In [2]:

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

In [3]:

```
a=pd.read_csv(r"C:\Users\user\Downloads\9_bottle.csv")
a
```

```
                                          20-
                                     1611SR-
864860   34404  864861   093.4       MX-310-    5   18.692  33.4150   5.796  23.88911  108.46  ...
                         026.4         2239-
                                09340264-
                                     0005A-3

                                          20-
                                     1611SR-
864861   34404  864862   093.4       MX-310-   10   18.161  33.4062   5.816  24.01426  107.74  ...
                         026.4         2239-
                                09340264-
                                     0010A-3

                                          20-
                                     1611SR-
864862   34404  864863   093.4       MX-310-   15   17.533  33.3880   5.774  24.15297  105.66  ...
                         026.4         2239-
                                09340264-
                                     0015A-3

864863 rows × 74 columns
```

In [4]:

```
a=a.head(10)
a
```

Out[4]:

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta	O2Sat	...	R_PHAEO	R
0	1	1	054.0 056.0	19-4903CR-HY-060-0930-05400560-0000A-3	0	10.50	33.440	NaN	25.649	NaN	...	NaN	
1	1	2	054.0 056.0	19-4903CR-HY-060-0930-05400560-0008A-3	8	10.46	33.440	NaN	25.656	NaN	...	NaN	
2	1	3	054.0 056.0	19-4903CR-HY-060-0930-05400560-0010A-7	10	10.46	33.437	NaN	25.654	NaN	...	NaN	
3	1	4	054.0 056.0	19-4903CR-HY-060-0930-05400560-0019A-3	19	10.45	33.420	NaN	25.643	NaN	...	NaN	
4	1	5	054.0 056.0	19-4903CR-HY-060-0930-05400560-0020A-7	20	10.45	33.421	NaN	25.643	NaN	...	NaN	
5	1	6	054.0 056.0	19-4903CR-HY-060-0930-05400560-0030A-7	30	10.45	33.431	NaN	25.651	NaN	...	NaN	
6	1	7	054.0 056.0	19-4903CR-HY-060-0930-05400560-0039A-3	39	10.45	33.440	NaN	25.658	NaN	...	NaN	
7	1	8	054.0 056.0	19-4903CR-HY-060-0930-05400560-0050A-7	50	10.24	33.424	NaN	25.682	NaN	...	NaN	
8	1	9	054.0 056.0	19-4903CR-HY-060-0930-05400560-0058A-3	58	10.06	33.420	NaN	25.710	NaN	...	NaN	
9	1	10	054.0 056.0	19-4903CR-HY-060-0930-05400560-0075A-7	75	9.86	33.494	NaN	25.801	NaN	...	NaN	

```
10 rows × 74 columns
```

In [5]:

```
a.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 74 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   Cst_Cnt              10 non-null     int64  
 1   Btl_Cnt              10 non-null     int64  
 2   Sta_ID               10 non-null     object 
 3   Depth_ID             10 non-null     object 
 4   Depthm               10 non-null     int64  
 5   T_degC               10 non-null     float64
 6   Salnty               10 non-null     float64
 7   O2ml_L               0 non-null      float64
 8   STheta               10 non-null     float64
 9   O2Sat                0 non-null      float64
 10  Oxy_µmol/Kg          0 non-null      float64
 11  BtlNum               0 non-null      float64
 12  RecInd               10 non-null     int64  
 13  T_prec               10 non-null     float64
 14  T_qual               0 non-null      float64
 15  S_prec               10 non-null     float64
 16  S_qual               0 non-null      float64
 17  P_qual               10 non-null     float64
 18  O_qual               10 non-null     float64
 19  SThtaq               0 non-null      float64
 20  O2Satq               10 non-null     float64
 21  ChlorA               0 non-null      float64
 22  Chlqua               10 non-null     float64
 23  Phaeop               0 non-null      float64
 24  Phaqua               10 non-null     float64
 25  PO4uM                0 non-null      float64
 26  PO4q                 10 non-null     float64
 27  SiO3uM               0 non-null      float64
 28  SiO3qu               10 non-null     float64
 29  NO2uM                0 non-null      float64
 30  NO2q                 10 non-null     float64
 31  NO3uM                0 non-null      float64
 32  NO3q                 10 non-null     float64
 33  NH3uM                0 non-null      float64
 34  NH3q                 10 non-null     float64
 35  C14As1               0 non-null      float64
 36  C14A1p               0 non-null      float64
 37  C14A1q               10 non-null     float64
 38  C14As2               0 non-null      float64
 39  C14A2p               0 non-null      float64
 40  C14A2q               10 non-null     float64
 41  DarkAs               0 non-null      float64
 42  DarkAp               0 non-null      float64
 43  DarkAq               10 non-null     float64
 44  MeanAs               0 non-null      float64
 45  MeanAp               0 non-null      float64
 46  MeanAq               10 non-null     float64
 47  IncTim               0 non-null      object 
 48  LightP               0 non-null      float64
 49  R_Depth              10 non-null     float64
 50  R_TEMP               10 non-null     float64
 51  R_POTEMP             10 non-null     float64
 52  R_SALINITY           10 non-null     float64
 53  R_SIGMA              10 non-null     float64
 54  R_SVA                10 non-null     float64
 55  R_DYNHT              10 non-null     float64
 56  R_O2                 0 non-null      float64
 57  R_O2Sat              0 non-null      float64
 58  R_SIO3               0 non-null      float64
 59  R_PO4                0 non-null      float64
 60  R_NO3                0 non-null      float64
 61  R_NO2                0 non-null      float64
 62  R_NH4                0 non-null      float64
 63  R_CHLA               0 non-null      float64
 64  R_PHAEO              0 non-null      float64
 65  R_PRES               10 non-null     int64
 66  R_SAMP               0 non-null      float64
 67  DIC1                 0 non-null      float64
 68  DIC2                 0 non-null      float64
 69  TA1                  0 non-null      float64
 70  TA2                  0 non-null      float64
 71  pH2                  0 non-null      float64
 72  pH1                  0 non-null      float64
 73  DIC Quality Comment  0 non-null      object
dtypes: float64(65), int64(5), object(4)
memory usage: 5.9+ KB
```

In [6]:
```
a.columns
```

Out[6]:
```
Index(['Cst_Cnt', 'Btl_Cnt', 'Sta_ID', 'Depth_ID', 'Depthm', 'T_degC',
       'Salnty', 'O2ml_L', 'STheta', 'O2Sat', 'Oxy_µmol/Kg', 'BtlNum',
       'RecInd', 'T_prec', 'T_qual', 'S_prec', 'S_qual', 'P_qual', 'O_qual',
       'SThtaq', 'O2Satq', 'ChlorA', 'Chlqua', 'Phaeop', 'Phaqua', 'PO4uM',
       'PO4q', 'SiO3uM', 'SiO3qu', 'NO2uM', 'NO2q', 'NO3uM', 'NO3q', 'NH3uM',
       'NH3q', 'C14As1', 'C14A1p', 'C14A1q', 'C14As2', 'C14A2p', 'C14A2q',
       'DarkAs', 'DarkAp', 'DarkAq', 'MeanAs', 'MeanAp', 'MeanAq', 'IncTim',
       'LightP', 'R_Depth', 'R_TEMP', 'R_POTEMP', 'R_SALINITY', 'R_SIGMA',
       'R_SVA', 'R_DYNHT', 'R_O2', 'R_O2Sat', 'R_SIO3', 'R_PO4', 'R_NO3',
       'R_NO2', 'R_NH4', 'R_CHLA', 'R_PHAEO', 'R_PRES', 'R_SAMP', 'DIC1',
       'DIC2', 'TA1', 'TA2', 'pH2', 'pH1', 'DIC Quality Comment'],
      dtype='object')
```

In [27]:
```
d=a[['RecInd']]
d
```

Out[27]:

	RecInd
0	3
1	3
2	7
3	3
4	7
5	7
6	3
7	7
8	3
9	7

In [28]:
```
a.describe()
```

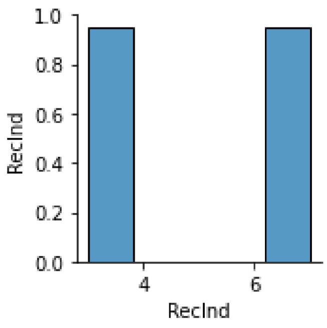
Out[28]:

	Cst_Cnt	Btl_Cnt	Depthm	T_degC	Salnty	O2ml_L	STheta	O2Sat	Oxy_µmol/Kg	BtlNu
count	10.0	10.00000	10.000000	10.000000	10.000000	0.0	10.000000	0.0	0.0	0.
mean	1.0	5.50000	30.900000	10.338000	33.436700	NaN	25.674700	NaN	NaN	Na
std	0.0	3.02765	24.237024	0.216426	0.021894	NaN	0.048922	NaN	NaN	Na
min	1.0	1.00000	0.000000	9.860000	33.420000	NaN	25.643000	NaN	NaN	Na
25%	1.0	3.25000	12.250000	10.292500	33.421750	NaN	25.649500	NaN	NaN	Na
50%	1.0	5.50000	25.000000	10.450000	33.434000	NaN	25.655000	NaN	NaN	Na
75%	1.0	7.75000	47.250000	10.457500	33.440000	NaN	25.676000	NaN	NaN	Na
max	1.0	10.00000	75.000000	10.500000	33.494000	NaN	25.801000	NaN	NaN	Na

8 rows × 70 columns

In [29]:
```
sns.pairplot(d)
```

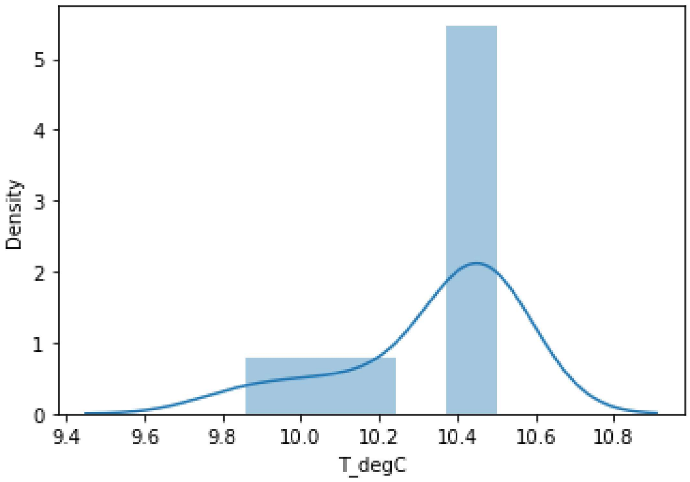
Out[29]:
```
<seaborn.axisgrid.PairGrid at 0x190867f1130>
```



In [30]:
```
sns.distplot(a[ 'T_degC'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarnin
g: `distplot` is a deprecated function and will be removed in a future version. Please
adapt your code to use either `displot` (a figure-level function with similar flexibil
ity) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

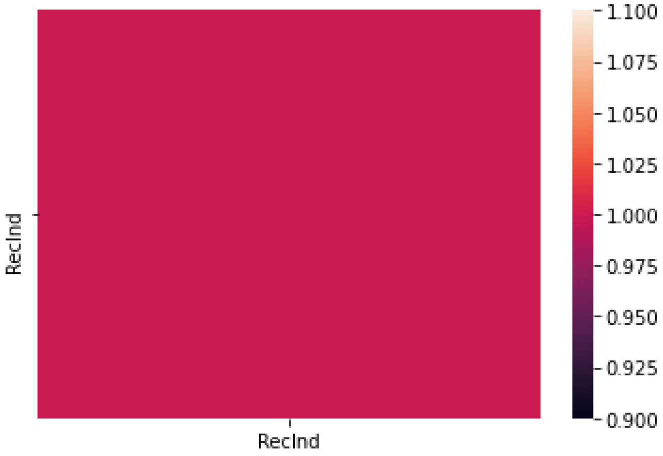
Out[30]:
```
<AxesSubplot:xlabel='T_degC', ylabel='Density'>
```



In [32]:
```
x1=a[['RecInd']]
```

In [33]:
```
sns.heatmap(x1.corr())
```

Out[33]:
```
<AxesSubplot:>
```



In [34]:
```
x=a[['RecInd']]
y=a['T_degC']
```

In [35]:
```
from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [36]:

```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[36]:

```
LinearRegression()
```

In [37]:

```
print(lr.intercept_)
```

```
10.503125
```

In [38]:

```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[38]:

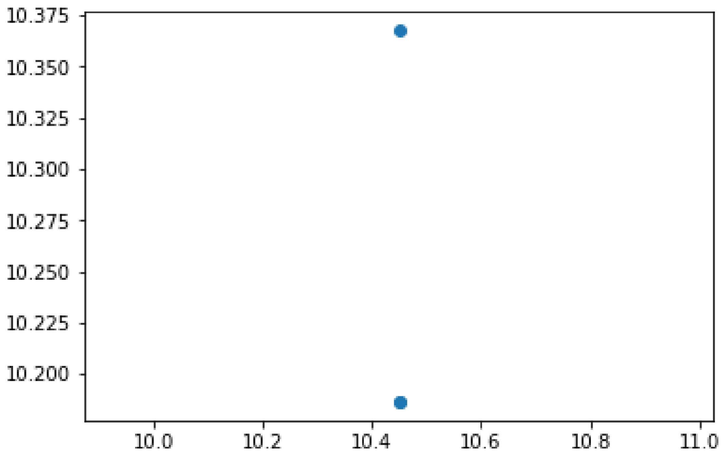
	Co-efficient
RecInd	-0.045208

In [39]:

```
prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[39]:

```
<matplotlib.collections.PathCollection at 0x190869934c0>
```



In [40]:

```
print(lr.score(x_test,y_test))
```

```
0.0
```

In [41]:

```
from sklearn.linear_model import Ridge,Lasso
```

In [42]:

```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[42]:

```
Ridge(alpha=10)
```

In [43]:

```
rr.score(x_test,y_test)
```

Out[43]:

```
0.0
```

In [44]:

```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[44]:

```
Lasso(alpha=10)
```

```
In [45]: la.score(x_test,y_test)

Out[45]: 0.0
```

```
In [46]: from sklearn.linear_model import ElasticNet
         en=ElasticNet()
         en.fit(x_train,y_train)

Out[46]: ElasticNet()
```

```
In [47]: print(en.coef_)

         [-0.]
```

```
In [48]: print(en.intercept_)

         10.290000000000001
```

```
In [49]: print(en.predict(x_test))

         [10.29 10.29 10.29]
```

```
In [50]: en.score(x_test,y_test)

Out[50]: 0.0
```

```
In [51]: from sklearn import metrics
```

```
In [52]: print("Mean Absolute Error",metrics.mean_absolute_error(y_test,prediction))

         Mean Absolute Error 0.20305555555555385
```

```
In [53]: print("Mean Squared Error",metrics.mean_squared_error(y_test,prediction))

         Mean Squared Error 0.048498379629628996
```

```
In [54]: print(" Root Mean Squared Error",np.sqrt(metrics.mean_squared_error(y_test,prediction))

          Root Mean Squared Error 0.22022347656330601
```