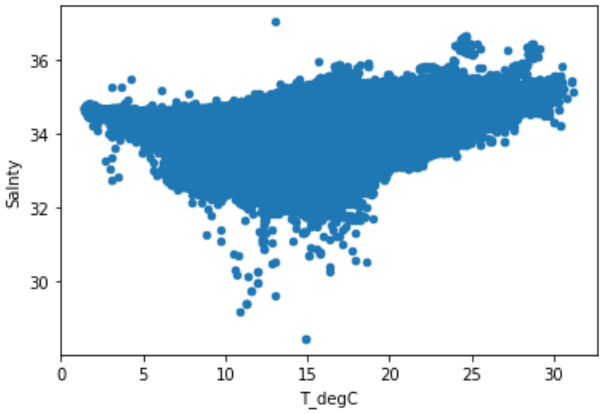
In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import random
import math
```

In [4]:
```
bottle = pd.read_csv('bottles.csv')
```

In [7]:
```
bottle.plot.scatter(x='T_degC', y='Salnty')
```

Out[7]: <AxesSubplot:xlabel='T_degC', ylabel='Salnty'>

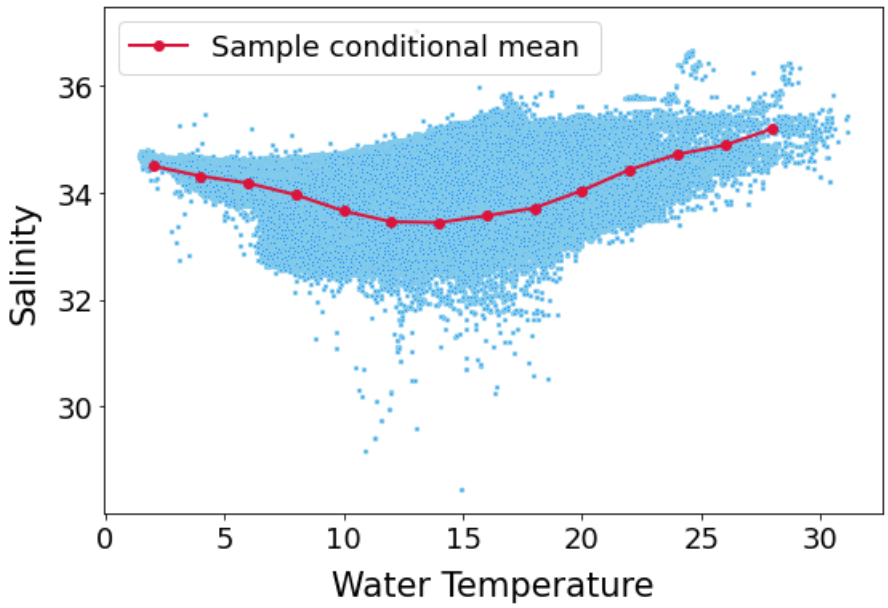


In [72]:
```
salnty = np.array(bottle['Salnty'])
temp = np.array(bottle['T_degC'])

salnty_clean =salnty[~np.isnan(temp)]
temp_clean = temp[~np.isnan(temp)]

temp_clean = temp_clean[~np.isnan(salnty_clean)]
salnty_clean = salnty_clean[~np.isnan(salnty_clean)]
```

In [113]:
```
max_val = np.max(temp[~np.isnan(temp)])
width_bin = 2
fig = plt.figure(figsize = (9,6))
plt.scatter(temp,salnty, s=5, c="dodgerblue", marker='o', edgecolor="skyblue")

# TODO: create bins from 0 to the maximum to discretize continous temperatures
grid = list(range(0,math.ceil(max_val),2))

# TODO: Compute the conditional expection of sanlinity given temperture
cond_average_salnty = np.zeros(len(grid))
cond_average_salnty_ind = np.zeros(len(grid))
cond_average_salnty_lists = dict()

i=0
while i < len(salnty_clean):

    index= math.floor(temp_clean[i]/2)
    cond_average_salnty[index]+=salnty_clean[i]
    cond_average_salnty_ind[index]+=1

    if index in cond_average_salnty_lists.keys():
        cond_average_salnty_lists[index].append(temp_clean[i])

    else:
        cond_average_salnty_lists[index]= [temp_clean[i]]

    i+=1

i=0
```

```
while i<len (cond_average_salnty_ind):
    cond_average_salnty[i] = cond_average_salnty[i]/cond_average_salnty_ind[i]
    i+=1



plt.plot(grid[1:-1],cond_average_salnty[1:-1],'-o',lw=2,color='crimson', label="Sample conditi
plt.ylabel("Salinity", fontsize=21,labelpad=10)
plt.xlabel("Water Temperature", fontsize=21,labelpad=10)

plt.legend(fontsize=18)
# plt.xlim(-5,30)
# plt.ylim(-12,23)
plt.xticks(fontsize=18)
plt.yticks(fontsize=18)
plt.gcf().subplots_adjust(bottom=0.15)
plt.gcf().subplots_adjust(left=0.15)
plt.savefig('conditional_expectation.pdf')
```

```
<ipython-input-113-9b9c79b847ec>:47: UserWarning: Creating legend with loc="best" can be slo
w with large amounts of data.
  plt.savefig('conditional_expectation.pdf')
<ipython-input-113-9b9c79b847ec>:47: UserWarning: Creating legend with loc="best" can be slo
w with large amounts of data.
  plt.savefig('conditional_expectation.pdf')
C:\Users\jonah\anaconda3\lib\site-packages\IPython\core\pylabtools.py:132: UserWarning: Crea
ting legend with loc="best" can be slow with large amounts of data.
  fig.canvas.print_figure(bytes_io, **kw)
```

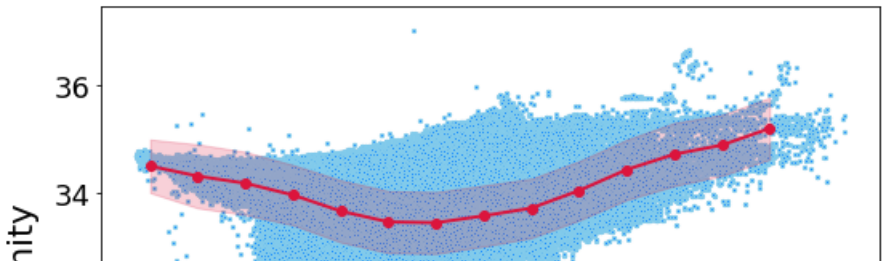


In [114]:

```
# TODO: Compute the conditional standard deviation of sanlinity given temperture
cond_std_salnty = np.zeros(len(grid))

i=0
while i<len (cond_average_salnty_ind):
    cond_average_salnty_lists[i] = np.std(cond_average_salnty_lists[i] )
    cond_std_salnty[i]=cond_average_salnty_lists[i]
    i+=1


fig = plt.figure(figsize = (9,6))
plt.scatter(temp,salnty, s=5, c="dodgerblue", marker='o', edgecolor="skyblue")


plt.plot(grid[1:-1],cond_average_salnty[1:-1],'-o',lw=2,color='crimson', label="Sample conditi
plt.fill_between(grid[1:-1], cond_average_salnty[1:-1]-cond_std_salnty[1:-1],
                 cond_average_salnty[1:-1]+cond_std_salnty[1:-1], color='crimson', alpha=0.2, la

plt.ylabel("Salinity", fontsize=21,labelpad=10)
```

```
plt.xlabel("Water Temperature", fontsize=21,labelpad=10)

plt.legend(fontsize=18)
plt.xticks(fontsize=18)
plt.yticks(fontsize=18)
plt.gcf().subplots_adjust(bottom=0.15)
plt.gcf().subplots_adjust(left=0.15)
plt.savefig('conditional_expectation_w_std.pdf')
```

```
<ipython-input-114-266567546691>:27: UserWarning: Creating legend with loc="best" can be slo
w with large amounts of data.
  plt.savefig('conditional_expectation_w_std.pdf')
<ipython-input-114-266567546691>:27: UserWarning: Creating legend with loc="best" can be slo
w with large amounts of data.
  plt.savefig('conditional_expectation_w_std.pdf')
C:\Users\jonah\anaconda3\lib\site-packages\IPython\core\pylabtools.py:132: UserWarning: Crea
ting legend with loc="best" can be slow with large amounts of data.
  fig.canvas.print_figure(bytes_io, **kw)
```



In [107]:
```
cond_average_salnty = np.zeros(len(grid))
cond_average_salnty_ind = np.zeros(len(grid))
cond_average_salnty_lists = dict()

i=0
while i < len(salnty_clean):

    index= math.floor(temp_clean[i]/2)
    cond_average_salnty[index]+=salnty_clean[i]
    cond_average_salnty_ind[index]+=1

    if index in cond_average_salnty_lists.keys():
        cond_average_salnty_lists[index].append(temp_clean[i])

    else:
        cond_average_salnty_lists[index]= [temp_clean[i]]

    i+=1
```

In [104]:
```
i=0
while i<len (cond_average_salnty_ind):
    cond_average_salnty_lists[i] = np.std(cond_average_salnty_lists[i] )
    i+=1

i=0
while i<len(cond_std_salnty):

    cond_std_salnty[i]=cond_average_salnty_lists[i]
    i+=1
```

In [ ]: