SQL - 2 - Assignment

By Prakash Ghosh

Read the following data set:

https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data (https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data)

In [1]:

```
# import the libraries
import numpy as np
import pandas as pd
import sqlite3
import pandasql
import sqlalchemy
from sqlalchemy import orm
```

In [2]:

```
# Read Adult Dataset from the URL
str_URL='https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data'
df_adult=pd.read_csv(str_URL,header=None, index_col=False)

# Create a list of Columns to Rename the columns of Adult Dataset
lst_df_Col=['AGE',
        'WORKCLASS',
        'FNLWGT',
        'EDUCATION',
        'EDUCATION_NUM',
        'MARITAL_STATUS',
        'OCCUPATION',
        'RELATIONSHIP',
        'RACE',
        'SEX',
        'CAPITAL_GAIN',
        'CAPITAL_LOSS',
        'HOURS_PER_WEEK',
        'NATIVE_COUNTRY',
        'GT50_OR_LT50K']

df_adult.columns=lst_df_Col
# Print first 5 records of df_adult_data (after renaming the columns)
df_adult.head(5)
```

Out[2]:

	AGE	WORKCLASS	FNLWGT	EDUCATION	EDUCATION_NUM	MARITAL_STATUS
0	39	State-gov	77516	Bachelors	13	Never-married
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse
2	38	Private	215646	HS-grad	9	Divorced
3	53	Private	234721	11th	7	Married-civ-spouse
4	28	Private	338409	Bachelors	13	Married-civ-spouse

Task 1. Create an sqlalchemy engine using a sample from the data set

In [7]:

```
# Create a sqlite Database connectiona and Create the table ADULTS
engine = sqlalchemy.create_engine('sqlite:///:memory:',echo=False)
conn = engine.connect()

# Create the table ADULTS
engine.execute('''
    CREATE TABLE IF NOT EXISTS ADULTS (
         AGE             INTEGER,
         WORKCLASS       VARCHAR(100),
         FNLWGT          INTEGER,
         EDUCATION       VARCHAR(100),
         EDUCATION_NUM   INTEGER,
         MARITAL_STATUS  VARCHAR(100),
         OCCUPATION      VARCHAR(100),
         RELATIONSHIP    VARCHAR(100),
         RACE            VARCHAR(100),
         SEX             VARCHAR(20),
         CAPITAL_GAIN    INTEGER,
         CAPITAL_LOSS    INTEGER,
         HOURS_PER_WEEK  INTEGER,
         NATIVE_COUNTRY  VARCHAR(100),
         GT50_OR_LT50K   VARCHAR(20))
''')

# Take all sample and create sample DataFrame
df_adult_sample=df_adult

# Create a dict from the Dataframe df_adult_sample
dict_adult = df_adult_sample.to_dict(orient='records')

# Create the table ADULTS
metadata = sqlalchemy.schema.MetaData(bind=engine)
ADULT_TAB = sqlalchemy.Table('ADULTS', metadata, autoload=True)

# Open a session for the bulk insert
Session = orm.sessionmaker(bind=engine)
session = Session()

# Inser the dataframe into the database in one bulk
conn.execute(ADULT_TAB.insert(), dict_adult)

# Commit the changes
session.commit()

# Close the session
session.close()

# print first 5 records from the ADULTS table
sql_select="SELECT * FROM ADULTS LIMIT 5;"
conn=engine

result_adult_data=pd.read_sql_query(sql_select, conn)
print( "\n5 records from the adult sqladb:")
result_adult_data
```

```
5 records from the adult sqladb:
```

```
Out[7]:
```

	AGE	WORKCLASS	FNLWGT	EDUCATION	EDUCATION_NUM	MARITAL_STATUS
0	39	State-gov	77516	Bachelors	13	Never-married
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse
2	38	Private	215646	HS-grad	9	Divorced
3	53	Private	234721	11th	7	Married-civ-spouse
4	28	Private	338409	Bachelors	13	Married-civ-spouse

Task 2. Write two basic update queries

In [8]:

```
# First update Query - Update all the MARITAL_STATUS like 'Married%' - update them to M
arried
print('First update Query:')
sql_update1 = " UPDATE ADULTS SET MARITAL_STATUS='Married' where MARITAL_STATUS like '
 Married%';"
print(sql_update1,'\n\n')
engine.execute(sql_update1)


# Second update Query - Update all the NATIVE_COUNTRY = ? - update them to Married
print('Second update Query:')
sql_update2 = " UPDATE ADULTS SET NATIVE_COUNTRY='Not Known' where NATIVE_COUNTRY ='
 ?';"
print(sql_update2,'\n\n')
engine.execute(sql_update2)


print('\nSelect the updated records (first 10 only):')
sql_select="SELECT * FROM ADULTS WHERE (NATIVE_COUNTRY='Not Known') OR (MARITAL_STATUS
='Married') LIMIT 10;"
conn=engine

result_adult_data=pd.read_sql_query(sql_select, conn)
result_adult_data
```

```
First update Query:
 UPDATE ADULTS SET MARITAL_STATUS='Married' where MARITAL_STATUS like ' Ma
rried%';


Second update Query:
 UPDATE ADULTS SET NATIVE_COUNTRY='Not Known' where NATIVE_COUNTRY =' ?';



Select the updated records (first 10 only):


Out[8]:
```

	AGE	WORKCLASS	FNLWGT	EDUCATION	EDUCATION_NUM	MARITAL_STATUS
0	50	Self-emp-not-inc	83311	Bachelors	13	Married
1	53	Private	234721	11th	7	Married
2	28	Private	338409	Bachelors	13	Married
3	37	Private	284582	Masters	14	Married
4	49	Private	160187	9th	5	Married
5	52	Self-emp-not-inc	209642	HS-grad	9	Married
6	42	Private	159449	Bachelors	13	Married
7	37	Private	280464	Some-college	10	Married
8	30	State-gov	141297	Bachelors	13	Married
9	40	Private	121772	Assoc-voc	11	Married

Task 3. Write two delete queries

In [9]:

```
print('First Delete Query:')
#form the Query
sql_delete= "DELETE FROM ADULTS WHERE OCCUPATION =' ?' ;"
print(sql_delete,'\n\n')

# Check no of records where OCCUPATION==' ?' before delete
sql_select="SELECT * FROM ADULTS WHERE OCCUPATION = ' ?';"
conn=engine
result_adult_data=pd.read_sql_query(sql_select, conn)
print('No of records where OCCUPATION=\' ?\' before delete:\t',\
      result_adult_data['OCCUPATION'].count())

# Execute the Delete Query
engine.execute(sql_delete)

# Check no of records where OCCUPATION==' ?' after delete
sql_select="SELECT * FROM ADULTS;"
conn=engine
result_adult_data=pd.read_sql_query(sql_select, conn)

print('No of records where OCCUPATION=\' ?\' after delete:\t', \
      result_adult_data['OCCUPATION'][(result_adult_data.OCCUPATION==' ?')].count())
```

```
First Delete Query:
DELETE FROM ADULTS WHERE OCCUPATION =' ?' ;


No of records where OCCUPATION=' ?' before delete:       1843
No of records where OCCUPATION=' ?' after delete:        0
```

In [10]:

```
print('2nd Delete Query:')
#form the Query
sql_delete = "DELETE FROM ADULTS WHERE WORKCLASS =' ?' ;"
print(sql_delete,'\n\n')

# Check no of records where WORKCLASS==' ?' before delete
sql_select="SELECT * FROM ADULTS WHERE WORKCLASS = ' ?';"
conn=engine
result_adult_data=pd.read_sql_query(sql_select, conn)
print('No of records where WORKCLASS=\' ?\' before delete:\t', \
              result_adult_data['WORKCLASS'].count())

# Execute the Delete Query
engine.execute(sql_delete)

# Check no of records where WORKCLASS==' ?' after delete
sql_select="SELECT * FROM ADULTS;"
conn=engine
result_adult_data=pd.read_sql_query(sql_select, conn)

print('No of records where WORKCLASS=\' ?\' after delete:\t', \
      result_adult_data['WORKCLASS'][(result_adult_data.WORKCLASS==' ?')].count())
```

```
2nd Delete Query:
DELETE FROM ADULTS WHERE WORKCLASS =' ?' ;


No of records where WORKCLASS=' ?' before delete:	 0
No of records where WORKCLASS=' ?' after delete:	 0
```

Task 4. Write two filter queries

In [11]:

```
print('First filter Query:')
sql_select="SELECT * FROM ADULTS where AGE >=80 AND EDUCATION=' Masters';"
print(sql_select)

conn=engine
result_adult_data=pd.read_sql_query(sql_select, conn)
result_adult_data
```

```
First filter Query:
SELECT * FROM ADULTS where AGE >=80 AND EDUCATION=' Masters';
```

Out[11]:

	AGE	WORKCLASS	FNLWGT	EDUCATION	EDUCATION_NUM	MARITAL_STATUS
0	80	Private	157778	Masters	14	Widowed
1	90	Local-gov	227796	Masters	14	Married
2	90	Private	51744	Masters	14	Never-married
3	84	Private	241065	Masters	14	Never-married
4	81	Private	201398	Masters	14	Widowed
5	90	Private	115306	Masters	14	Never-married
6	90	Private	206667	Masters	14	Married
7	86	Private	149912	Masters	14	Never-married

In [12]:

```
print('Second filter Query:')

sql_select="SELECT * FROM ADULTS"
sql_select=sql_select+ " WHERE SEX=' Female' "
sql_select=sql_select+ " AND MARITAL_STATUS=' Never-married'"
sql_select=sql_select+ " AND WORKCLASS!=' Private'"
sql_select=sql_select+ " AND AGE<18;"

print(sql_select)

conn=engine
result_adult_data=pd.read_sql_query(sql_select, conn)
result_adult_data
```

```
Second filter Query:
SELECT * FROM ADULTS WHERE SEX=' Female'  AND MARITAL_STATUS=' Never-marri
ed' AND WORKCLASS!=' Private' AND AGE<18;
```

Out[12]:

	AGE	WORKCLASS	FNLWGT	EDUCATION	EDUCATION_NUM	MARITAL_STATUS
0	17	Local-gov	182070	11th	7	Never-married
1	17	Local-gov	246308	11th	7	Never-married
2	17	Local-gov	148194	11th	7	Never-married
3	17	Self-emp-not-inc	228786	10th	6	Never-married
4	17	Local-gov	308901	11th	7	Never-married
5	17	Self-emp-inc	413557	9th	5	Never-married
6	17	Local-gov	244856	11th	7	Never-married
7	17	Local-gov	170916	10th	6	Never-married
8	17	Local-gov	340043	12th	8	Never-married
9	17	Federal-gov	99893	11th	7	Never-married
10	17	Local-gov	39815	10th	6	Never-married

Task 5. Write two function queries

In [13]:

```
print('First function Query:')
sql_select="SELECT COUNT(*) AS COUNT_AGE_GT_60 FROM ADULTS where AGE >=60;"
print(sql_select)

conn=engine
result_adult_data=pd.read_sql_query(sql_select, conn)
result_adult_data
```

```
First function Query:
SELECT COUNT(*) AS COUNT_AGE_GT_60 FROM ADULTS where AGE >=60;
```

Out[13]:

	COUNT_AGE_GT_60
0	2120

In [15]:

```
print('Second function Query:')
sql_select="SELECT SEX, COUNT(*) AS COUNT_GENDER FROM ADULTS GROUP BY SEX;"
print(sql_select)

conn=engine
result_adult_data=pd.read_sql_query(sql_select, conn)
result_adult_data
```

```
Second function Query:
SELECT SEX, COUNT(*) AS COUNT_GENDER FROM ADULTS GROUP BY SEX;
```

Out[15]:

	SEX	COUNT_GENDER
0	Female	9930
1	Male	20788