In [1]:
```
import warnings
warnings.filterwarnings("ignore")

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df=pd.read_csv("data-MYoQk.csv")
df
```

Out[2]:

	Project title	Country	City	Environment	Line name	Description	Status	Start planning	Start Construction	End Year	...	Elevated percentage	
0	Broadway	Canada	Vancouver	Urban	Millenium	Extension of existing millennium line to add i...	In construction	2018	2020	2025	...	5%	D
1	Vaughan	Canada	Toronto	Urban	Line 1	Extension of existing line to a new terminus	Complete	2005	2009	2017	...	0%	D
2	Scarborough	Canada	Toronto	Urban	Line 2	Will extend existing subway to the city of Sca...	In construction	2020	2023	2030	...	0%	D
3	Ontario	Canada	Toronto	Urban	Ontario Line	New line through the centre of Toronto	In construction	2019	2022	2031	...	37%	D
4	Yonge to Richmond Hill	Canada	Toronto	Urban	Line 1	Extending an existing line to new suburbs	In construction	2021	2023	2030	...	0%	D
...	...	...	...	...	...	...	...	...	...	...	...	...	
133	Extension to Bourtzwiller	France	Mulhouse	Urban	Mulhouse Tram	Construction of an extension of the existing line	Complete	?	2007	2009	...	0%	
134	U4 to Eibbrucken	Germany	Hamburg	Urban	U4	Extension of the existing line	Complete	2013	2014	2018	...	0%	D
135	U4 to HafenCity U	Germany	Hamburg	Urban	U4	Extension of the existing line	Complete	?	2007	2012	...	0%	D
136	Wehrhahn line	Germany	Dusseldorf	Urban	Wehrhahn Line	Construction of a new metro for Dusseldorf tha...	Complete	?	2007	2016	...	0%	D
137	Phase 2 Nottingham Trams	UK	Nottingham	Urban	Phase 2	Construction of an extension to the south and ...	Complete	2006	2012	2015	...	0%	

138 rows × 26 columns

```
- Our variable features:


- Project title: Name given to every project.
- Country: Names of the country where work has been done of rail transport.
- City: Names of the city of the various country where work has been done of rail transport.
- Environment: Names of the different areas of multiple city where work has been done of rail transport.
- Line name: Names of the rail tranport lines.
- Description: Which type of work is going on is given.
- Status: Status of the project is given.
- Start planning: In the year planning of the project has been started.
- Start Construction: In the year project has been started.
- End year: In the year project will be completing.
- Length (Miles): How much long rail line working has been done or have to be done.
- Numbers of Stations: Who many railway stations is there in different areas and citys.
- Type of Project: What type of rail work has been done in the project.
- Type of Line: What type of rail line work has been done.
- Tunnelling method: Which type of method is been used for making tunnels is given.
- Tunnel percentage: Percentage of tunnel work has been done in different countrys/citys/areas.
- Elevated percentage: Pending of work of tunnel has been given.
- Source: Information saved about rail line projects.
- Cost (m): How much cost has been used for rail line work.
- Currency: Which type of currency for completion of rail line.
- Year: year has been given.
- Converted to mil GBP: Here length has been converted into miles.
- CPI adjusted (mil GBP): Consumer price index is given.
- Cost per mile (mil GBP): cost of the rail line per mile is given.
- Source 1: websites is given related rail line work.
- Source 2: contain null values.
```

EDA

In [3]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 138 entries, 0 to 137
Data columns (total 26 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Project title           138 non-null    object 
 1   Country                 138 non-null    object 
 2   City                    138 non-null    object 
 3   Environment             138 non-null    object 
 4   Line name               138 non-null    object 
 5   Description             138 non-null    object 
 6   Status                  138 non-null    object 
 7   Start planning          137 non-null    object 
 8   Start Construction      138 non-null    object 
 9   End Year                138 non-null    object 
 10  Length (Miles)          138 non-null    float64
 11  Number of Stations      138 non-null    int64  
 12  Type of project         138 non-null    object 
 13  Type of Line            138 non-null    object 
 14  Tunnelling method       95 non-null     object 
 15  Tunnel percentage       138 non-null    object 
 16  Elevated percentage     138 non-null    object 
 17  Source                  138 non-null    object 
 18  Cost (m)                138 non-null    float64
 19  Currency                138 non-null    object 
 20  Year                    138 non-null    int64  
 21  Converted to mil GBP    138 non-null    int64  
 22  CPI adjusted (mil GBP)  138 non-null    int64  
 23  Cost per mile (mil GBP) 138 non-null    int64  
 24  Source 1                138 non-null    object 
 25  Source 2                12 non-null     object 
dtypes: float64(2), int64(5), object(19)
memory usage: 28.2+ KB
```

```
here we have used df.info() to get detail about the columns how many columns present in the csv entries
present in the columns and there data types.
```

In [4]:
```
df.isnull().sum()
```

Out[4]:
```
Project title               0
Country                     0
City                        0
Environment                 0
Line name                   0
Description                 0
Status                      0
Start planning              1
Start Construction          0
End Year                    0
Length (Miles)              0
Number of Stations          0
Type of project             0
Type of Line                0
Tunnelling method          43
Tunnel percentage           0
Elevated percentage         0
Source                      0
Cost (m)                    0
Currency                    0
Year                        0
Converted to mil GBP        0
CPI adjusted (mil GBP)      0
Cost per mile (mil GBP)     0
Source 1                    0
Source 2                  126
dtype: int64
```

```
Here we have used df.isnull().sum() to identify that in how many there are null values.
```

In [5]:
```
df.head()
```

Out[5]:

	Project title	Country	City	Environment	Line name	Description	Status	Start planning	Start Construction	End Year	...	Elevated percentage	Sou
0	Broadway	Canada	Vancouver	Urban	Millenium	Extension of existing millennium line to add i...	In construction	2018	2020	2025	...	5%	Databa
1	Vaughan	Canada	Toronto	Urban	Line 1	Extension of existing line to a new terminus	Complete	2005	2009	2017	...	0%	Databa
2	Scarborough	Canada	Toronto	Urban	Line 2	Will extend existing subway to the city of Sca...	In construction	2020	2023	2030	...	0%	Databa
3	Ontario	Canada	Toronto	Urban	Ontario Line	New line through the centre of Toronto	In construction	2019	2022	2031	...	37%	Databa
4	Yonge to Richmond Hill	Canada	Toronto	Urban	Line 1	Extending an existing line to new suburbs	In construction	2021	2023	2030	...	0%	Databa

5 rows × 26 columns

```
Here we have used df.head() we see first five columns of data in the present csv.
```

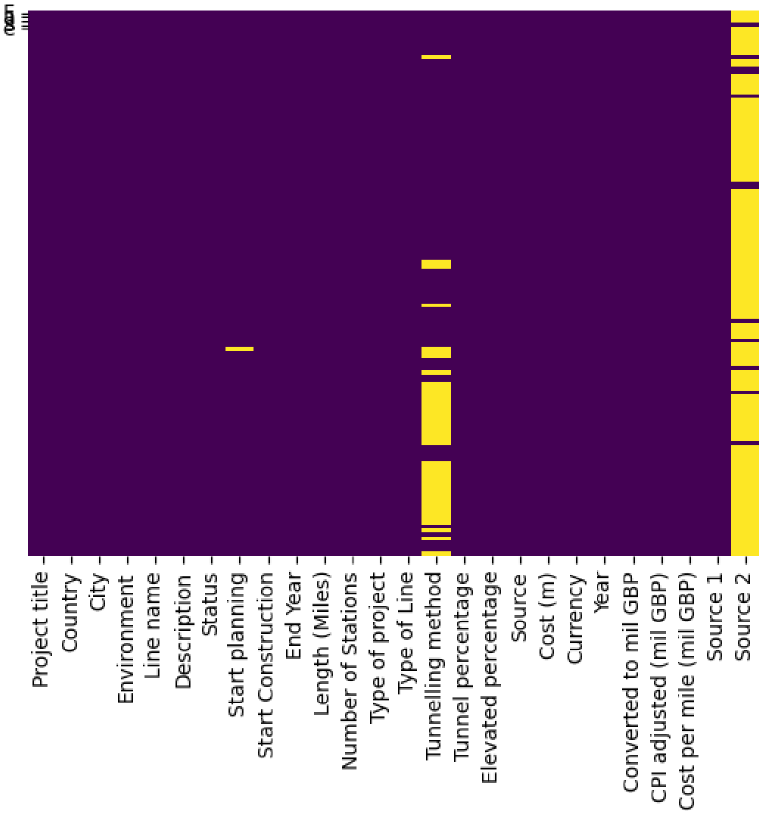
Data Cleaning

In [6]:
```
df.isnull().sum()
```

Out[6]:
```
Project title               0
Country                     0
City                        0
Environment                 0
Line name                   0
Description                 0
Status                      0
Start planning              1
Start Construction          0
End Year                    0
Length (Miles)              0
Number of Stations          0
Type of project             0
Type of Line                0
Tunnelling method          43
Tunnel percentage           0
Elevated percentage         0
Source                      0
Cost (m)                    0
Currency                    0
Year                        0
Converted to mil GBP        0
CPI adjusted (mil GBP)      0
Cost per mile (mil GBP)     0
Source 1                    0
Source 2                  126
dtype: int64
```

In [ ]:
```
Here we have used df.isnull().sum() to identify that in how many there are null values.
```

In [7]:
```
sns.heatmap(df.isnull(),yticklabels="False",cbar=False,cmap="viridis")
plt.show()
```

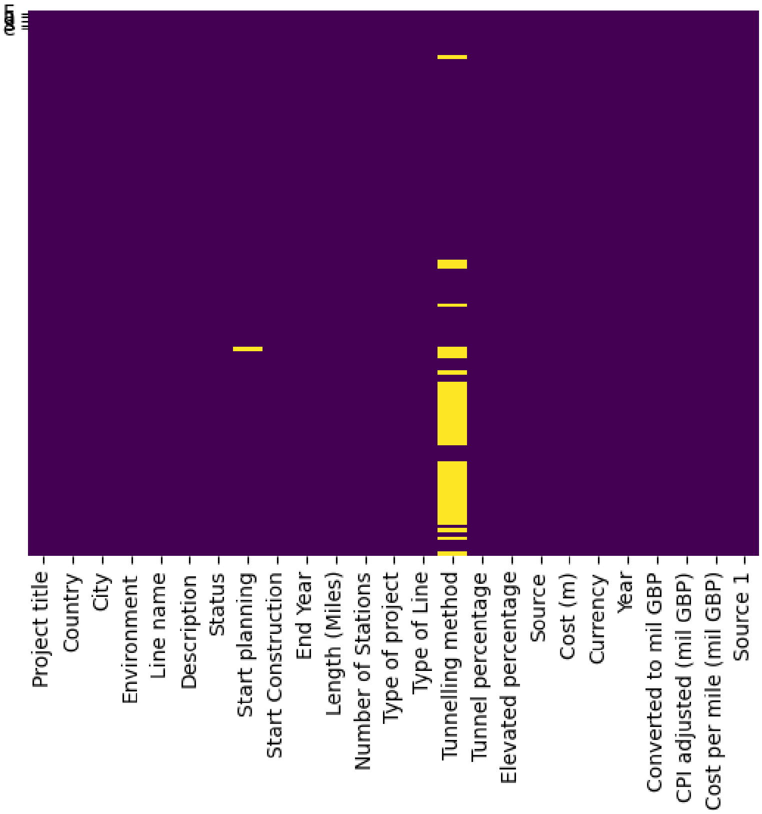


```
Here we have made heatmap to identify as you see the chart there null values present in Start
planning,Tunnelling method and Source2 there are maximum null values in Source2.
```

In [8]:
```
df.drop("Source 2",axis=1,inplace=True)
```

```
So here we have drop source2 using df.drop() because it contains 75% plus null values.
```

In [9]:
```
sns.heatmap(df.isnull(),yticklabels="False",cbar=False,cmap="viridis")
plt.show()
```



```
after using df.drop() on source2 as you can see above the column of source2 has been droped there are null
values present in Tunnelling percentage.
```

DATA VISUALIZATION

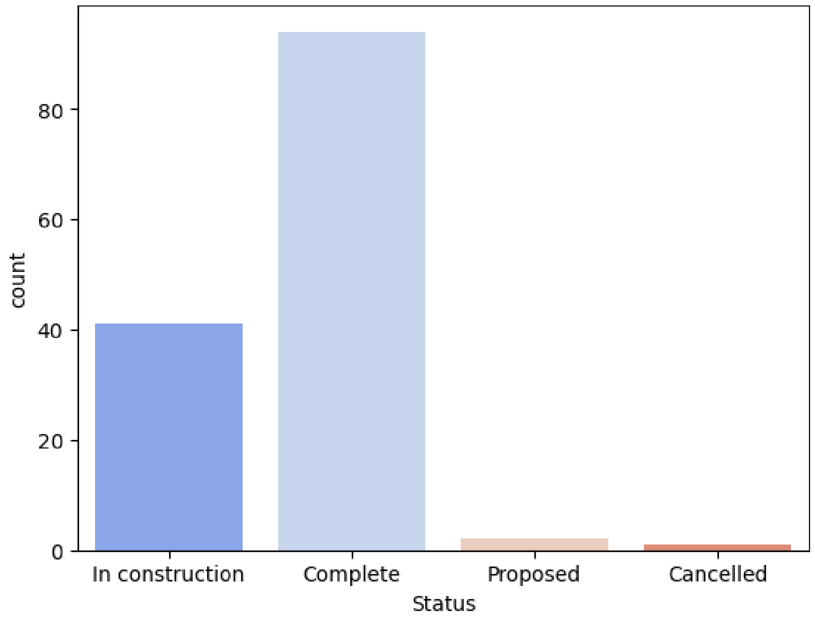
In [10]:
```
df["Status"].value_counts()
```

Out[10]:
```
Complete           94
In construction    41
Proposed            2
Cancelled           1
Name: Status, dtype: int64
```

```
Here we have used value_counts() on status columns to see that how many rail line has been complete/in
construction/proposed and cancelled.
```

```
In [11]: sns.countplot(data=df,x="Status",palette='coolwarm')
```

```
Out[11]: <Axes: xlabel='Status', ylabel='count'>
```



```
This is the bar graph of Status columns.
```

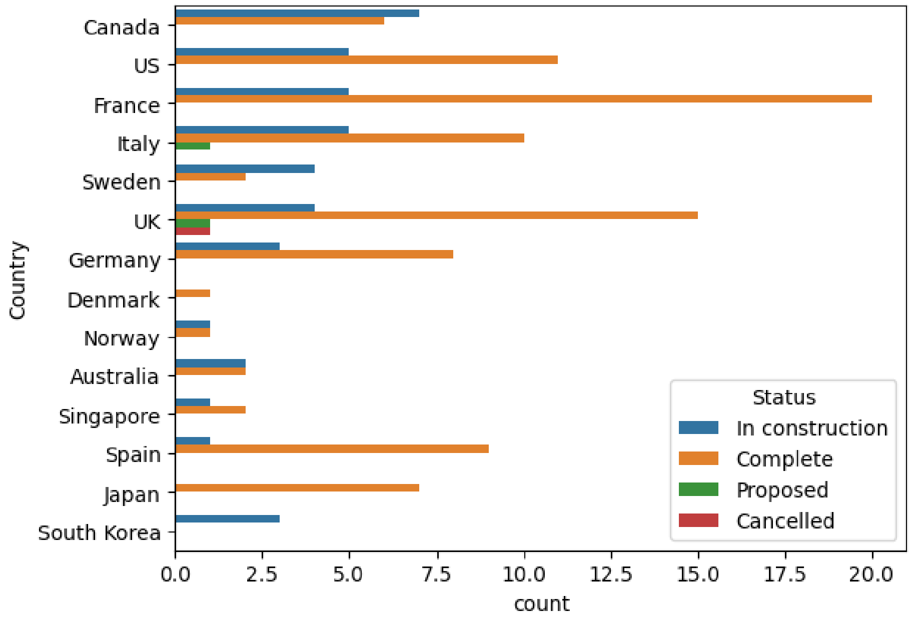
```
In [12]: df["Country"].value_counts()
```

```
Out[12]: France         25
         UK             21
         US             16
         Italy          16
         Canada         13
         Germany        11
         Spain          10
         Japan           7
         Sweden          6
         Australia       4
         Singapore       3
         South Korea     3
         Norway          2
         Denmark         1
         Name: Country, dtype: int64
```

```
Here we have used value_counts() on Country columns to see that in how many Countrys railways is going on/has
been done.
```

```
In [13]: sns.countplot(data=df,y="Country",hue="Status")
```

```
Out[13]: <Axes: xlabel='count', ylabel='Country'>
```



This is the countplot of the Country columns and Status to see in which country the rail line are in construction/complete/proposed and cancelled.

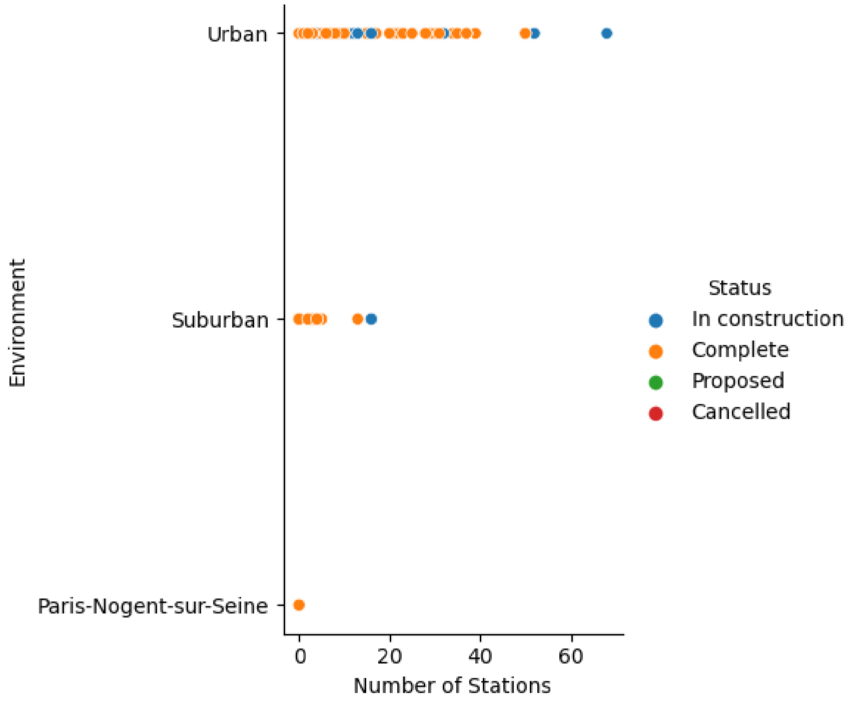
```
In [14]: df["Environment"].value_counts()
```

```
Out[14]: Urban                      122
         Suburban                    15
         Paris-Nogent-sur-Seine       1
         Name: Environment, dtype: int64
```

Here we have used value_counts() on Environment columns to see that in which areas rail line work is in progress/complete.

```
In [15]: sns.relplot(x="Number of Stations",y="Environment",data=df,hue="Status")
```

```
Out[15]: <seaborn.axisgrid.FacetGrid at 0x23f5cc893f0>
```



Here we have made relplot of Number of Stations/Environment/Status columns to see that there are how many stations present in Areas and are they in construction/complete/proposed and cancelled.

In [16]:

```
df["Length (Miles)"].value_counts()
```

Out[16]:

```
1.68     4
2.80     3
0.81     2
1.74     2
0.99     2
        ..
4.66     1
2.86     1
46.58    1
9.94     1
10.87    1
Name: Length (Miles), Length: 114, dtype: int64
```

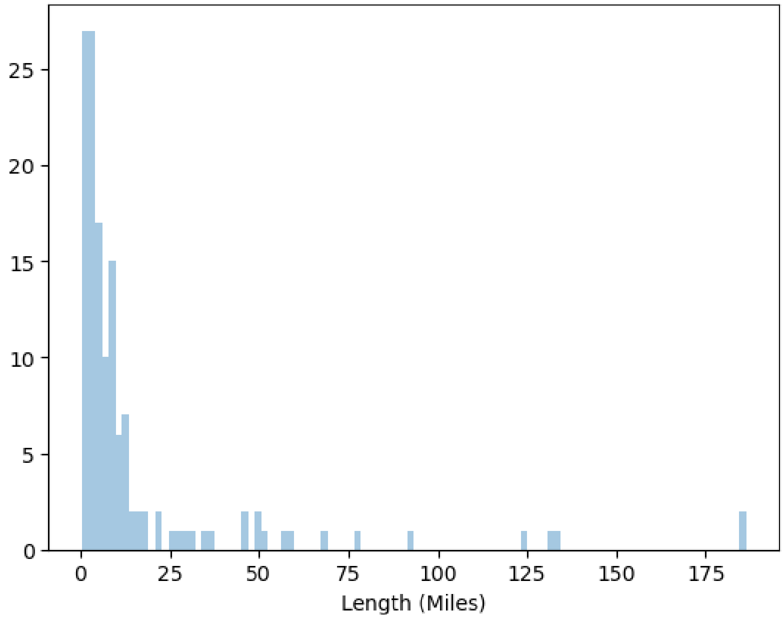
Here we have used value_counts() on Length (Miles) columns to see how long rail line is.

In [17]:

```
sns.distplot(df["Length (Miles)"],kde=False,bins=100)
```

Out[17]:

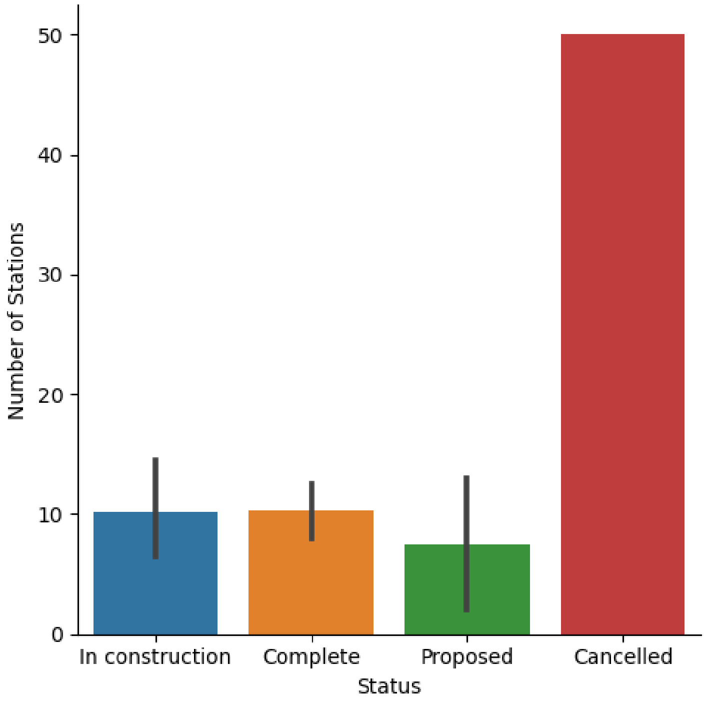
```
<Axes: xlabel='Length (Miles)'>
```



In [ ]: Here we have made distplot of Length (miles) to identify the length of the rail line.

In [18]:
```
sns.catplot(x="Status",y="Number of Stations",data=df,kind='bar')
```

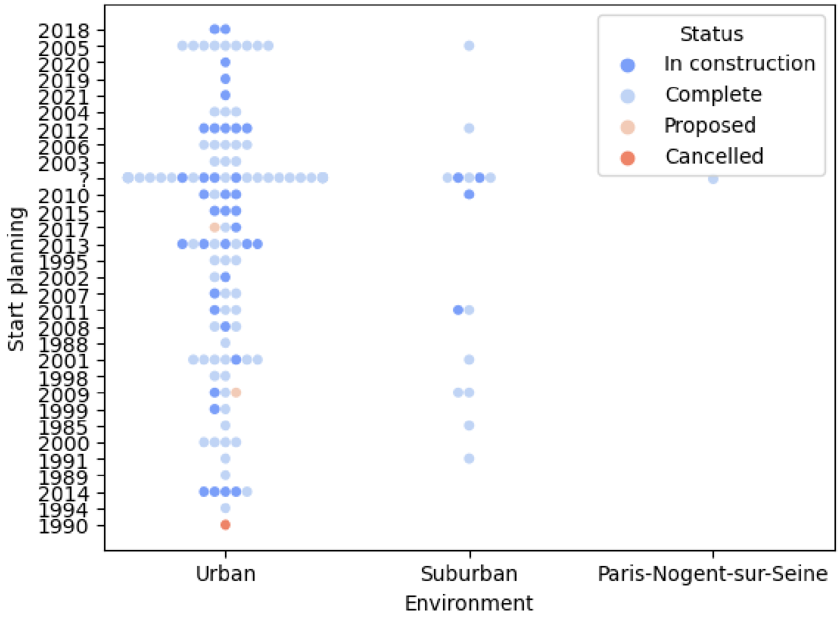
Out[18]:
```
<seaborn.axisgrid.FacetGrid at 0x23f5d527400>
```



```
Here we have catplot(bar) to identify about Number of Stations which are in
construction/complete/proposed/cancelled.
```

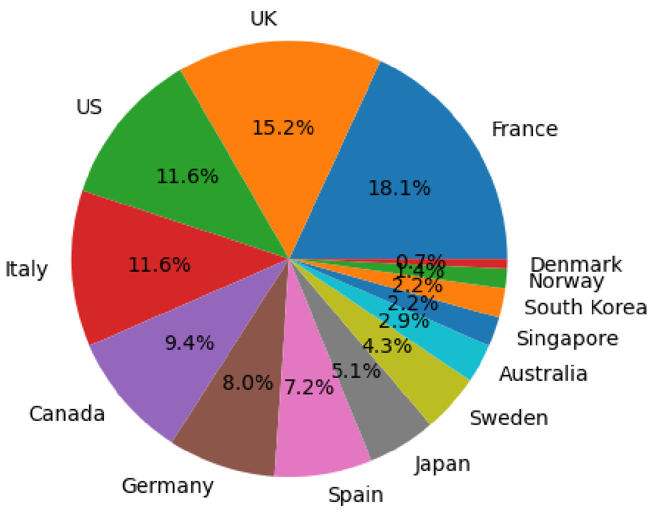
In [19]:
```
sns.swarmplot(x="Environment",y="Start planning",data=df,hue="Status",p)
```

Out[19]:
```
<Axes: xlabel='Environment', ylabel='Start planning'>
```



In [ ]:
```
Here we have made swarmplot to identify about Environment(areas) where rail line work planning has been started
```

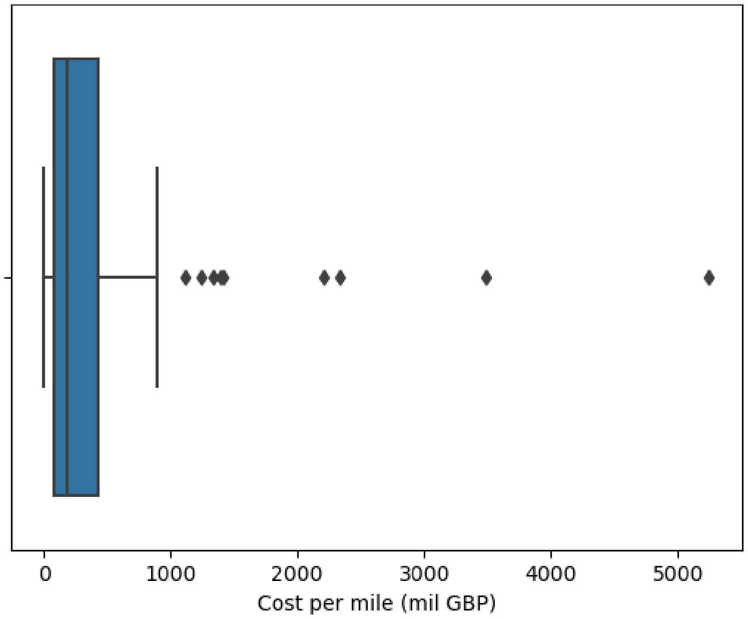
In [20]:
```
plt.pie(df["Country"].value_counts(), labels=["France","UK","US","Italy","Canada","Germany","Spain","Japan","Sw
plt.show()
```



```
Here we have to made pieplot to identify in which Country the highest number of rail line work is in progress.
```

In [21]:
```
sns.boxplot(data=df,x="Cost per mile (mil GBP)")
```

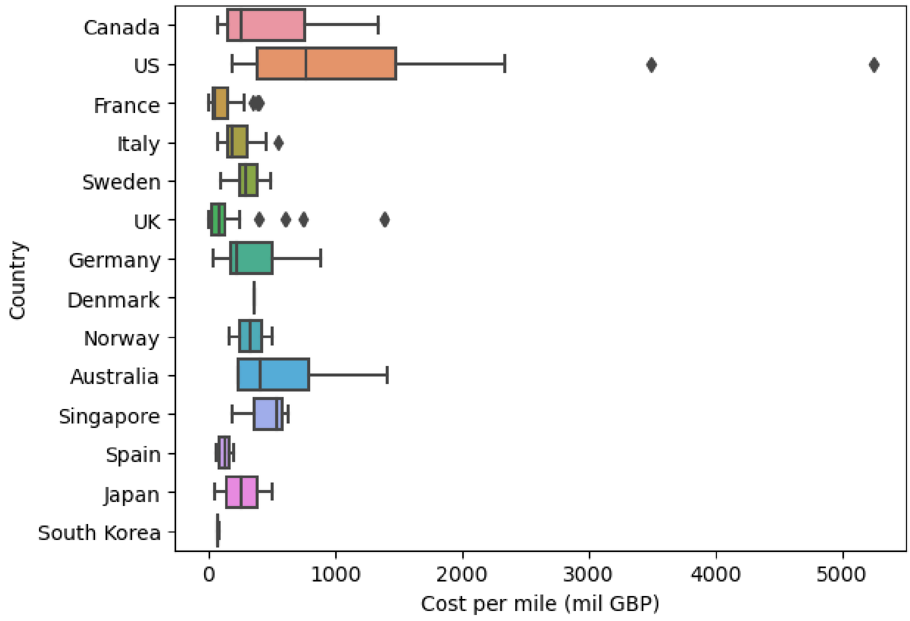
Out[21]:
```
<Axes: xlabel='Cost per mile (mil GBP)'>
```



```
Here we have made boxplot to identify highest Cost per mile (mil GBP) is in use for rail line work
```

In [22]:
```
sns.boxplot(data=df,x="Cost per mile (mil GBP)",y="Country")
```

Out[22]:
```
<Axes: xlabel='Cost per mile (mil GBP)', ylabel='Country'>
```



Here we have made boxplot to identify highest Cost per mile (mil GBP) is used in which country.

In [23]:
```
df.describe()
```

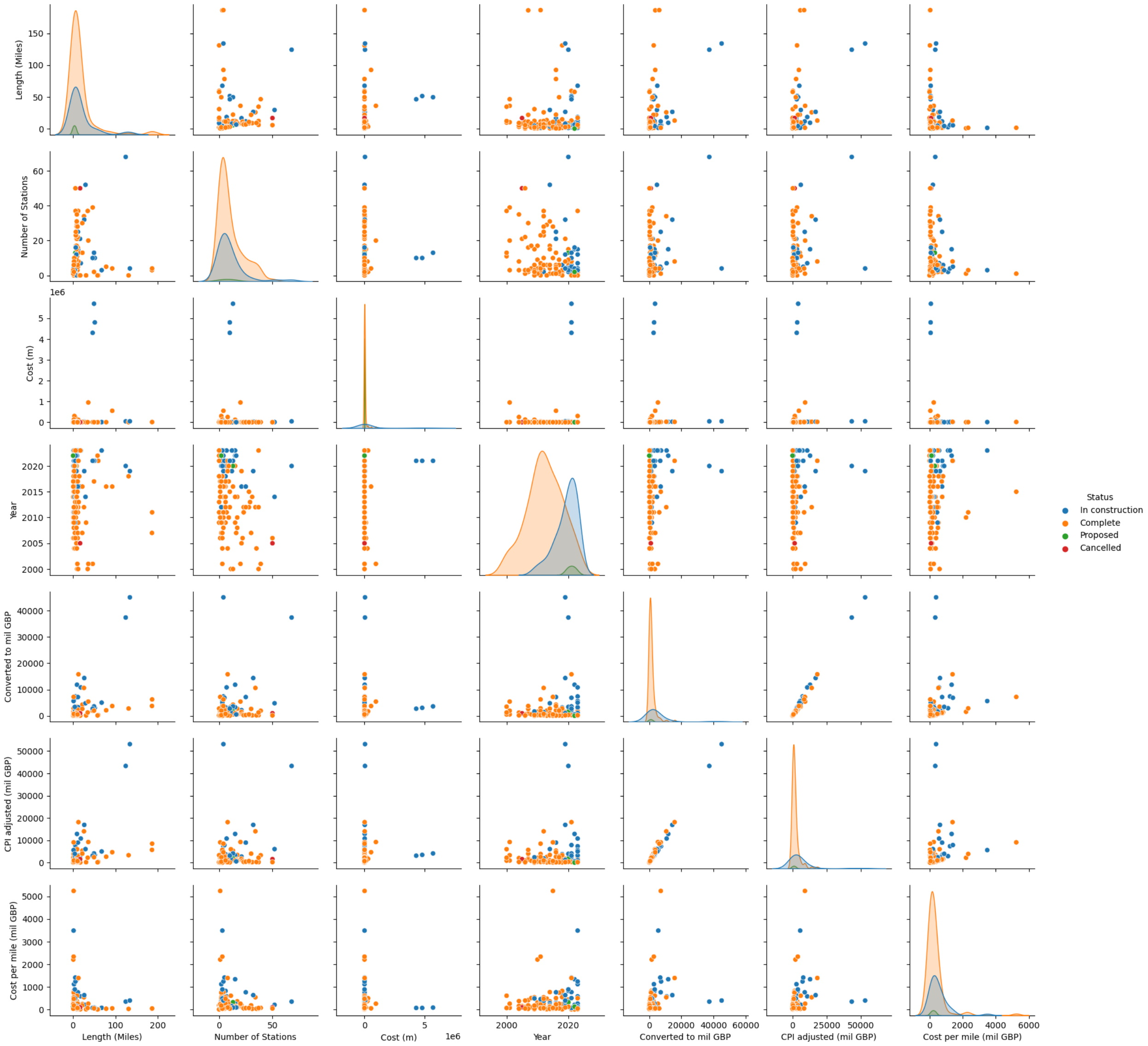
Out[23]:

	Length (Miles)	Number of Stations	Cost (m)	Year	Converted to mil GBP	CPI adjusted (mil GBP)	Cost per mile (mil GBP)
count	138.000000	138.000000	1.380000e+02	138.000000	138.000000	138.000000	138.000000
mean	16.331806	10.485507	1.280785e+05	2014.449275	2429.884058	2968.514493	371.217391
std	31.247769	12.383316	7.307356e+05	6.088247	5441.881471	6374.796669	623.070238
min	0.430000	0.000000	2.550000e+01	2000.000000	23.000000	33.000000	3.000000
25%	2.640000	2.250000	3.992500e+02	2010.250000	327.000000	428.500000	84.750000
50%	5.870000	6.000000	1.443650e+03	2015.000000	940.500000	1216.500000	186.000000
75%	11.770000	15.000000	5.575000e+03	2020.000000	2168.500000	2729.000000	431.750000
max	186.340000	68.000000	5.700000e+06	2023.000000	45000.000000	53101.000000	5244.000000

In [ ]:
```
Here we have df.describe() to see short describetion of the project columns.
```

```
In [24]: sns.pairplot(data=df,hue="Status")
```

```
Out[24]: <seaborn.axisgrid.PairGrid at 0x23f5cc8b340>
```



Here we have made pairplot to identify length (miles)/number of stations/cost (m)/year/converted to mil GBP/CPI adjusted (mil GBP)/Cost per mile (mil GBP) with status.