Import e Dataset

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

In [2]:
```
dataset=pd.read_csv('data_base/census.csv')
```

In [3]:
```
dataset.head(10)
```

Out[3]:

	age	workclass	final-weight	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	\

In [4]:
```
dataset.shape
```

Out[4]:
```
(32561, 15)
```

In [5]:
```
type(dataset)
```

Out[5]:
```
pandas.core.frame.DataFrame
```

In [6]:
```
dataset.tail
```

Out[6]:
```
<bound method NDFrame.tail of        age          workclass  final-weight    ed
ucation  education-num  \
0       39          State-gov         77516    Bachelors             13
1       50   Self-emp-not-inc         83311    Bachelors             13
2       38            Private        215646      HS-grad              9
3       53            Private        234721         11th              7
4       28            Private        338409    Bachelors             13
...    ...                ...           ...          ...            ...
32556   27            Private        257302   Assoc-acdm             12
32557   40            Private        154374      HS-grad              9
32558   58            Private        151910      HS-grad              9
32559   22            Private        201490      HS-grad              9
32560   52       Self-emp-inc        287927      HS-grad              9

            marital-status          occupation    relationship    race  \
0            Never-married        Adm-clerical   Not-in-family   White
1       Married-civ-spouse     Exec-managerial         Husband   White
2                 Divorced   Handlers-cleaners   Not-in-family   White
3       Married-civ-spouse   Handlers-cleaners         Husband   Black
4       Married-civ-spouse      Prof-specialty            Wife   Black
...                    ...                 ...             ...     ...
32556   Married-civ-spouse        Tech-support            Wife   White
32557   Married-civ-spouse   Machine-op-inspct         Husband   White
32558              Widowed        Adm-clerical       Unmarried   White
32559        Never-married        Adm-clerical       Own-child   White
32560   Married-civ-spouse     Exec-managerial            Wife   White

           sex  capital-gain  capital-loos  hour-per-week  native-country  \
0         Male          2174             0             40   United-States
1         Male             0             0             13   United-States
2         Male             0             0             40   United-States
3         Male             0             0             40   United-States
4       Female             0             0             40            Cuba
...        ...           ...           ...            ...             ...
32556   Female             0             0             38   United-States
32557     Male             0             0             40   United-States
32558   Female             0             0             40   United-States
32559     Male             0             0             20   United-States
32560   Female         15024             0             40   United-States

       income
0       <=50K
1       <=50K
2       <=50K
3       <=50K
4       <=50K
...       ...
32556   <=50K
32557    >50K
32558   <=50K
32559   <=50K
32560    >50K

[32561 rows x 15 columns]>
```

Amostragem Aleatoria

Trata-se de realizar a aquisição de n amostras de modo aleatório via função sample do pandas

In [7]:
```
df_amostra_aleatoria_simples = dataset.sample(n = 100,random_state=1)#desmarcar
df_amostra_aleatoria_simples
```

Out[7]:

	age	workclass	final-weight	education	education-num	marital-status	occupation	relations
9646	62	Self-emp-not-inc	26911	7th-8th	4	Widowed	Other-service	Not far
709	18	Private	208103	11th	7	Never-married	Other-service	Oth rela
7385	25	Private	102476	Bachelors	13	Never-married	Farming-fishing	Own-c
16671	33	Private	511517	HS-grad	9	Married-civ-spouse	Prof-specialty	Husb
21932	36	Private	292570	11th	7	Never-married	Machine-op-inspct	Unmar
...	...	...	...	...	...	...	...	
27578	64	?	200017	HS-grad	9	Widowed	?	Not far
2544	19	?	192773	Some-college	10	Never-married	?	Own-c
2486	75	?	164849	9th	5	Married-civ-spouse	?	Husb
13143	28	Private	154863	Bachelors	13	Never-married	Adm-clerical	Own-c
551	36	Self-emp-not-inc	27053	HS-grad	9	Separated	Other-service	Unmar

100 rows × 15 columns

In [8]:
```
df_amostra_aleatoria_simples.shape
```

Out[8]: (100, 15)

In [9]:
```
df_amostra_aleatoria_simples
```

Out[9]:

	age	workclass	final-weight	education	education-num	marital-status	occupation	relations
9646	62	Self-emp-not-inc	26911	7th-8th	4	Widowed	Other-service	Not far
709	18	Private	208103	11th	7	Never-married	Other-service	Oth rela
7385	25	Private	102476	Bachelors	13	Never-married	Farming-fishing	Own-c
16671	33	Private	511517	HS-grad	9	Married-civ-spouse	Prof-specialty	Husb
21932	36	Private	292570	11th	7	Never-married	Machine-op-inspct	Unmar
...	...	...	...	...	...	...	...	
27578	64	?	200017	HS-grad	9	Widowed	?	Not far
2544	19	?	192773	Some-college	10	Never-married	?	Own-c
2486	75	?	164849	9th	5	Married-civ-spouse	?	Husb
13143	28	Private	154863	Bachelors	13	Never-married	Adm-clerical	Own-c
551	36	Self-emp-not-inc	27053	HS-grad	9	Separated	Other-service	Unmar

100 rows × 15 columns

In [10]:
```
def get_random_sample(dataset,n_samples,fixed):
    return dataset.sample(n=n_samples,random_state=fixed)
```

In [11]:
```
amostra=get_random_sample(dataset,100,None) #none deixa como random_state vazio
```

In [12]:
```
amostra.head(10)
```

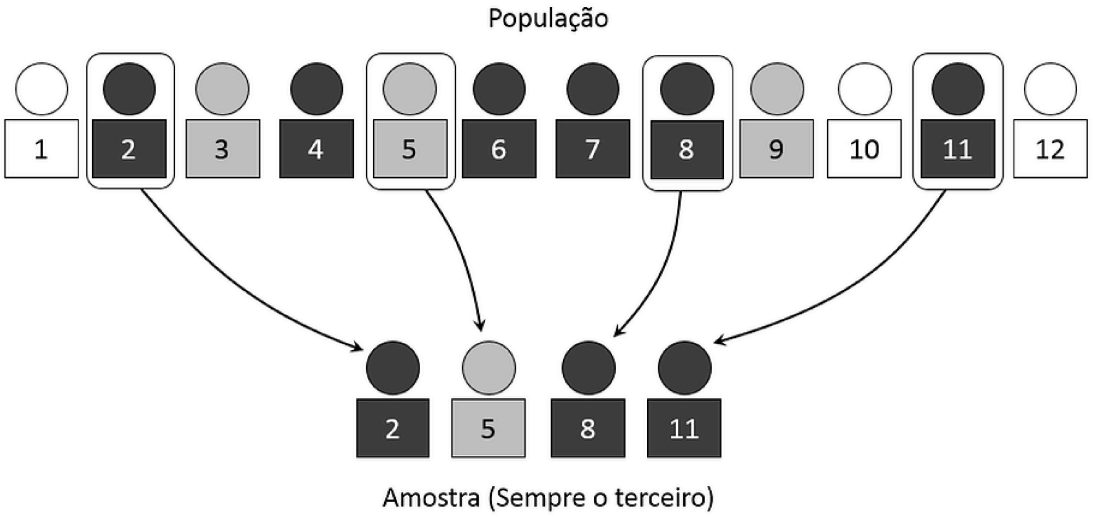
Out[12]:

	age	workclass	final-weight	education	education-num	marital-status	occupation	relationsh
7537	25	Private	104439	Some-college	10	Never-married	Adm-clerical	Not-i fam
16627	37	Self-emp-not-inc	265266	Prof-school	15	Married-civ-spouse	Prof-specialty	Husba
2318	62	State-gov	33142	Masters	14	Married-civ-spouse	Prof-specialty	Husba
6873	73	Self-emp-not-inc	143833	12th	8	Married-civ-spouse	Farming-fishing	Husba
14170	49	Private	28791	Some-college	10	Divorced	Other-service	Not-i fam
17723	18	?	115258	11th	7	Never-married	?	Own-ch
13405	29	Private	104256	HS-grad	9	Divorced	Sales	Not-i fam
14977	36	Private	225399	HS-grad	9	Married-civ-spouse	Craft-repair	Husba
14108	19	?	175495	HS-grad	9	Never-married	?	Own-ch
3817	51	Self-emp-inc	304955	Masters	14	Married-civ-spouse	Exec-managerial	Husba

Amostragem Sistemica

Na amostragem sistemica temos a divisão do total de itens pelo número de amostras:

32561 é o total de amostras, logo se quisermos realizar um grupo de 100 amostras com aquisição sistêmica podemos fixar um intervarlo, sortear um numero aleatório e pegar amostras com steps de 325, similar a figura abaixo



In [ ]:

In [13]:
```
def amostragem_sistematica(dataset, amostras):
    intervalo = len(dataset) // amostras #32561//100 o // serve para  pegar o in
    random.seed(1)
    inicio = random.randint(0, intervalo) # seleciona um valor entre 0 e 325
    indices = np.arange(inicio, len(dataset), step = intervalo)#vamos do inicio
    amostra_sistematica = dataset.iloc[indices]
    return amostra_sistematica
```

In [14]:
```
df_amostra_sistematica = amostragem_sistematica(dataset, 100)
df_amostra_sistematica.shape
```

Out[14]: (100, 15)

In [15]:
```
32561//100
```

Out[15]: 325

Amostragem por Grupos

Divide em intervalos de grupos, por exemplo temos 30 amostras e iremos fazer 5 grupos, logo cada grupo terá 6 amostras.

In [16]:
```
def amostragem_grupos(dataset,numgroup):
    select=random.randint(1,numgroup) #pega um dos grupos
    num_amostra=int(mth.floor(dataset.shape[0]/numgroup)) #numero de amostras ar
    offset=(select-1)*num_amostra
    return dataset.iloc[offset:offset+num_amostra]
print(dataset.shape[0]/20)
```

```
1628.05
```

In [17]:
```
dataset.iloc[5:5+8]
```

Out[17]:

	age	workclass	final-weight	education	education-num	marital-status	occupation	relationship
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
10	37	Private	280464	Some-college	10	Married-civ-spouse	Exec-managerial	Husband
11	30	State-gov	141297	Bachelors	13	Married-civ-spouse	Prof-specialty	Husband
12	23	Private	122272	Bachelors	13	Never-married	Adm-clerical	Own-child

In [18]:
```
df_aux=dataset.iloc[0:0+8]
```

In [19]:
```
select=random.randint(1,3)
select
```

Out[19]: 3

In [20]:
```
num_amostra=int(np.ceil(df_aux.shape[0]/3))
num_amostra
```

Out[20]: 3

In [21]:
```
offset=(select-1)*num_amostra
offset
```

Out[21]: 6

In [22]:
```
df_aux.iloc[offset:offset+num_amostra]
```

Out[22]:

	age	workclass	final-weight	education	education-num	marital-status	occupation	relationship
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

In [23]:
```
df_group_sample=amostragem_grupos(dataset,1000)
```

In [24]:
```
df_group_sample.shape
```

Out[24]: (32, 15)

In [25]:
```
df_group_sample
```

Out[25]:

	age	workclass	final-weight	education	education-num	marital-status	occupation	relations
27744	37	Private	231491	HS-grad	9	Married-civ-spouse	Craft-repair	Husb
27745	36	Self-emp-not-inc	239415	HS-grad	9	Married-civ-spouse	Farming-fishing	Husb
27746	38	Private	179262	Some-college	10	Divorced	Adm-clerical	Unmar
27747	72	Without-pay	121004	HS-grad	9	Married-civ-spouse	Other-service	Husb
27748	40	Private	252392	5th-6th	3	Married-civ-spouse	Other-service	Husb
27749	19	Private	163578	Some-college	10	Never-married	Sales	Own-c
27750	55	Private	143266	Assoc-voc	11	Married-civ-spouse	Craft-repair	Husb
27751	30	Private	285902	Assoc-acdm	12	Married-civ-spouse	Craft-repair	Husb
27752	52	Private	113094	Bachelors	13	Separated	Adm-clerical	Unmar
27753	29	Private	278637	Bachelors	13	Married-civ-spouse	Sales	Husb
27754	41	Private	174540	Some-college	10	Married-civ-spouse	Prof-specialty	Husb
27755	29	Private	188729	Bachelors	13	Never-married	Adm-clerical	Not fa
27756	24	Private	72143	Bachelors	13	Never-married	Sales	Not fa
27757	46	Self-emp-not-inc	328216	Prof-school	15	Married-civ-spouse	Prof-specialty	Husb
27758	44	Private	165815	HS-grad	9	Married-civ-spouse	Exec-managerial	Husb
27759	17	Private	317702	10th	6	Never-married	Sales	Own-c

	age	workclass	final-weight	education	education-num	marital-status	occupation	relations
27760	35	Private	215323	Assoc-voc	11	Divorced	Other-service	Unmar
27761	38	Private	192939	HS-grad	9	Divorced	Craft-repair	Unmar
27762	36	Private	156352	9th	5	Never-married	Handlers-cleaners	Own-c
27763	24	Private	155066	11th	7	Married-civ-spouse	Machine-op-inspct	Husb
27764	38	Self-emp-not-inc	152621	Bachelors	13	Married-civ-spouse	Farming-fishing	Husb
27765	19	Private	298891	11th	7	Never-married	Sales	Not fa
27766	30	Private	193298	HS-grad	9	Married-civ-spouse	Transport-moving	Husb
27767	36	Local-gov	150309	Assoc-voc	11	Married-civ-spouse	Transport-moving	Husb
27768	27	Private	384308	Some-college	10	Never-married	Craft-repair	Not fa
27769	27	Private	305647	Assoc-acdm	12	Married-civ-spouse	Adm-clerical	Husb
27770	66	?	182378	7th-8th	4	Married-civ-spouse	?	Husb
27771	65	Federal-gov	23494	Some-college	10	Married-civ-spouse	Exec-managerial	Husb
27772	37	Private	421633	Masters	14	Never-married	Exec-managerial	Not fa
27773	17	Private	57723	11th	7	Never-married	Sales	Own-c
27774	19	?	307837	Some-college	10	Never-married	?	Own-c
27775	57	Private	103540	5th-6th	3	Married-civ-spouse	Transport-moving	Husb

In [26]:
```
dataset.tail(5)
```

Out[26]:

	age	workclass	final-weight	education	education-num	marital-status	occupation	relationsl
32556	27	Private	257302	Assoc-acdm	12	Married-civ-spouse	Tech-support	W
32557	40	Private	154374	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husba
32558	58	Private	151910	HS-grad	9	Widowed	Adm-clerical	Unmarr
32559	22	Private	201490	HS-grad	9	Never-married	Adm-clerical	Own-ch
32560	52	Self-emp-inc	287927	HS-grad	9	Married-civ-spouse	Exec-managerial	W

In [ ]:

Amostragem Estratificada

A ideia é que uma população tenha divisão entre tipos de pessoas, logo pegamos uma portagem referente ao valor de cada categoria exemplo 100 pessoas, 60 mulheres e 40 homens, se fizermos amostragem aleatoria tem a chance de pegarmos mais mulheres, logo a estratificada mantém a propoção pagaremos 10% das pessoas, 10 delas com 4 homens e 6 mulheres

In [27]:
```
from sklearn.model_selection import StratifiedShuffleSplit
dataset['income']==' >50K'
```

Out[27]:
```
0        False
1        False
2        False
3        False
4        False
         ...
32556    False
32557     True
32558    False
32559    False
32560     True
Name: income, Length: 32561, dtype: bool
```

In [28]:
```
str(dataset['income'][0])
```

Out[28]:
```
' <=50K'
```

In [29]:
```
def amostragem_estratificada(dataset, num_amostra):
    amostra_more_50k=(dataset[dataset['income']==' >50K'].shape[0]/dataset.shape
    amostra_less_50k=(dataset[dataset['income']==' <=50K'].shape[0]/dataset.shap
```

```
    more_50k=dataset[dataset['income']==' >50K']
    less_50k=dataset[dataset['income']==' <=50K']

    return pd.concat([more_50k.sample(round(amostra_more_50k)),less_50k.sample(r
df_estratificada=amostragem_estratificada(dataset,100)
```

In [30]:
```
aux1=dataset[dataset['income']==' >50K'][1:11]
aux2=dataset[dataset['income']==' <=50K'][1:11]
res=pd.concat([aux1,aux2])
res['income'].value_counts()
```

Out[30]:
```
 >50K     10
 <=50K    10
Name: income, dtype: int64
```

In [31]:
```
df_estratificada['income'].value_counts()
```

Out[31]:
```
 <=50K    76
 >50K     24
Name: income, dtype: int64
```

Outro meio

In [32]:
```
split = StratifiedShuffleSplit(test_size=100/dataset.shape[0])
for x, y in split.split(dataset, dataset['income']):
    df_x = dataset.iloc[x]
    df_y = dataset.iloc[y]
```

In [33]:
```
df_x
```

Out[33]:

	age	workclass	final-weight	education	education-num	marital-status	occupation	relations
14332	53	Private	158746	Some-college	10	Married-civ-spouse	Tech-support	Husb
1321	20	Private	114874	Some-college	10	Never-married	Adm-clerical	Own-c
21020	31	Private	219619	HS-grad	9	Never-married	Sales	Oth rela
4638	66	State-gov	41506	10th	6	Divorced	Other-service	Not far
23143	26	Private	456618	HS-grad	9	Never-married	Other-service	Not far
...	...	...	...	...	...	...	...	
17513	46	Federal-gov	78022	HS-grad	9	Married-civ-spouse	Adm-clerical	Husb
13936	31	Private	369825	7th-8th	4	Never-married	Handlers-cleaners	Oth rela
13055	58	Private	216941	HS-grad	9	Divorced	Exec-managerial	Unmar
17720	25	Private	185942	Masters	14	Never-married	Prof-specialty	Own-c
11603	35	Private	108907	HS-grad	9	Separated	Machine-op-inspct	Unmar

32461 rows × 15 columns

In [34]:
```
df_y
```

Out[34]:

	age	workclass	final-weight	education	education-num	marital-status	occupation	relationsl
15102	76	Local-gov	169133	Some-college	10	Widowed	Adm-clerical	Not-fam
16769	23	State-gov	215443	Some-college	10	Never-married	Adm-clerical	Not-fam
26391	29	State-gov	106972	Masters	14	Never-married	Prof-specialty	Not-fam
27882	41	Private	216461	Some-college	10	Divorced	Adm-clerical	Own-ch
6439	80	Private	249983	7th-8th	4	Widowed	Other-service	Not-fam
...	...	...	...	...	...	...	...	
1024	23	Private	129042	HS-grad	9	Never-married	Machine-op-inspct	Unmarr
31957	26	Private	473625	HS-grad	9	Married-civ-spouse	Other-service	W
28923	26	Local-gov	197764	Bachelors	13	Never-married	Prof-specialty	Not-fam
17585	42	Private	345363	HS-grad	9	Divorced	Exec-managerial	Not-fam
25670	29	Private	130438	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husba

100 rows × 15 columns

Amostragem Por Reservatório

Utilizada para base de dados não estática, como e-commerce e itens de tamanho desconhecidos. Cada item deve possuir a mesma probabilidade de ser selecionado

In [35]:
```
def amostragem_reservatorio(dataset, amostras):
    stream = []
    for i in range(len(dataset)):
        stream.append(i)

    i = 0
    tamanho = len(dataset)

    reservatorio = [0] * amostras
    for i in range(amostras):
        reservatorio[i] = stream[i]

    while i < tamanho: #aqui ele vai percorrendo e trocando o valor
```

```
        j = random.randrange(i + 1)
        if j < amostras:
            reservatorio[j] = stream[i]
        i += 1

    return dataset.iloc[reservatorio]
```

In [36]:
```
df_reservatorio=amostragem_reservatorio(dataset,100)
```

In [37]:
```
df_reservatorio
```

Out[37]:

	age	workclass	final-weight	education	education-num	marital-status	occupation	relationsh
29608	41	Self-emp-inc	114580	Prof-school	15	Married-civ-spouse	Exec-managerial	Wi
21696	37	Federal-gov	329088	HS-grad	9	Married-civ-spouse	Adm-clerical	Husbar
30676	42	Private	355728	Assoc-voc	11	Never-married	Craft-repair	Not-i fam
28550	43	Private	110970	Bachelors	13	Married-civ-spouse	Prof-specialty	Husbar
8768	52	Federal-gov	221532	Bachelors	13	Married-civ-spouse	Protective-serv	Husbar
...	...	...	...	...	...	...	...	
8495	47	Self-emp-not-inc	159869	Doctorate	16	Married-civ-spouse	Craft-repair	Husbar
11236	31	Private	96480	HS-grad	9	Never-married	Adm-clerical	Not-i fam
29452	38	Private	189916	Some-college	10	Married-civ-spouse	Sales	Wi
23275	20	Private	380544	Assoc-acdm	12	Never-married	Transport-moving	Own-chi
25870	41	Private	43945	7th-8th	4	Married-civ-spouse	Craft-repair	Husbar

100 rows × 15 columns

In [38]:
```
import pandas as pd
import numpy as np

# Criar um DataFrame fictício
```

```
np.random.seed(42)
num_dados = 1000
dados = {
    'tempo': np.arange(num_dados),
    'tensao': np.random.random(num_dados),
    'corrente': np.random.random(num_dados)
}
df = pd.DataFrame(dados)
display(df)
# Converter a coluna 'tempo' de segundos para dias
df['tempo'] = df['tempo'] // (24 * 60 * 60)  # Convertendo segundos para dias

# Agrupar por dia e calcular a média para cada grupo
df_agrupado = df.groupby('tempo').mean().reset_index()

# Renomear a coluna 'tempo' para 'dia' (opcional)
df_agrupado = df_agrupado.rename(columns={'tempo': 'dia'})

# Visualizar o DataFrame resultante
print(df_agrupado)
```

	tempo	tensao	corrente
0	0	0.374540	0.185133
1	1	0.950714	0.541901
2	2	0.731994	0.872946
3	3	0.598658	0.732225
4	4	0.156019	0.806561
...	...	...	...
995	995	0.091582	0.656955
996	996	0.917314	0.956615
997	997	0.136819	0.068958
998	998	0.950237	0.057055
999	999	0.446006	0.282187

1000 rows × 3 columns

```
   dia    tensao  corrente
0    0  0.490257  0.507017
```

In [ ]: