'*80)

#Converting all the Datetime fields not having values to NaT

users['Registered'] = pd.to_datetime(users.Registered,errors='coerce')
users['Cancelled'] = pd.to_datetime(users.Cancelled,errors='coerce')
transactions['TransactionDate'] = pd.to_datetime(transactions.TransactionDate,errors='coerce')


transactions.merge(users,left_on='UserID',right_on='UserID',how='inner')
```

In [ ]:
```
#*********************************************************************************
#                      task 15  : Transactions having Users matching and also non-matching in Users
#*********************************************************************************
#importing the packages and setting the alias

import pandas as pd



users = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/users.csv' )


sessions = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/sessions.csv' )


products = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/products.csv' )


transactions = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/transactions.c
sv')

#Converting all the Datetime fields not having values to NaT

users['Registered'] = pd.to_datetime(users.Registered,errors='coerce')
users['Cancelled'] = pd.to_datetime(users.Cancelled,errors='coerce')
transactions['TransactionDate'] = pd.to_datetime(transactions.TransactionDate,errors='coerce')
# Printing transactions having Users not in UserID

print('-'*80,"\n Trasnactions having Users  that are in and also not in  the Users Data Set \n",'-'*80)

transactions.merge(users,left_on='UserID',right_on='UserID',how='outer')
```

In [ ]:
```
#*****************************************************************************************
#                      task 16  : To Determine which sessions occured on the same day user registers
#
#*****************************************************************************************
#importing the packages and setting the alias

import pandas as pd



users = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/users.csv' )


sessions = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/sessions.csv' )


products = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/products.csv' )


transactions = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/transactions.c
sv')

# Printing the sessions when USers had registered the same Day

print('-'*80,"\n To show sessions that occured on same day as user egistered  \n",'-'*80)

users['Registered'] = pd.to_datetime(users.Registered)

sessions['SessionDate'] = pd.to_datetime(sessions.SessionDate)


users.merge(sessions, left_on =['UserID','Registered'], right_on =['UserID','SessionDate'],how='inner')
```

In [ ]:
```
#*********************************************************************************
#                      task 17  : Building DataSet with UserID and ProductID combinaton
#
#*********************************************************************************
#importing the packages and setting the alias

import pandas as pd



users = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/users.csv' )


products = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/products.csv' )

transactions = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/transactions.c
sv')

# Printing the DataSet having all combinations of UserId and ProductID

print('-'*80,"\n To Build DataSet with UserId and ProductID combination  \n",'-'*80)


df1 = pd.DataFrame(users['UserID'])
df1['Key'] = 0
df2 = pd.DataFrame(products['ProductID'])
df2['Key'] = 0

# now merging the two DF's on common field 'Key' and dropping it later
df1 = df1.merge(df2,on='Key',how='outer')

df1.drop('Key',axis=1, inplace=True)
df1
```

In [ ]:
```
#*****************************************************************************
#                      task 18  : To Determine how much quantity of each product was puchased
#                               : by each user
#
#*****************************************************************************
#importing the packages and setting the alias

import pandas as pd
import sqlite3



users = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/users.csv' )


products = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/products.csv' )

transactions = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/transactions.c
sv')

# connecting to the memory of database
conn = sqlite3.connect(":memory:")
c = conn.cursor()

# setting the Users, products and Transactions  Dataframe to memory database
users.to_sql("users",conn, if_exists="replace")
products.to_sql("products",conn,if_exists="replace")
transactions.to_sql("transactions",conn,if_exists="replace")



# following is the Expected Output as needed by another way

#*****************************************************************************
# from the previous solutions

df1 = pd.DataFrame(users['UserID'])
df1['Key'] = 0
df2 = pd.DataFrame(products['ProductID'])
df2['Key'] = 0

# now merging the two DF's on common field 'Key' and dropping it later
```

```
df1 = df1.merge(df2,on='Key',how='outer')

df1.drop('Key',axis=1, inplace=True)



transactions['UserID'] = pd.to_numeric(transactions.UserID, downcast='signed',errors='coerce')

df1 = df1.merge(transactions , on=['UserID','ProductID'],how='outer')
df1.drop(['TransactionID','TransactionDate'],axis=1, inplace=True)

# filling the null values for Quantity as 0 and for UserID as some high number so it is displayed last
df1['Quantity'] = df1['Quantity'].fillna(0)
df1['UserID'] = df1['UserID'].fillna(99)
#df1
df1['UserID'] = pd.to_numeric(df1.UserID, downcast='integer',errors='coerce')


# setting the Users, products and Transactions  Dataframe to memory database
df1.to_sql("UData",conn, if_exists="replace")

# Query to get the Products bought along with quantity by the User
sqlQuery = '''select UserID, ProductID, sum(Quantity)
                from UData
                group by UserId , ProductID
                  ; '''
print('-'*80,"\n Following is the Data showing the Quantity of Products purchased by Users : \n",'-'*80)
#print(pd.read_sql_query(sqlQuery, conn))
dff = pd.read_sql_query(sqlQuery, conn)
conn.close()
dff
```

In [ ]:
```
#*********************************************************************************
#                      task 19  : For Each User, Get each possible pair of transactions
#                               :
#
#*********************************************************************************
#importing the packages and setting the alias

import pandas as pd
import sqlite3



transactions = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/transactions.csv')


#Sorting the Transctions based on the User and creating two DataFrames of Transactions

transactionsx = transactions.sort_values('UserID',ascending=False)
transactionsy = transactions.sort_values('UserID',ascending=False)

print('-'*80,"\n Following is the Transactions pair for each Users : \n",'-'*80)
transactionsx.merge(transactionsy,left_on=['UserID'],right_on=['UserID'],how="outer")
```

In [ ]:
```
#*****************************************************************************************
#
#                 task 20  : For Each User, to join with his/her first occurring transaction
#
#*****************************************************************************************
#importing the packages and setting the alias

import pandas as pd
import sqlite3



users = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/users.csv' )


transactions = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/transactions.c
sv')

# connecting to the memory of database
conn = sqlite3.connect(":memory:")
c = conn.cursor()


# setting the Users, products and Transactions  Dataframe to memory database
users.to_sql("users",conn, if_exists="replace")
transactions.to_sql("transactions",conn,if_exists="replace")

# merging the Users and Transactions Data using Left join , as all USers needed
df1 = users.merge(transactions, left_on=['UserID'], right_on=['UserID'], how='left')

# Converting the DataFrame to Sql DB in memory
df1.to_sql("UData",conn,if_exists="replace")

sqlQuery = ''' select UserID,User,Gender,Registered,Cancelled,TransactionID,min(TransactionDate) as Transacti
onDate,ProductID,Quantity from UData
               group by UserID ;  '''


print('-'*80,"\n Following is the Data Users with their first transanctions  : \n",'-'*80)

data = pd.read_sql_query(sqlQuery, conn)
conn.close()
```

```
#Converting all the Datetime fields not having values to NaT

data['Registered'] = pd.to_datetime(data.Registered,errors='coerce')
data['Cancelled'] = pd.to_datetime(data.Cancelled,errors='coerce')
data['TransactionDate'] = pd.to_datetime(data.TransactionDate,errors='coerce')
data
```

In [ ]:
```
#*********************************************************************************
#                      task 21  : Test to see if we can drop columns
#                               :Data Frame is obtained from Task 20 Output
#
#*********************************************************************************

my_columns = list(data.columns)
print(my_columns)

# Keeping only those columns which have at least 4 non na values
print(list(data.dropna(thresh=int(data.shape[0] * .9),axis=1).columns))


#set threshold to drop NAs
# ['UserID', 'User', 'Gender', 'Registered']

missing_info = list(data.columns[data.isnull().any()])

print(missing_info)

# missing_info
# ['Cancelled', 'TransactionID', 'TransactionDate', 'ProductID', 'Quantity']

print("Count of missing Data\n",'-'*80)

for col in missing_info:
    num_missing = data[data[col].isnull() == True].shape[0]
    print('number missing for column {}: {}'.format(col, num_missing))


# Output: Count of missing data number missing for column Cancelled: 3 number missing for column TransactionI
D: 2 number missing for column TransactionDate: 2 number missing for column ProductID: 2 number missing for c
olumn Quantity: 2
for col in missing_info:
    num_missing = data[data[col].isnull() == True].shape[0]
    print('number missing for column {}: {}'.format(col, num_missing))


#count of missing data

print("\nOutput of percentage missing data\n")
for col in missing_info:
```

```
    percent_missing = data[data[col].isnull() == True].shape[0]/data.shape[0]
    print('percent missing for column {}: {}'.format(col,percent_missing))
```