In [1]:
```
import os
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
from matplotlib.pylab import rcParams
rcParams['figure.figsize'] = 15, 5
```

Classification non supervisée :

clustering

Data set Base_freq

Ce jeu de données contient, entre autres, le nombre de sinistres, l'âge du conducteur et la classe du véhicule (Small, Medium et Large) et le bonus de chaque conducteur.

Nous allons appliquer les differentes méthodes de clustering pour établir des individus avec characteristiques similaires qui peuvent donner un profil de risque.

Classification ascendante hiérarchique CAH:

Le principe est regrouper des données en clusters similaires. Tout d'abord on considere chaque observations comme un cluster individuel. En suite, on va calculer la distance entre les observation en utilisant les notion de l'inertie intra-classe et l'inertie inter-classe.

inertie intra-classe:

Mesure la distance entre les observation. En creant des cluster à l'aide du clacul de la distance auclidienne d^2 = ||x1-x2||^2

inertie inter-classe: Mesure la distance entre les cluster (definis dans l'étape précedente) à l'aide de la distance de Ward dw^2 = (n1*n2/n1*n2)||x1-x2||^2

In [2]:
```
os.chdir("C:\\Users\\IDEAPAD5\\Documents\\Archivos Alejo\\Alejo\\Docs estudio y material clase\\Estudio U\\Mate
```

In [3]:
```
os.getcwd()
```

Out[3]:
```
'C:\\Users\\IDEAPAD5\\Documents\\Archivos Alejo\\Alejo\\Docs estudio y material clase\\Estudio U\\Material de c
lases\\Montpellier\\DU Big Data\\Econometrie\\Data'
```

In [4]:
```
base = pd.read_csv("base_freq.csv", delimiter = ";") ; base.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99997 entries, 0 to 99996
Data columns (total 17 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   PolNum      99997 non-null  int64
 1   CalYear     99997 non-null  int64
 2   Gender      99997 non-null  object
 3   Type        99997 non-null  object
 4   Category    99997 non-null  object
 5   Occupation  99997 non-null  object
 6   Age         99997 non-null  float64
 7   Group1      99997 non-null  int64
 8   Bonus       99997 non-null  int64
 9   Poldur      99997 non-null  float64
 10  Value       99997 non-null  float64
 11  Adind       99997 non-null  int64
 12  Group2      99997 non-null  object
 13  Density     99997 non-null  float64
 14  Expdays     99997 non-null  int64
 15  nb_sin      99997 non-null  float64
 16  chg_sin     99997 non-null  float64
dtypes: float64(6), int64(6), object(5)
memory usage: 13.0+ MB
```

In [5]:
```
base.replace({"Male":"H", "Female":"F"}, inplace=True) #Remplacement des modalités
```

In [6]:
```
df = base.copy()
df['Age_10'] = round(df.Age/5,0)*5 # simplificaiton pour affichage plus claire dans le graphique (mais à éviter
df.Age_10.value_counts()
```

Out[6]:
```
40.0    12864
35.0    12631
30.0    11944
45.0    11365
25.0    10677
50.0     9261
20.0     9102
55.0     7247
60.0     5374
65.0     4023
70.0     3552
75.0     1957
Name: Age_10, dtype: int64
```

In [7]:
```
granularité = ['Age_10','Gender']
agreg_base = pd.pivot_table(df,
                            values=['chg_sin','nb_sin','Bonus'],
                            index=granularité,
                            aggfunc=np.mean)
agreg_base.head()
```

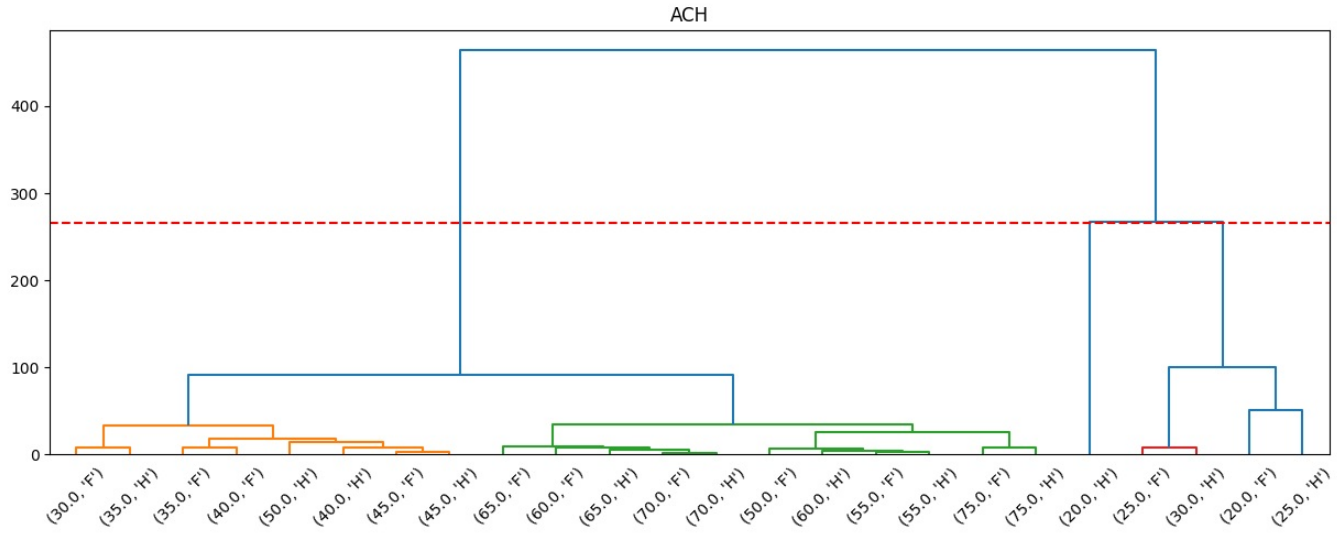
Out[7]:

		Bonus	chg_sin	nb_sin
Age_10	**Gender**			
20.0	**F**	3.715426	189.188439	0.207979
	H	3.554848	390.386385	0.422314
25.0	**F**	6.125521	146.894794	0.185966
	H	7.005819	239.712837	0.323636
30.0	**F**	2.719222	87.615864	0.138877

In [8]:
```
import scipy.cluster.hierarchy as sch
```

In [9]:
```
from scipy.cluster.hierarchy import dendrogram, linkage
```

In [10]:
```
Z = linkage(agreg_base, method = "ward", metric="euclidean")
plt.title("ACH")
dendrogram(Z,labels=agreg_base.index, color_threshold=50,orientation="top") #'top', 'left', 'bottom', or 'right
plt.axhline(y=265, color='r', linestyle='--')
plt.show()
plt.savefig('ACH.pdf')
```




```
<Figure size 1500x500 with 0 Axes>
```

Nous pouvons vérifier l'apparition de 4 clusters. Ici, on note une séparation marquée entre les personnes d'âge avancé, celles d'âge moyen et les plus jeunes. Compte tenu des variables analysées, à savoir le nombre de sinistres, la valeur du sinistre et la qualification de chaque conducteur, on peut déduire que le profil de risque est déterminé par l'âge des individus. Il est à noter que la hauteur de la barre verticale dans le graphique est importante car elle donne une idée de la similarité entre les classes.

In [11]:
```
from sklearn.cluster import AgglomerativeClustering
```

In [12]:
```
agg_clustering = AgglomerativeClustering(n_clusters=4) # On définie le nb de clusters
```

In [13]:
```
agg_clustering.fit(agreg_base)
```

Out[13]:
```
AgglomerativeClustering(n_clusters=4)
```

In [14]:
```
agreg_base['label'] = agg_clustering.labels_
agreg_base.head()
```

Out[14]:

		Bonus	chg_sin	nb_sin	label
Age_10	**Gender**				
20.0	**F**	3.715426	189.188439	0.207979	1
	H	3.554848	390.386385	0.422314	2
25.0	**F**	6.125521	146.894794	0.185966	3
	H	7.005819	239.712837	0.323636	1
30.0	**F**	2.719222	87.615864	0.138877	0

In [15]:
```
agreg_base.groupby(['label']).mean() # Le cluster 2, composé des hommes les plus jeunes, possède le nombre de s
```

Out[15]:

	Bonus	chg_sin	nb_sin
label			
0	-15.829846	57.957339	0.090315
1	5.360622	214.450638	0.265807
2	3.554848	390.386385	0.422314
3	3.987015	143.942197	0.195047

In [16]:
```
for i in range(4):
    print(f'Cluster {i}')
    print(agreg_base[agreg_base.label == i].index.tolist())
    print() #Les individus de chaque cluster
```

```
Cluster 0
[(30.0, 'F'), (35.0, 'F'), (35.0, 'H'), (40.0, 'F'), (40.0, 'H'), (45.0, 'F'), (45.0, 'H'), (50.0, 'F'), (50.0,
'H'), (55.0, 'F'), (55.0, 'H'), (60.0, 'F'), (60.0, 'H'), (65.0, 'F'), (65.0, 'H'), (70.0, 'F'), (70.0, 'H'), (
75.0, 'F'), (75.0, 'H')]

Cluster 1
[(20.0, 'F'), (25.0, 'H')]

Cluster 2
[(20.0, 'H')]

Cluster 3
[(25.0, 'F'), (30.0, 'H')]
```

K Means – Méthode des centres mobiles

On va se servir du data set précédent afin d'utiliser les variables qui nous donnent une notion de risque, à savoir le nombre de sinistres, la valeur du sinistre et la qualification de chaque conducteur. Cela va nous permettre d'identifier des groupes d'observations ayant des caractéristiques similaires (profils de risque).

In [18]:
```
df2 = base.copy()
df2 = df.pivot_table(index=['Gender','Age'],
                     columns=[],
                     values=['Bonus','chg_sin','nb_sin'],
                     aggfunc=np.mean)
df2.head()
```

Out[18]:

		Bonus	chg_sin	nb_sin
Gender	**Age**			
F	**18.0**	-0.043478	223.673246	0.253623
	19.0	2.810734	187.163376	0.194915
	20.0	4.293059	193.610861	0.215938
	21.0	4.897959	162.002003	0.182398
	22.0	6.037500	183.579375	0.197500

La procédure de l'algorithme se déroule comme suit. Tout d'abord, il y a une partition a priori en K classes. Ensuite, une réaffectation est effectuée, consistant à déplacer les objets (observations) d'un groupe à un autre pour obtenir une partition améliorée.

Finalement, une partition unique des données est obtenue. Avec la méthode K-Means, une solution unique est obtenue pour K donné, contrairement à une hiérarchie de partitions comme on le trouve dans la CAH, par exemple.

In [ ]:
```
```