SQL - 1 - 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)

Rename the columns as per the description from this file:

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

Task:

Create a sql db from adult dataset and name it sqladb

In [1]:

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

In [2]:

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

# Print first 5 records of df_adult_data (without naming the columns)
df_adult_data.head(5)
```

Out[2]:

	0	1	2	3	4	5	6	7	8	9	
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	21
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0

In [3]:

```
# Read Adult Name Dataset to get the column names as per the description
df_adult_column_name = pd.read_csv('https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.names', sep=":")

# Get the Column Names from df_adult_column_name into a list
lst_adult_column_name=df_adult_column_name.iloc[91:106,].index.tolist()

# Reorder columns in the list as per the description provided in the file
lst_adult_column_name=lst_adult_column_name[1:]+lst_adult_column_name[0::-1]
lst_adult_column_name

print('Column Names derived from Adult Name Dataset:\n',lst_adult_column_name)
```

```
Column Names derived from Adult Name Dataset:
 ['age', 'workclass', 'fnlwgt', 'education', 'education-num', 'marital-status', 'occupation', 'relationship', 'race', 'sex', 'capital-gain', 'capital-loss', 'hours-per-week', 'native-country', '>50K, <=50K.']
```

In [4]:

```
# Rename the columns of Adult Dataset as per lst_adult_column_name
df_adult_data.columns = lst_adult_column_name

# Print first 5 records of df_adult_data (after naming the columns)
df_adult_data.head(5)
```

Out[4]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife

In [6]:

```
# Create a sql db from adult dataset and name it sqladb
sqladb = sqlite3.connect("sqladb.db")

# Create the table ADULTS (with same column name with adult dataset)
sqladb.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))
''')
```

Out[6]:

```
<sqlite3.Cursor at 0x1f618029340>
```

In [7]:

```
# Insert df_adult_data data into ADULTS table

sql_insert = "INSERT INTO ADULTS (                  \
                    AGE,                            \
                    WORKCLASS,                      \
                    FNLWGT,                         \
                    EDUCATION,                      \
                    EDUCATION_NUM,                  \
                    MARITAL_STATUS,                 \
                    OCCUPATION,                     \
                    RELATIONSHIP,                   \
                    RACE,                           \
                    SEX,                            \
                    CAPITAL_GAIN,                   \
                    CAPITAL_LOSS,                   \
                    HOURS_PER_WEEK,                 \
                    NATIVE_COUNTRY,                 \
                    GT50_OR_LT50K) values           \
                    (%d,'%s', %d, '%s', %d, '%s','%s','%s','%s','%s',%d,%d,%d,'%s','%s
')"

for index, row in df_adult_data.iterrows():
    sqladb.execute(sql_insert % (row['age'],
                                 row['workclass'],
                                 row['fnlwgt'],
                                 row['education'],
                                 row['education-num'],
                                 row['marital-status'],
                                 row['occupation'],
                                 row['relationship'],
                                 row['race'],row['sex'],
                                 row['capital-gain'],
                                 row['capital-loss'],
                                 row['hours-per-week'],
                                 row['native-country'],
                                 row['>50K, <=50K.']))

sqladb.commit()
```

1. Select 10 records from the adult sqladb

In [8]:

```
# print first 10 records from the ADULTS table
sql_select="SELECT * FROM ADULTS LIMIT 10;"
conn=sqladb

result_adult_data=pd.read_sql_query(sql_select, conn)
print( "\n10 records from the adult sqladb:")
result_adult_data
```

```
10 records from the adult sqladb:
```

Out[8]:

	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
5	37	Private	284582	Masters	14	Married-civ-spouse
6	49	Private	160187	9th	5	Married-spouse-absent
7	52	Self-emp-not-inc	209642	HS-grad	9	Married-civ-spouse
8	31	Private	45781	Masters	14	Never-married
9	42	Private	159449	Bachelors	13	Married-civ-spouse

2. Show me the average hours per week of all men who are working in private sector

In [9]:

```
# print the average hours per week of all men who are working in private sector

# In the dataset for Men: SEX= ' Male' and for private sector: WORKCLASS=' Private'
sql_select="SELECT SEX, WORKCLASS, AVG(HOURS_PER_WEEK) FROM ADULTS WHERE SEX=' Male' an
d WORKCLASS=' Private'"

result_avg_hr_per_week=pd.read_sql_query(sql_select, conn)

print( "\naverage hours per week of all men who are working in private sector:")
result_avg_hr_per_week
```

```
average hours per week of all men who are working in private sector:
```

Out[9]:

	SEX	WORKCLASS	AVG(HOURS_PER_WEEK)
0	Male	Private	42.221226

3. Show me the frequency table for education, occupation and relationship, separately

In [10]:

```
# frequency table for education
sql_select="SELECT EDUCATION, COUNT(EDUCATION) FROM ADULTS GROUP BY EDUCATION;"
result_frequency_education=pd.read_sql_query(sql_select, conn)
print('\nfrequency table for education:')
result_frequency_education
```

```
frequency table for education:
```

Out[10]:

	EDUCATION	COUNT(EDUCATION)
0	10th	933
1	11th	1175
2	12th	433
3	1st-4th	168
4	5th-6th	333
5	7th-8th	646
6	9th	514
7	Assoc-acdm	1067
8	Assoc-voc	1382
9	Bachelors	5355
10	Doctorate	413
11	HS-grad	10501
12	Masters	1723
13	Preschool	51
14	Prof-school	576
15	Some-college	7291

In [11]:

```
# frequency table for occupation
sql_select="SELECT OCCUPATION, COUNT(OCCUPATION) FROM ADULTS GROUP BY EDUCATION;"
result_frequency_occupation=pd.read_sql_query(sql_select, conn)
print('\nfrequency table for occupation:')
result_frequency_occupation
```

```
frequency table for occupation:
```

Out[11]:

	OCCUPATION	COUNT(OCCUPATION)
0	Handlers-cleaners	933
1	Sales	1175
2	Protective-serv	433
3	Machine-op-inspct	168
4	Machine-op-inspct	333
5	Craft-repair	646
6	Other-service	514
7	Tech-support	1067
8	Sales	1382
9	Prof-specialty	5355
10	?	413
11	Exec-managerial	10501
12	Exec-managerial	1723
13	Other-service	51
14	Prof-specialty	576
15	Protective-serv	7291

In [12]:

```
# frequency table for occupation
sql_select="SELECT RELATIONSHIP, COUNT(RELATIONSHIP) FROM ADULTS GROUP BY RELATIONSHI
P;"
result_frequency_relationship=pd.read_sql_query(sql_select, conn)
print('\nfrequency table for relationship:')
result_frequency_relationship
```

```
frequency table for relationship:
```

Out[12]:

	RELATIONSHIP	COUNT(RELATIONSHIP)
0	Husband	13193
1	Not-in-family	8305
2	Other-relative	981
3	Own-child	5068
4	Unmarried	3446
5	Wife	1568

4. Are there any people who are married, working in private sector and having a masters degree

In [13]:

```
# Solution -1
# In the dataset
#    for married:         MARITAL_STATUS like ' Married%'
#    for private sector:  WORKCLASS = ' Private'
#    for masters:         EDUCATION = ' Masters'

sql_select = "SELECT MARITAL_STATUS, WORKCLASS, EDUCATION, COUNT(*) FROM ADULTS "
sql_select = sql_select + " WHERE MARITAL_STATUS like ' Married%' AND WORKCLASS = ' Pri
vate' AND EDUCATION = ' Masters'"
sql_select = sql_select + " GROUP BY MARITAL_STATUS, WORKCLASS, EDUCATION;"

result_married_private_sector_masters=pd.read_sql_query(sql_select, conn)
print('\nResult for married, working in private sector and having a masters degree:')
result_married_private_sector_masters
```

```
Result for married, working in private sector and having a masters degree:
```

Out[13]:

	MARITAL_STATUS	WORKCLASS	EDUCATION	COUNT(*)
0	Married-civ-spouse	Private	Masters	531
1	Married-spouse-absent	Private	Masters	9

In [14]:

```
# Solution -2
sql_select = "SELECT 'Married' AS MARITAL_STATUS, WORKCLASS, EDUCATION, COUNT(*) FROM A
DULTS "
sql_select = sql_select + " WHERE MARITAL_STATUS like ' Married%' AND WORKCLASS = ' Pri
vate' AND EDUCATION = ' Masters'"
sql_select = sql_select + " GROUP BY WORKCLASS, EDUCATION;"

result_married_private_sector_masters=pd.read_sql_query(sql_select, conn)
print('\nResult for married, working in private sector and having a masters degree:')
result_married_private_sector_masters
```

```
Result for married, working in private sector and having a masters degree:
```

Out[14]:

	MARITAL_STATUS	WORKCLASS	EDUCATION	COUNT(*)
0	Married	Private	Masters	540

5. What is the average, minimum and maximum age group for people working in different sectors

In [15]:

```
# average, minimum and maximum age group for people working in different sectors
sql_select = "SELECT WORKCLASS, AVG(AGE) , MIN(AGE), MAX(AGE) FROM ADULTS GROUP BY WORK
CLASS;"

result_people_diff_sector=pd.read_sql_query(sql_select, conn)
print('\nResult of average, minimum and maximum age group for people working in differe
nt sectors:')
result_people_diff_sector
```

```
Result of average, minimum and maximum age group for people working in dif
ferent sectors:
```

Out[15]:

	WORKCLASS	AVG(AGE)	MIN(AGE)	MAX(AGE)
0	?	40.960240	17	90
1	Federal-gov	42.590625	17	90
2	Local-gov	41.751075	17	90
3	Never-worked	20.571429	17	30
4	Private	36.797585	17	90
5	Self-emp-inc	46.017025	17	84
6	Self-emp-not-inc	44.969697	17	90
7	State-gov	39.436055	17	81
8	Without-pay	47.785714	19	72

6. Calculate age distribution by country

In [16]:

```
# age distribution by country
sql_select = "SELECT NATIVE_COUNTRY, AGE, COUNT(AGE) FROM ADULTS GROUP BY NATIVE_COUNTR
Y, AGE;"

result_age_distribution_by_country=pd.read_sql_query(sql_select, conn)
print('\nResult for age distribution by country:')
result_age_distribution_by_country
```

Result for age distribution by country:

Out[16]:

	NATIVE_COUNTRY	AGE	COUNT(AGE)
0	?	17	2
1	?	18	8
2	?	19	5
3	?	20	10
4	?	21	11
5	?	22	12
6	?	23	6
7	?	24	14
8	?	25	11
9	?	26	18
10	?	27	15
11	?	28	19
12	?	29	12
13	?	30	19
14	?	31	18
15	?	32	17
16	?	33	13
17	?	34	24
18	?	35	18
19	?	36	23
20	?	37	22
21	?	38	20
22	?	39	19
23	?	40	12
24	?	41	22
25	?	42	24
26	?	43	14
27	?	44	10
28	?	45	17
29	?	46	15
...	...	...	...
1251	Vietnam	37	2
1252	Vietnam	38	1

	NATIVE_COUNTRY	AGE	COUNT(AGE)
1253	Vietnam	40	1
1254	Vietnam	41	1
1255	Vietnam	43	2
1256	Vietnam	44	3
1257	Vietnam	45	3
1258	Vietnam	46	1
1259	Vietnam	48	1
1260	Vietnam	50	1
1261	Vietnam	51	1
1262	Vietnam	52	1
1263	Vietnam	53	1
1264	Vietnam	54	1
1265	Vietnam	63	1
1266	Vietnam	70	1
1267	Vietnam	73	2
1268	Yugoslavia	20	1
1269	Yugoslavia	22	1
1270	Yugoslavia	25	1
1271	Yugoslavia	29	1
1272	Yugoslavia	31	1
1273	Yugoslavia	35	2
1274	Yugoslavia	36	1
1275	Yugoslavia	40	1
1276	Yugoslavia	41	2
1277	Yugoslavia	43	1
1278	Yugoslavia	45	1
1279	Yugoslavia	56	2
1280	Yugoslavia	66	1

1281 rows × 3 columns

7. Compute a new column as 'Net-Capital-Gain' from the two columns 'capital-gain' and 'capital-loss'

In [17]:

```
# Solution - 1
# Compute at Database level using SQL operation

sql_select = "SELECT ADULTS.*, (capital_gain + capital_loss) as 'Net-Capital-Gain' FROM
 ADULTS;"

result_Net_Capital_Gain=pd.read_sql_query(sql_select, conn)
print('\nResult with new column Net-Capital-Gain:')
result_Net_Capital_Gain.head(10)
```

```
Result with new column Net-Capital-Gain:
```

Out[17]:

	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
5	37	Private	284582	Masters	14	Married-civ-spouse
6	49	Private	160187	9th	5	Married-spouse-absent
7	52	Self-emp-not-inc	209642	HS-grad	9	Married-civ-spouse
8	31	Private	45781	Masters	14	Never-married
9	42	Private	159449	Bachelors	13	Married-civ-spouse

In [18]:

```
# Solution - 2
# Compute at DataFrame level using Column operation

df_adult_data['Net-Capital-Gain']=df_adult_data['capital-gain']+df_adult_data['capital-
loss']

print('\nResult with new column Net-Capital-Gain:')
df_adult_data.head(10)
```

```
Result with new column Net-Capital-Gain:
```

Out[18]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife
5	37	Private	284582	Masters	14	Married-civ-spouse	Exec-managerial	Wife
6	49	Private	160187	9th	5	Married-spouse-absent	Other-service	Not-in-family
7	52	Self-emp-not-inc	209642	HS-grad	9	Married-civ-spouse	Exec-managerial	Husband
8	31	Private	45781	Masters	14	Never-married	Prof-specialty	Not-in-family
9	42	Private	159449	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband