In [26]:
```
import os, pandas as pd, numpy as np, matplotlib.pyplot as pl, pickle as pkl
from sklearn import model_selection as modele, linear_model as lm, metrics as mes
import matplotlib as plt
import seaborn as sb
```

In [27]:
```
os.getcwd()
```

Out[27]:
```
'C:\\Users\\HPPC\\regession logistic'
```

In [28]:
```
os.chdir("C:\\Users\\HPPC\\regession logistic")
```

In [29]:
```
data = pd.read_excel("null.xlsx")
data.head()
```

Out[29]:

	AGE	SEXE	TDT	PAR	CHOLESTEROL	GAJ	ECG	FCMAX	ANGINE	DEPRESSION	PENT
0	40	homme	AA	140	289	0	Normal	172	Non	0.0	Ascenda
1	49	femme	DNA	160	180	0	Normal	156	Non	1.0	Pl
2	37	homme	AA	130	283	0	ST	98	Non	0.0	Ascenda
3	48	femme	ASY	138	214	0	Normal	108	Oui	1.5	Pl
4	54	homme	DNA	150	195	0	Normal	122	Non	0.0	Ascenda

In [30]:
```
df = data.copy()
print(df)
```

```
     AGE   SEXE  TDT  PAR  CHOLESTEROL  GAJ     ECG  FCMAX ANGINE  \
0     40  homme   AA  140          289    0  Normal    172    Non
1     49  femme  DNA  160          180    0  Normal    156    Non
2     37  homme   AA  130          283    0      ST     98    Non
3     48  femme  ASY  138          214    0  Normal    108    Oui
4     54  homme  DNA  150          195    0  Normal    122    Non
..   ...    ...  ...  ...          ...  ...     ...    ...    ...
913   45  homme   AT  110          264    0  Normal    132    Non
914   68  homme  ASY  144          193    1  Normal    141    Non
915   57  homme  ASY  130          131    0  Normal    115    Oui
916   57  femme   AA  130          236    0     LVH    174    Non
917   38  homme  DNA  138          175    0  Normal    173    Non

     DEPRESSION       PENTE  CŒUR
0           0.0  Ascendant     0
1           1.0        Plat     1
2           0.0  Ascendant     0
3           1.5        Plat     1
4           0.0  Ascendant     0
..          ...         ...   ...
913         1.2        Plat     1
914         3.4        Plat     1
915         1.2        Plat     1
916         0.0        Plat     1
917         0.0  Ascendant     0

[918 rows x 12 columns]
```

In [31]:
```
df.duplicated().sum()
```

Out[31]:
```
0
```

In [32]:
```
df.nunique()
```

Out[32]:
```
AGE             50
SEXE             2
TDT              4
PAR             67
CHOLESTEROL    222
GAJ              2
ECG              3
FCMAX          119
ANGINE           2
DEPRESSION      53
PENTE            3
CŒUR             2
dtype: int64
```

In [33]:
```
df.isna().sum()
```

Out[33]:
```
AGE             0
SEXE            0
TDT             0
PAR             0
CHOLESTEROL     0
GAJ             0
ECG             0
FCMAX           0
ANGINE          0
DEPRESSION      0
PENTE           0
CŒUR            0
dtype: int64
```

In [38]:
```
quant = ['AGE', 'PAR', 'CHOLESTEROL', 'FCMAX', 'DEPRESSION ']
#df.rename('DEPRESSION')
```

In [52]:
```
def recoder(df):
    for i in df.select_dtypes('object').columns:
        df[i]=df[i].astype('category').cat.codes
    return(df)
recoder(df)
```

Out[52]:

	AGE	SEXE	TDT	PAR	CHOLESTEROL	GAJ	ECG	FCMAX	ANGINE	DEPRESSION	PENTE	CŒ
0	0.75	1	0	1.06	1.45	0	1	1.26	0	0.00	0	
1	0.92	0	3	1.21	0.91	0	1	1.14	0	1.13	2	
2	0.69	1	0	0.98	1.42	0	2	0.72	0	0.00	0	
3	0.90	0	1	1.04	1.08	0	1	0.79	1	1.69	2	
4	1.01	1	3	1.13	0.98	0	1	0.89	0	0.00	0	
...	...	...	...	...	...	...	...	...	...	...	...	
913	0.84	1	2	0.83	1.33	0	1	0.96	0	1.35	2	

	AGE	SEXE	TDT	PAR	CHOLESTEROL	GAJ	ECG	FCMAX	ANGINE	DEPRESSION	PENTE	CŒ
914	1.27	1	1	1.09	0.97	1	1	1.03	0	3.83	2	
915	1.07	1	1	0.98	0.66	0	1	0.84	1	1.35	2	
916	1.07	0	0	0.98	1.19	0	0	1.27	0	0.00	2	
917	0.71	1	3	1.04	0.88	0	1	1.26	0	0.00	0	

918 rows × 12 columns

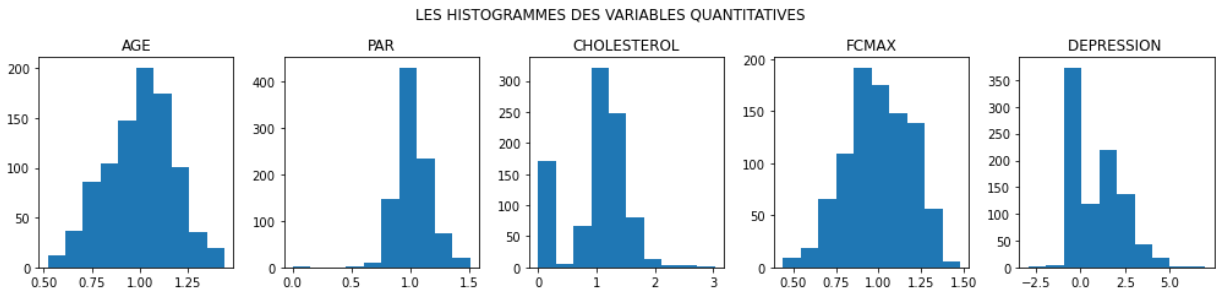
In [53]:
```
df[quant] = round(df[quant]/df[quant].mean(), 2)
```

In [54]:
```
df.head()
```

Out[54]:

	AGE	SEXE	TDT	PAR	CHOLESTEROL	GAJ	ECG	FCMAX	ANGINE	DEPRESSION	PENTE	CŒU
0	0.75	1	0	1.06	1.45	0	1	1.26	0	0.00	0	
1	0.92	0	3	1.21	0.91	0	1	1.14	0	1.13	2	
2	0.69	1	0	0.98	1.42	0	2	0.72	0	0.00	0	
3	0.90	0	1	1.04	1.08	0	1	0.79	1	1.69	2	
4	1.01	1	3	1.13	0.98	0	1	0.89	0	0.00	0	

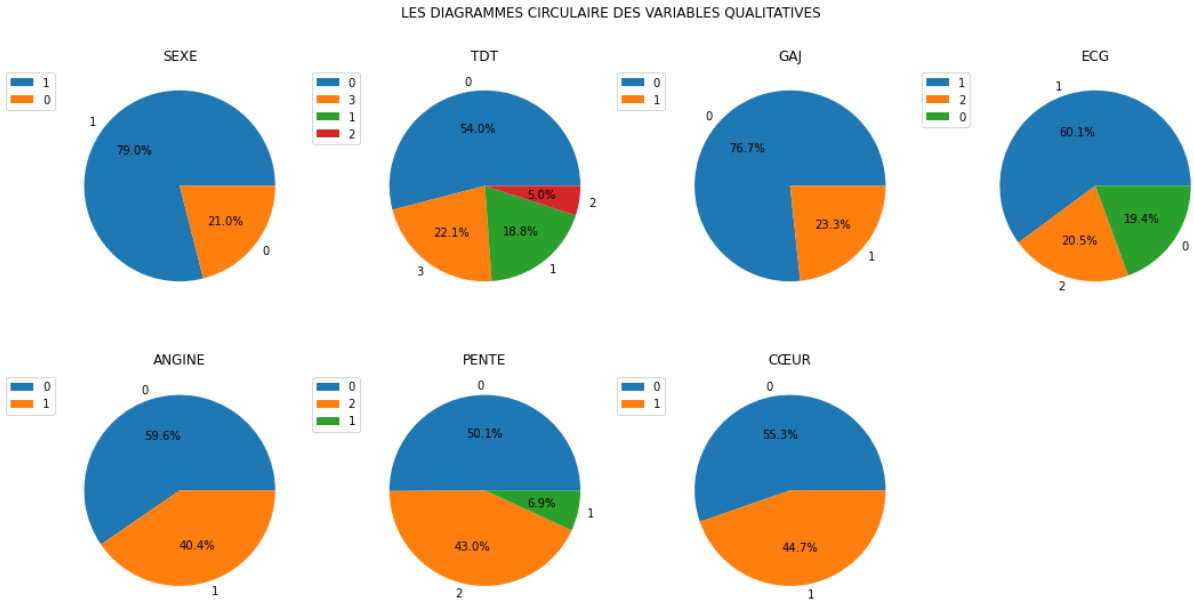
In [55]:
```
pl.figure(tight_layout = True, figsize = (14, 6))
pl.suptitle("LES HISTOGRAMMES DES VARIABLES QUANTITATIVES")
for y,x in enumerate(quant):
    pl.subplot(2,5,y+1)
    pl.hist(df[x])
    pl.title(f"{quant[y]}")
pl.show()
```



In [56]:
```
qual = ['SEXE', 'TDT', 'GAJ', 'ECG', 'ANGINE', 'PENTE', 'CŒUR'] # la liste des varia

pl.figure(tight_layout = True, figsize = (15,8))
pl.suptitle("LES DIAGRAMMES CIRCULAIRE DES VARIABLES QUALITATIVES")
for y,x in enumerate(qual):
    eff = df[x].value_counts()
    modalite = df[x].unique()
    pl.subplot(2,4,y+1)
    pl.pie(eff, labels = modalite, autopct = '%1.1f%%')
    pl.legend(bbox_to_anchor = (0, 1))
```

```
    pl.title(f"{qual[y]}")
pl.show()
```



In [57]:
```
X = df.iloc[:, 0:11]
X
```

Out[57]:

	AGE	SEXE	TDT	PAR	CHOLESTEROL	GAJ	ECG	FCMAX	ANGINE	DEPRESSION	PENTE
0	0.75	1	0	1.06	1.45	0	1	1.26	0	0.00	0
1	0.92	0	3	1.21	0.91	0	1	1.14	0	1.13	2
2	0.69	1	0	0.98	1.42	0	2	0.72	0	0.00	0
3	0.90	0	1	1.04	1.08	0	1	0.79	1	1.69	2
4	1.01	1	3	1.13	0.98	0	1	0.89	0	0.00	0
...	...	...	...	...	...	...	...	...	...	...	...
913	0.84	1	2	0.83	1.33	0	1	0.96	0	1.35	2
914	1.27	1	1	1.09	0.97	1	1	1.03	0	3.83	2
915	1.07	1	1	0.98	0.66	0	1	0.84	1	1.35	2
916	1.07	0	0	0.98	1.19	0	0	1.27	0	0.00	2
917	0.71	1	3	1.04	0.88	0	1	1.26	0	0.00	0

918 rows × 11 columns

In [58]:
```
y = data.iloc[:,11:]
y
```

Out[58]:

	CŒUR
0	0
1	1
2	0
3	1

	CŒUR
4	0
...	...
913	1
914	1
915	1
916	1
917	0

918 rows × 1 columns

In [59]:
```
X_train, X_test, Y_train, Y_test = modele.train_test_split(X, y, test_size = 0.2, ra
```

In [60]:
```
mod = lm.LogisticRegression(penalty = 'none', solver = 'newton-cg')
mod.fit(X_train, Y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConve
rsionWarning: A column-vector y was passed when a 1d array was expected. Please chan
ge the shape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
```

Out[60]:
```
LogisticRegression(penalty='none', solver='newton-cg')
```

In [61]:
```
mod.predict_proba(X_test)
```

Out[61]:
```
array([[0.35496203, 0.64503797],
       [0.16960133, 0.83039867],
       [0.03457301, 0.96542699],
       [0.0764487 , 0.9235513 ],
       [0.17768705, 0.82231295],
       [0.74672317, 0.25327683],
       [0.29597548, 0.70402452],
       [0.51735064, 0.48264936],
       [0.96541828, 0.03458172],
       [0.2826058 , 0.7173942 ],
       [0.88994162, 0.11005838],
       [0.6539992 , 0.3460008 ],
       [0.40417279, 0.59582721],
       [0.16705882, 0.83294118],
       [0.09203056, 0.90796944],
       [0.88406903, 0.11593097],
       [0.07255202, 0.92744798],
       [0.79787133, 0.20212867],
       [0.04776453, 0.95223547],
       [0.21131067, 0.78868933],
       [0.07855208, 0.92144792],
       [0.96654432, 0.03345568],
       [0.03879931, 0.96120069],
       [0.96237084, 0.03762916],
       [0.41115221, 0.58884779],
       [0.02625307, 0.97374693],
       [0.86081418, 0.13918582],
       [0.184074  , 0.815926  ],
       [0.14306526, 0.85693474],
```

```
[0.84833901, 0.15166099],
[0.98822374, 0.01177626],
[0.93968456, 0.06031544],
[0.69221562, 0.30778438],
[0.96131221, 0.03868779],
[0.10723281, 0.89276719],
[0.06521403, 0.93478597],
[0.09615314, 0.90384686],
[0.04487176, 0.95512824],
[0.94696335, 0.05303665],
[0.61414904, 0.38585096],
[0.22452249, 0.77547751],
[0.2325561 , 0.7674439 ],
[0.08042256, 0.91957744],
[0.78349444, 0.21650556],
[0.96391422, 0.03608578],
[0.986358  , 0.013642  ],
[0.11334199, 0.88665801],
[0.81843   , 0.18157   ],
[0.95221882, 0.04778118],
[0.13528833, 0.86471167],
[0.85314986, 0.14685014],
[0.07228734, 0.92771266],
[0.01298134, 0.98701866],
[0.79992191, 0.20007809],
[0.339827  , 0.660173  ],
[0.85369522, 0.14630478],
[0.07269278, 0.92730722],
[0.91619274, 0.08380726],
[0.33051036, 0.66948964],
[0.83005714, 0.16994286],
[0.01856838, 0.98143162],
[0.43551455, 0.56448545],
[0.24693029, 0.75306971],
[0.5685169 , 0.4314831 ],
[0.96279401, 0.03720599],
[0.07861632, 0.92138368],
[0.10312646, 0.89687354],
[0.18763544, 0.81236456],
[0.9679706 , 0.0320294 ],
[0.01794743, 0.98205257],
[0.03016454, 0.96983546],
[0.9105003 , 0.0894997 ],
[0.13276655, 0.86723345],
[0.07152434, 0.92847566],
[0.93459766, 0.06540234],
[0.10696642, 0.89303358],
[0.90419903, 0.09580097],
[0.16606083, 0.83393917],
[0.03020698, 0.96979302],
[0.33841854, 0.66158146],
[0.02984471, 0.97015529],
[0.05997388, 0.94002612],
[0.08251973, 0.91748027],
[0.15846023, 0.84153977],
[0.1623932 , 0.8376068 ],
[0.03220352, 0.96779648],
[0.67115575, 0.32884425],
[0.56227941, 0.43772059],
[0.89738474, 0.10261526],
[0.71542579, 0.28457421],
[0.03055251, 0.96944749],
[0.05390453, 0.94609547],
[0.92583897, 0.07416103],
```

```
       [0.44620256, 0.55379744],
       [0.07400792, 0.92599208],
       [0.15027603, 0.84972397],
       [0.07246277, 0.92753723],
       [0.1857801 , 0.8142199 ],
       [0.13377028, 0.86622972],
       [0.63356425, 0.36643575],
       [0.28735637, 0.71264363],
       [0.9659129 , 0.0340871 ],
       [0.1756883 , 0.8243117 ],
       [0.17112415, 0.82887585],
       [0.08141389, 0.91858611],
       [0.92355839, 0.07644161],
       [0.90402068, 0.09597932],
       [0.54537251, 0.45462749],
       [0.04300625, 0.95699375],
       [0.81612043, 0.18387957],
       [0.84740312, 0.15259688],
       [0.03079608, 0.96920392],
       [0.93610009, 0.06389991],
       [0.80066341, 0.19933659],
       [0.98075761, 0.01924239],
       [0.94266284, 0.05733716],
       [0.1665684 , 0.8334316 ],
       [0.98523093, 0.01476907],
       [0.63669803, 0.36330197],
       [0.12532174, 0.87467826],
       [0.18458031, 0.81541969],
       [0.44485531, 0.55514469],
       [0.06414396, 0.93585604],
       [0.96693085, 0.03306915],
       [0.00488167, 0.99511833],
       [0.76680806, 0.23319194],
       [0.05064536, 0.94935464],
       [0.84542875, 0.15457125],
       [0.0481394 , 0.9518606 ],
       [0.17078444, 0.82921556],
       [0.59743877, 0.40256123],
       [0.08829912, 0.91170088],
       [0.8876137 , 0.1123863 ],
       [0.03362888, 0.96637112],
       [0.02557452, 0.97442548],
       [0.24947139, 0.75052861],
       [0.02407659, 0.97592341],
       [0.06909475, 0.93090525],
       [0.81325681, 0.18674319],
       [0.08863516, 0.91136484],
       [0.02686139, 0.97313861],
       [0.83116316, 0.16883684],
       [0.84021365, 0.15978635],
       [0.01628384, 0.98371616],
       [0.2188689 , 0.7811311 ],
       [0.20984284, 0.79015716],
       [0.24260794, 0.75739206],
       [0.11317876, 0.88682124],
       [0.44319422, 0.55680578],
       [0.0312486 , 0.9687514 ],
       [0.0358787 , 0.9641213 ],
       [0.65569386, 0.34430614],
       [0.89579027, 0.10420973],
       [0.90985105, 0.09014895],
       [0.92803873, 0.07196127],
       [0.04143907, 0.95856093],
       [0.98723148, 0.01276852],
```

```
       [0.04003012, 0.95996988],
       [0.13481194, 0.86518806],
       [0.04486161, 0.95513839],
       [0.30865514, 0.69134486],
       [0.42271784, 0.57728216],
       [0.08754011, 0.91245989],
       [0.61294864, 0.38705136],
       [0.05356502, 0.94643498],
       [0.58391665, 0.41608335],
       [0.96259529, 0.03740471],
       [0.03546116, 0.96453884],
       [0.0365794 , 0.9634206 ],
       [0.04124809, 0.95875191],
       [0.73310092, 0.26689908],
       [0.27857483, 0.72142517],
       [0.08148662, 0.91851338],
       [0.17840631, 0.82159369],
       [0.12022803, 0.87977197],
       [0.05141357, 0.94858643],
       [0.86555268, 0.13444732],
       [0.81765148, 0.18234852],
       [0.03583028, 0.96416972],
       [0.54436461, 0.45563539],
       [0.48625252, 0.51374748],
       [0.97215613, 0.02784387],
       [0.87130014, 0.12869986],
       [0.58628658, 0.41371342]])
```

In [62]:
```
pred = mod.predict(X_test)

pred
```

Out[62]:
```
array([1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0,
       1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0,
       0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1,
       1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0,
       0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0,
       0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1,
       0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0,
       0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1,
       0, 0, 1, 0, 1, 0, 0, 0], dtype=int64)
```

In [63]:
```
mes.confusion_matrix(Y_test, pred)
```

Out[63]:
```
array([[ 66,   8],
       [  8, 102]], dtype=int64)
```

In [64]:
```
mes.accuracy_score(Y_test, pred)
```

Out[64]:
```
0.9130434782608695
```

In [65]:
```
mes.recall_score(Y_test, pred)
```

Out[65]:
```
0.9272727272727272
```

In [66]:
```
mes.precision_score(Y_test, pred)
```

Out[66]:
```
0.9272727272727272
```

In [67]:
```
# sauvegarde du modele de prediction mod

pkl.dump(mod, open('mod.pkl', 'wb' ))
```

In [ ]:
```

```