Aline Barboza Soares RA:1800136 E-mail: aline.soares@aluno.faculdadeimpacta.com.br

Luciana silveira RA:1800005 E-mail: luciana.silveira@aluno.faculdadeimpacta.com.br

1. (1,0) Leia a base de dados de nome cidade_br.csv e exiba os dados brutos no Jupyter Notebook.

In [69]:
```
import pandas
```

In [70]:
```
dados = pandas.read_csv('A:/DP-FACULDADE/AC03/cidades_br.csv')
```

In [71]:
```
print(dados)
```

```
      Código IBGE    Nome do Município  Código UF  UF              Estado  \
0         5200050      Abadia de Goiás         52  GO               Goiás
1         3100104  Abadia dos Dourados         31  MG        Minas Gerais
2         5200100            Abadiânia         52  GO               Goiás
3         3100203               Abaeté         31  MG        Minas Gerais
4         1500107           Abaetetuba         15  PA                Pará
...           ...                  ...        ...  ..                 ...
5565      4314548       Pinto Bandeira         43  RS  Rio Grande do Sul
5566      4220000     Balneário Rincão         42  SC     Santa Catarina
5567      4212650       Pescaria Brava         42  SC     Santa Catarina
5568      1504752     Mojuí dos Campos         15  PA                Pará
5569      5006275    Paraíso das Águas         50  MS  Mato Grosso do Sul

      Latitude  Longitude
0    -16.75730   -49.4412
1    -18.48310   -47.3916
2    -16.19700   -48.7057
3    -19.15510   -45.4444
4     -1.72183   -48.8788
...        ...        ...
5565 -29.09750   -51.4503
5566 -28.83140   -49.2352
5567 -28.39660   -48.8864
5568  -2.68220   -54.6425
5569 -19.02160   -53.0116

[5570 rows x 7 columns]
```

2. (1,0) Atribua x para Latitude e y para Longitude e visualize as coordenadas das cidades no gráfico tipo scatter

In [72]:
```
from pandas import DataFrame
```

In [73]:
```
import matplotlib.pyplot as plt
```

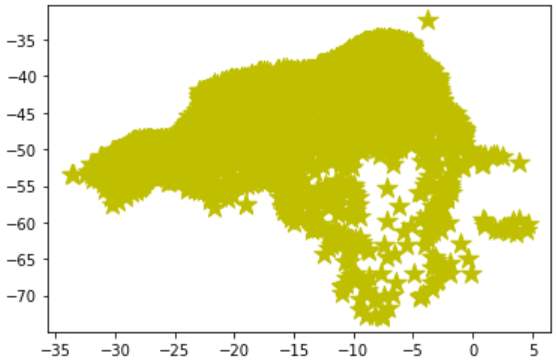
In [74]:
```
from sklearn.cluster import KMeans
```

In [75]:
```
Cidade = {'x': dados.Latitude,
          'y': dados.Longitude
         }
```

In [76]:
```
df = DataFrame(Cidade,columns=['x','y'])
```

In [77]:
```
plt.scatter(Cidade['x'],Cidade['y'], label = 'Pontos', color = 'y', marker = '*', s = 200)
```

```
<matplotlib.collections.PathCollection at 0x1d11b9eb148>
```



3. Aplique o algoritmo K-Médias para agrupar a base:

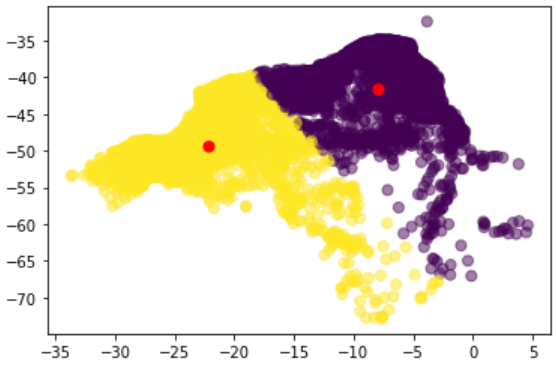
(1,0) Para dois clusters. Apresente os gráficos correspondentes com os pontos de cada cluster em uma cor diferente

In [78]:
```
kmeans = KMeans(n_clusters=2).fit(df)
centroids = kmeans.cluster_centers_
print(centroids)
```

```
[[ -8.00238897 -41.55229953]
 [-22.23791819 -49.43743607]]
```

In [79]:
```
plt.scatter(df['x'], df['y'], c= kmeans.labels_.astype(float), s=50, alpha=0.5)
plt.scatter(centroids[:, 0], centroids[:, 1], c='r', s=50)
```

```
<matplotlib.collections.PathCollection at 0x1d11a881088>
```



(1,0) Para o valor de clusters n a sua escolha.

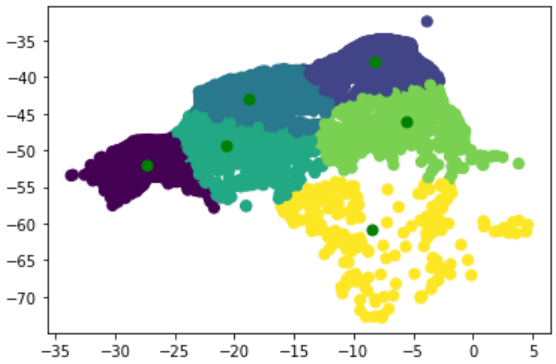
Apresente os gráficos correspondentes com os pontos de cada cluster em uma cor diferente

In [80]:
```
kmeans = KMeans(n_clusters=6).fit(df)
centroids = kmeans.cluster_centers_
print(centroids)
```

```
[[-27.37052782 -52.02494798]
 [ -8.16573994 -37.96833149]
 [-18.7982683  -42.95060045]
 [-20.70610984 -49.37727555]
 [ -5.52076186 -46.1457416 ]
 [ -8.45392943 -60.91168438]]
```

In [81]:
```
plt.scatter(df['x'], df['y'], c= kmeans.labels_.astype(float), s=50, alpha=1)
plt.scatter(centroids[:, 0], centroids[:, 1], c='g', s=50)
```

```
<matplotlib.collections.PathCollection at 0x1d11a7f7ac8>
```



4. Aplique o algoritmo DBSCAN para agrupar a base:

(1,0) Para os parâmetros eps=.3, min_samples=2. Apresente os gráficos correspondentes.

In [82]:
```
from sklearn.cluster import KMeans
from sklearn.cluster import DBSCAN
import matplotlib.pyplot as plt
import numpy as np
```

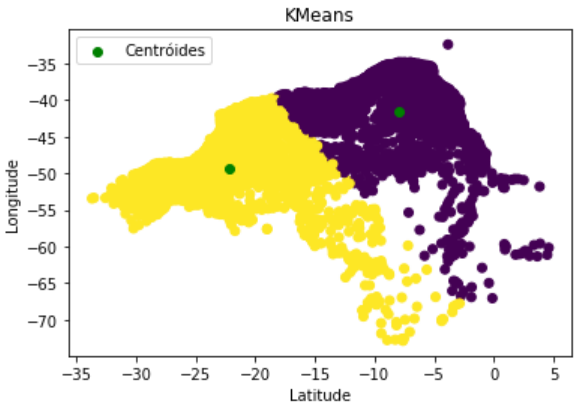
In [83]:
```
Brasil = dados[['Latitude','Longitude']].where(dados['UF'].isin(['RO','AC','AM','RR','PA','AP',
'TO','MA','PI','CE','RN','PB','PE','AL','SE','BA','MG','ES','RJ','SP','PR','SC','RS','MS','MT',
'GO','DF'
])).dropna()
x = Brasil.values
```

In [84]:
```
print(Brasil)
kmeans = KMeans(n_clusters=2)
kmeans.fit(x)
```

```
      Latitude  Longitude
0    -16.75730   -49.4412
1    -18.48310   -47.3916
2    -16.19700   -48.7057
3    -19.15510   -45.4444
4     -1.72183   -48.8788
...        ...        ...
5565 -29.09750   -51.4503
5566 -28.83140   -49.2352
5567 -28.39660   -48.8864
5568  -2.68220   -54.6425
5569 -19.02160   -53.0116

[5570 rows x 2 columns]


KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
       n_clusters=2, n_init=10, n_jobs=None, precompute_distances='auto',
       random_state=None, tol=0.0001, verbose=0)
```

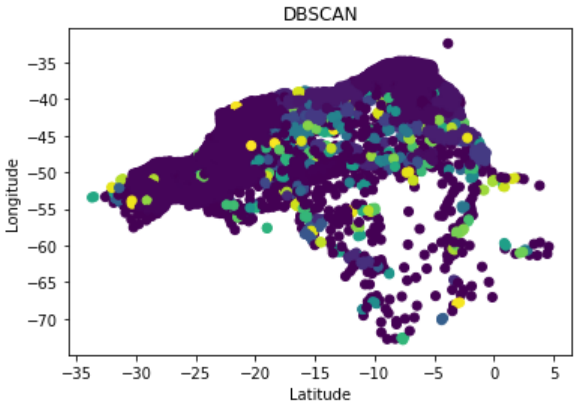
In [85]:
```
Brasil['clusters'] = kmeans.predict(x)
centros = kmeans.cluster_centers_
```

In [86]:
```
plt.title('KMeans')
plt.xlabel('Latitude')
plt.ylabel('Longitude')
plt.scatter(Brasil['Latitude'].values, Brasil['Longitude'].values, c = Brasil['clusters'])
plt.scatter(centros[:,0], centros[:,1], c = 'g', label='Centróides')
plt.legend()
plt.show()
```



In [87]:
```
dbscan = DBSCAN(eps=.3,min_samples=2)
z = dbscan.fit_predict(x)
```

In [88]:
```
plt.title('DBSCAN')
plt.xlabel('Latitude')
plt.ylabel('Longitude')
plt.scatter(Brasil['Latitude'].values, Brasil['Longitude'].values, c = z)
plt.show()
```



5.(2,0) Aplique os algoritmos K-Médias e DBSCAN para agrupar as cidades da região Sudeste.

Ajuste os parâmetros para obtermos 5 clusters com ambos os algoritmos.

Apresente os gráficos correspondentes.

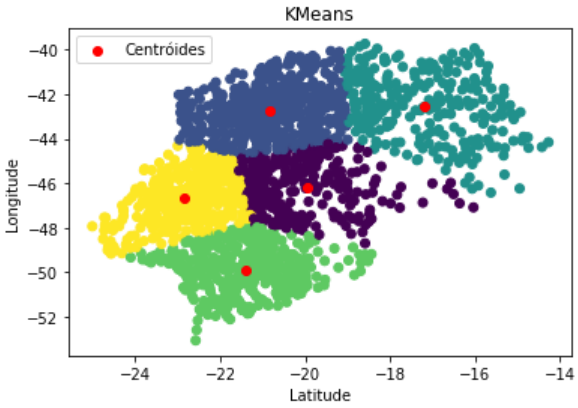
In [89]:
```
Sudeste = dados[['Latitude','Longitude']].where(dados['UF'].isin(['SP','sp','MG','mg','ES','es'
,'RJ','rj'])).dropna()
x = Sudeste.values
```

In [90]:
```
print(Sudeste)
kmeans = KMeans(n_clusters=5)
kmeans.fit(x)
```

```
       Latitude  Longitude
1    -18.483100 -47.391600
3    -19.155100 -45.444400
12   -20.299600 -42.474300
15   -20.359000 -43.143900
28   -19.067100 -42.541900
...         ...        ...
5541 -23.443890 -46.917778
5542 -23.221994 -45.310883
5543 -23.541056 -46.370972
5544 -21.549999 -47.780083
5545 -22.868000 -50.681374

[1668 rows x 2 columns]


KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
       n_clusters=5, n_init=10, n_jobs=None, precompute_distances='auto',
       random_state=None, tol=0.0001, verbose=0)
```

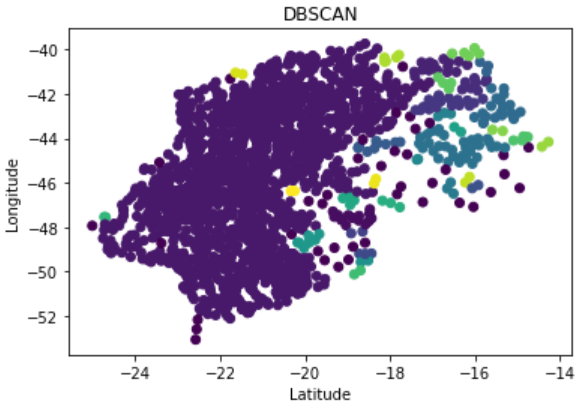
In [91]:
```
Sudeste['clusters'] = kmeans.predict(x)
centros = kmeans.cluster_centers_
```

In [92]:
```
plt.title('KMeans')
plt.xlabel('Latitude')
plt.ylabel('Longitude')
plt.scatter(Sudeste['Latitude'].values, Sudeste['Longitude'].values, c = Sudeste['clusters'])
plt.scatter(centros[:,0], centros[:,1], c = 'r', label='Centróides')
plt.legend()
plt.show()
```



In [93]:
```
dbscan = DBSCAN(eps=.3,min_samples=2)
z = dbscan.fit_predict(x)
```

In [94]:
```
plt.title('DBSCAN')
plt.xlabel('Latitude')
plt.ylabel('Longitude')
plt.scatter(Sudeste['Latitude'].values, Sudeste['Longitude'].values, c = z)
plt.show()
```



6.(2,0) Aplique os algoritmos K-Médias e DBSCAN para agrupar as cidades do estado do Amazonas.

Ajuste os parâmetros para obtermos 5 clusters com ambos os algoritmos. Apresente os gráficos correspondentes.

In [95]:
```
Amazonas = dados[['Latitude','Longitude']].where(dados['UF'].isin(['AM','am'])).dropna()
x = Amazonas.values
```

In [96]:
```
print(Amazonas)
```

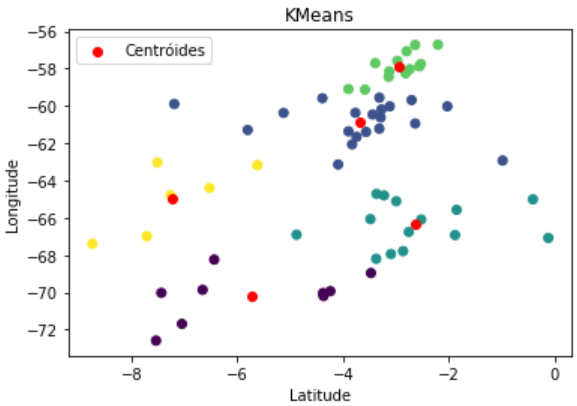
```
      Latitude  Longitude
167  -3.227270 -64.800700
192  -3.374550 -68.200500
213  -3.566970 -61.396300
249  -3.746030 -61.657500
283  -7.194090 -59.896000
...        ...        ...
5249 -2.996090 -65.113300
5304 -2.529360 -57.753800
5308 -3.128410 -58.149600
5473 -3.314000 -59.555737
5474 -6.438519 -68.243736

[62 rows x 2 columns]
```

In [97]:
```
kmeans = KMeans(n_clusters=5)
kmeans.fit(x)
```

```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
       n_clusters=5, n_init=10, n_jobs=None, precompute_distances='auto',
       random_state=None, tol=0.0001, verbose=0)
```

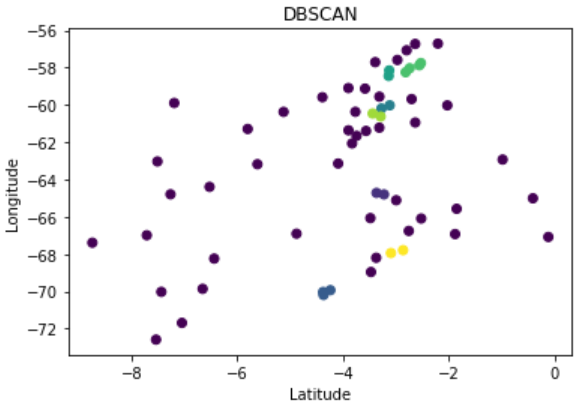
In [98]:
```
Amazonas['clusters'] = kmeans.predict(x)
centros = kmeans.cluster_centers_
```

In [99]:
```
plt.title('KMeans')
plt.xlabel('Latitude')
plt.ylabel('Longitude')
plt.scatter(Amazonas['Latitude'].values, Amazonas['Longitude'].values, c = Amazonas['clusters'
])
plt.scatter(centros[:,0], centros[:,1], c = 'r', label='Centróides')
plt.legend()
plt.show()
```



In [100]:
```
dbscan = DBSCAN(eps=.3,min_samples=2)
z = dbscan.fit_predict(x)
```

In [101]:
```
plt.title('DBSCAN')
plt.xlabel('Latitude')
plt.ylabel('Longitude')
plt.scatter(Amazonas['Latitude'].values, Amazonas['Longitude'].values, c = z)
plt.show()
```



In [ ]: