Problem Statement

Read the following data set:

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

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)

In [1]:
```
import numpy as np
import pandas as pd

from pandasql import sqldf


url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data
# Create a sql db from adult dataset and name it sqladb
### Create a sql db from adult dataset and name it sqladb
sqladb = pd.read_csv(url , sep=',',header=None, index_col=None)
sqladb
```

3	53	Private	234721	11th	7	civ-spouse	cleaners	Husband	Black	Male	
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	
5	37	Private	284582	Masters	14	Married-civ-spouse	Exec-managerial	Wife	White	Female	
6	49	Private	160187	9th	5	Married-spouse-absent	Other-service	Not-in-family	Black	Female	
7	52	Self-emp-not-inc	209642	HS-grad	9	Married-civ-spouse	Exec-managerial	Husband	White	Male	
8	31	Private	45781	Masters	14	Never-married	Prof-specialty	Not-in-family	White	Female	14

In [2]:
```
# Adding columns to the dataframe
sqladb.columns = ['age', 'workclass', 'fnlwgt', 'education', 'education-num',
    'marital-status', 'occupation', 'relationship', 'race', 'sex', 'capital-gain'
    'capital-loss', 'hours-per-week', 'native-country', 'income']
sqladb

# Changing columns name from hyphen ("-") to underscore ("_") as "-" doesn't work
```

						spouse			
5	37	Private	284582	Masters	14	Married-civ-spouse	Exec-managerial	Wife	Whi
6	49	Private	160187	9th	5	Married-spouse-absent	Other-service	Not-in-family	Bla
7	52	Self-emp-not-inc	209642	HS-grad	9	Married-civ-spouse	Exec-managerial	Husband	Whi
8	31	Private	45781	Masters	14	Never-married	Prof-specialty	Not-in-family	Whi
9	42	Private	159449	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	Whi
10	37	Private	280464	Some-college	10	Married-civ-spouse	Exec-managerial	Husband	Bla

Data Preprocessing

In [3]:
```
# Triming all the strings to remove whitespaces from R.H.S. and L.H.S.
def trimAllColumns(df):
    trimStrings = lambda x : x.strip() if type(x) is str else x
    return df.applymap(trimStrings)
sqladb = trimAllColumns(sqladb)

# Replace all the columns having hyphen ("-")  to underscore ("_") as "-" columns
sqladb.columns = sqladb.columns.str.replace("-","_")
sqladb
```

[illegible]	[illegible]	[illegible]	[illegible]	acdm	[illegible]	[illegible]	[illegible]	[illegible]
14	40	Private	121772	Assoc-voc	11	Married-civ-spouse	Craft-repair	Husbar
15	34	Private	245487	7th-8th	4	Married-civ-spouse	Transport-moving	Husbar
16	25	Self-emp-not-inc	176756	HS-grad	9	Never-married	Farming-fishing	Own-chi
17	32	Private	186824	HS-grad	9	Never-married	Machine-op-inspct	Unmarrie
18	38	Private	28887	11th	7	Married-civ-spouse	Sales	Husbar
19	43	Self-emp-not-inc	292175	Masters	14	Divorced	Exec-managerial	Unmarrie

In [4]:
```
#pandasql uses SQLite syntax.
#Any pandas dataframes will be automatically detected by pandasql.
#You can query them as you would any regular SQL table.

# define helper function for pandas SQL
# You can pass locals()/globals() to pandasql when executing a SQL statement
pysqldf = lambda q: sqldf(q, globals())
pysqldf
```

Out[4]:
```
<function __main__.<lambda>>
```

Select 10 records from the adult sqladb

In [5]:
```
pysqldf("SELECT * FROM sqladb LIMIT 10;") # selecting 10 records
```

Out[5]:

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

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

In [6]:
```
pysqldf("SELECT sex as Sex, workclass as 'Work_Class', avg(hours_per_week) as Ave
```

Out[6]:

	Sex	Work_Class	Average_Hours_Week_Private
0	Male	Private	42.221226

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

In [7]:
```
pysqldf("SELECT education as Education, count(education) as Education_Count FROM
```

Out[7]:

	Education	Education_Count
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 [8]:
```
pysqldf("SELECT occupation as Occupation, count(occupation) as Occupation_Count F
```

Out[8]:

	Occupation	Occupation_Count
0	?	1843
1	Adm-clerical	3770
2	Armed-Forces	9
3	Craft-repair	4099
4	Exec-managerial	4066
5	Farming-fishing	994
6	Handlers-cleaners	1370
7	Machine-op-inspct	2002
8	Other-service	3295
9	Priv-house-serv	149
10	Prof-specialty	4140
11	Protective-serv	649
12	Sales	3650
13	Tech-support	928
14	Transport-moving	1597

In [9]:
```
pysqldf("SELECT relationship as Relationship, count(relationship) as Relationship_
```

Out[9]:

	Relationship	Relationship_Count
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 [10]:
```
# Both the query will work
pysqldf("SELECT * FROM sqladb where education = 'Masters' and workclass='Private'
pysqldf("SELECT * FROM sqladb where education = 'Masters' and workclass='Private'
```

Out[10]:

	age	workclass	fnlwgt	education	education_num	marital_status	occupation	relationship
0	37	Private	284582	Masters	14	Married-civ-spouse	Exec-managerial	Wife
1	33	Private	202051	Masters	14	Married-civ-spouse	Prof-specialty	Husband
2	76	Private	124191	Masters	14	Married-civ-spouse	Exec-managerial	Husband
3	31	Private	99928	Masters	14	Married-civ-spouse	Prof-specialty	Wife
4	35	Private	138992	Masters	14	Married-civ-spouse	Prof-specialty	Other-relative
5	34	Private	142897	Masters	14	Married-civ-spouse	Exec-managerial	Husband
6	62	Private	270092	Masters	14	Married-civ-	Prof-	Husband

In [11]:
```
# solution of 4 as a count
pysqldf("SELECT education as Education, workclass as Work_Class, count(education)
```

Out[11]:

	Education	Work_Class	No_Of_Masters_WorkClass
0	Masters	Private	894

In [12]:
```
#pysqldf("SELECT distinct (education) FROM sqladb ;")
```

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

In [13]:
```
df(" select occupation, avg(age) as Average_Age, min(age) as Minimum_Age, max(age)
```

Out[13]:

	occupation	Average_Age	Minimum_Age	Maximum_Age
0	?	40.882800	17	90
1	Adm-clerical	36.964456	17	90
2	Armed-Forces	30.222222	23	46
3	Craft-repair	39.031471	17	90
4	Exec-managerial	42.169208	17	90
5	Farming-fishing	41.211268	17	90
6	Handlers-cleaners	32.165693	17	90
7	Machine-op-inspct	37.715285	17	90
8	Other-service	34.949621	17	90
9	Priv-house-serv	41.724832	17	81
10	Prof-specialty	40.517633	17	90
11	Protective-serv	38.953775	17	90
12	Sales	37.353973	17	90
13	Tech-support	37.022629	17	73
14	Transport-moving	40.197871	17	90

6. Calculate age distribution by country

In [14]:
```
ve_country as Native_Country, count(age) as Age_Count FROM sqladb where native_cour
```

Out[14]:

	Age	Native_Country	Age_Count
0	18	Cambodia	1
1	25	Cambodia	1
2	27	Cambodia	2
3	28	Cambodia	1
4	32	Cambodia	1
5	34	Cambodia	1
6	35	Cambodia	1
7	36	Cambodia	1
8	37	Cambodia	2
9	40	Cambodia	2
10	42	Cambodia	1
11	46	Cambodia	1
12	48	Cambodia	1
13	50	Cambodia	1
14	51	Cambodia	1
15	65	Cambodia	1
16	17	Canada	2
17	18	Canada	1
18	19	Canada	1
19	20	Canada	2
20	22	Canada	1
21	23	Canada	3
22	24	Canada	3
23	25	Canada	5
24	26	Canada	2
25	27	Canada	2
26	28	Canada	3
27	29	Canada	4
28	30	Canada	3
29	31	Canada	2
...	...	...	...
1192	37	Vietnam	2
1193	38	Vietnam	1

	Age	Native_Country	Age_Count
1194	40	Vietnam	1
1195	41	Vietnam	1
1196	43	Vietnam	2
1197	44	Vietnam	3
1198	45	Vietnam	3
1199	46	Vietnam	1
1200	48	Vietnam	1
1201	50	Vietnam	1
1202	51	Vietnam	1
1203	52	Vietnam	1
1204	53	Vietnam	1
1205	54	Vietnam	1
1206	63	Vietnam	1
1207	70	Vietnam	1
1208	73	Vietnam	2
1209	20	Yugoslavia	1
1210	22	Yugoslavia	1
1211	25	Yugoslavia	1
1212	29	Yugoslavia	1
1213	31	Yugoslavia	1
1214	35	Yugoslavia	2
1215	36	Yugoslavia	1
1216	40	Yugoslavia	1
1217	41	Yugoslavia	2
1218	43	Yugoslavia	1
1219	45	Yugoslavia	1
1220	56	Yugoslavia	2
1221	66	Yugoslavia	1

1222 rows × 3 columns

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

In [15]:
```
pysqldf(" select  capital_gain -capital_loss as Net_Capital_Gain, *  FROM sqladb
```

Out[15]:

	Net_Capital_Gain	age	workclass	fnlwgt	education	education_num	marital_status	occu
0	2174	39	State-gov	77516	Bachelors	13	Never-married	
1	0	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	mar
2	0	38	Private	215646	HS-grad	9	Divorced	Ha c
3	0	53	Private	234721	11th	7	Married-civ-spouse	Ha c
4	0	28	Private	338409	Bachelors	13	Married-civ-spouse	s
5	0	37	Private	284582	Masters	14	Married-civ-spouse	mar
6	0	49	Private	160187	9th	5	Married-spouse-absent	