In [1]:

```
import seaborn as sns
from sklearn.cluster import KMeans
import pandas as pd
import numpy as np
```

In [2]:

```
data = pd.read_csv('driver-data.csv')
data.head()
```

Out[2]:

	id	mean_dist_day	mean_over_speed_perc
0	3423311935	71.24	28
1	3423313212	52.53	25
2	3423313724	64.54	27
3	3423311373	55.69	22
4	3423310999	54.58	25

In [3]:

```
kmeans = KMeans(n_clusters=4)
kmeans.fit(data)

print("Cluster's center\n")
print(kmeans.cluster_centers_)
```

```
Cluster's center

[[3.42331195e+09 7.76839364e+01 1.10059642e+01]
 [3.42331395e+09 7.49493952e+01 1.09657258e+01]
 [3.42331095e+09 7.78073473e+01 1.01551552e+01]
 [3.42331295e+09 7.37155633e+01 1.07567298e+01]]
```

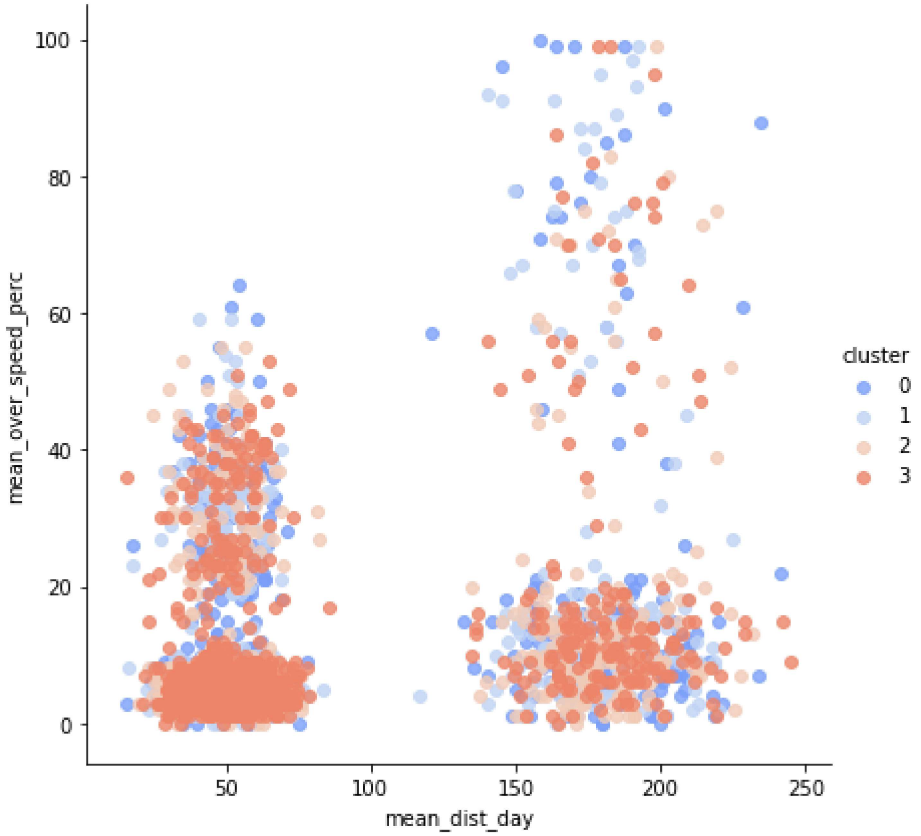
In [4]:

```
#Find count of each clusters
unique, counts = np.unique(kmeans.labels_, return_counts=True)
dict_data = dict(zip(unique, counts))
print("Count of each cluster>>>", dict_data)
```

```
Count of each cluster>>> {0: 1003, 1: 995, 2: 1001, 3: 1001}
```

In [7]:
```
#plot the clusters
data["cluster"] = kmeans.labels_
sns.lmplot('mean_dist_day', 'mean_over_speed_perc', data=data, hue='cluster', palette='coolwarm', size=6, aspect=1, fit_reg=False)
```

Out[7]:
```
<seaborn.axisgrid.FacetGrid at 0x1d5bae7c0c8>
```



In [8]:
```
#Inertia is the sum of squared error for each cluster. Therefore the smaller the inertia the denser the cluster is
print("Inertia\n")
print(kmeans.inertia_)
```

```
Inertia

345521258.9375957
```

In [9]:

```
#Print the data
print("Datawith clusters>>> \n", data)
```

```
Datawith clusters>>> 
               id  mean_dist_day  mean_over_speed_perc  clusters  cluster
0     3423311935          71.24                    28         0        0
1     3423313212          52.53                    25         3        3
2     3423313724          64.54                    27         1        1
3     3423311373          55.69                    22         2        2
4     3423310999          54.58                    25         2        2
...          ...            ...                   ...       ...      ...
3995  3423310685         160.04                    10         2        2
3996  3423312600         176.17                     5         3        3
3997  3423312921         170.91                    12         3        3
3998  3423313630         176.14                     5         1        1
3999  3423311533         168.03                     9         0        0

[4000 rows x 5 columns]
```

In [ ]:

```
```