Introduction to Python and Jupyter Notebooks

Data Manipulation using Pandas

This is the reference component to the NMSU Pandas workshop 2018.

Set Up

Run the following cell to load (1) the pandas library and (2) the dataset we will be working with today (`Hint: run the code by pushing shift + enter`)

In [1]:
```
import pandas as pd
pols = pd.read_csv("pollutants.csv", dtype = object)
```

Selecting specific values of a `pandas` `DataFrame` or `Series` to work on is an implicit step in almost any data operation you'll run. Hence a solid understanding of how to slice and dice a dataset is vital.

We specified dtype = object earlier because there are many different data types present in this data frame- yet all of them our objects. Now, we need to convert some columns to numeric for manipulation with the numpy library (Pandas uses numpy, numpy uses numbers)

In [2]:
```
cols = ['fiscal_year', 'result']
pols[cols] = pols[cols].apply(pd.to_numeric, errors = 'coerce')
```

Native accessors

Native Python objects provide many good ways of indexing data. `pandas` carries all of these over, which helps make it easy to start with.

Consider this `DataFrame` (Hint: run the code by pushing shift + enter)

In [3]:
```
pols
```

Out[3]:

	coastal_ecological_area	collection_date	fiscal_year	general_location
0	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
1	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
2	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
3	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
4	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
5	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
6	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
7	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
8	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
9	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
10	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
11	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
12	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
13	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
14	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
15	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
16	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
17	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
18	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea

	coastal_ecological_area	collection_date	fiscal_year	general_location
19	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
20	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
21	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
22	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
23	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
24	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
25	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
26	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
27	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
28	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
29	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea
...	...	...	...	...
769896	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769897	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769898	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769899	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769900	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769901	NaN	30AUG2013:00:00:00.000	2013.0	Tinian

	coastal_ecological_area	collection_date	fiscal_year	general_location
769902	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769903	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769904	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769905	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769906	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769907	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769908	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769909	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769910	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769911	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769912	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769913	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769914	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769915	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769916	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769917	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769918	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769919	NaN	30AUG2013:00:00:00.000	2013.0	Tinian

	coastal_ecological_area	collection_date	fiscal_year	general_location
769920	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769921	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769922	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769923	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769924	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769925	NaN	27AUG2013:00:00:00.000	2013.0	Tinian

769926 rows × 22 columns

In Python we can access the property of an object by accessing it as an attribute. A `book` object, for example, might have a `title` property, which we can access by calling `book.title`. Columns in a `pandas` `DataFrame` work in much the same way.

Hence to access the `fiscal_year` property of our `pols` we can use:

In [4]:
```
pols.fiscal_year
```

```
Out[4]: 0         2015.0
        1         2015.0
        2         2015.0
        3         2015.0
        4         2015.0
        5         2015.0
        6         2015.0
        7         2015.0
        8         2015.0
        9         2015.0
        10        2015.0
        11        2015.0
        12        2015.0
        13        2015.0
        14        2015.0
        15        2015.0
        16        2015.0
        17        2015.0
        18        2015.0
        19        2015.0
        20        2015.0
        21        2015.0
        22        2015.0
        23        2015.0
        24        2015.0
        25        2015.0
        26        2015.0
        27        2015.0
        28        2015.0
        29        2015.0
                   ...
        769896    2013.0
        769897    2013.0
        769898    2013.0
        769899    2013.0
        769900    2013.0
        769901    2013.0
        769902    2013.0
        769903    2013.0
        769904    2013.0
        769905    2013.0
        769906    2013.0
        769907    2013.0
        769908    2013.0
        769909    2013.0
        769910    2013.0
        769911    2013.0
        769912    2013.0
        769913    2013.0
        769914    2013.0
        769915    2013.0
        769916    2013.0
        769917    2013.0
        769918    2013.0
        769919    2013.0
        769920    2013.0
        769921    2013.0
```

```
769922    2013.0
769923    2013.0
769924    2013.0
769925    2013.0
Name: fiscal_year, Length: 769926, dtype: float64
```

If we have a `dict` object in Python, we can access its values using the indexing (`[]`) operator. Again, we can do the same with `pandas` `DataFrame` columns. It "just works":

In [5]:
```
pols['fiscal_year']
```

```
Out[5]: 0         2015.0
        1         2015.0
        2         2015.0
        3         2015.0
        4         2015.0
        5         2015.0
        6         2015.0
        7         2015.0
        8         2015.0
        9         2015.0
        10        2015.0
        11        2015.0
        12        2015.0
        13        2015.0
        14        2015.0
        15        2015.0
        16        2015.0
        17        2015.0
        18        2015.0
        19        2015.0
        20        2015.0
        21        2015.0
        22        2015.0
        23        2015.0
        24        2015.0
        25        2015.0
        26        2015.0
        27        2015.0
        28        2015.0
        29        2015.0
                   ...
        769896    2013.0
        769897    2013.0
        769898    2013.0
        769899    2013.0
        769900    2013.0
        769901    2013.0
        769902    2013.0
        769903    2013.0
        769904    2013.0
        769905    2013.0
        769906    2013.0
        769907    2013.0
        769908    2013.0
        769909    2013.0
        769910    2013.0
        769911    2013.0
        769912    2013.0
        769913    2013.0
        769914    2013.0
        769915    2013.0
        769916    2013.0
        769917    2013.0
        769918    2013.0
        769919    2013.0
        769920    2013.0
        769921    2013.0
```

```
769922    2013.0
769923    2013.0
769924    2013.0
769925    2013.0
Name: fiscal_year, Length: 769926, dtype: float64
```

These are the two ways of selecting a specific columnar `Series` out of a `pandas` `DataFrame`. Neither of them is more or less syntactically valid than the other, but the indexing operator `[]` does have the advantage that it can handle column names with reserved characters in them (e.g. if we had a column named `fiscal year` instead of `fiscal_year`, `pols.fiscal year` wouldn't work). For the same reason, people tend to name objects with underscores instead of spaces

Doesn't a `pandas` `Series` look kind of like a fancy `dict`? It pretty much is, so it's no surprise that, to drill down to a single specific value, we need only use the indexing operator `[]` once more:

```
In [6]: pols['fiscal_year'][0]
```

```
Out[6]: 2015.0
```

Index-based selection

The indexing operator and attribute selection are nice because they work just like they do in the rest of the Python ecosystem. As a novice, this makes them easy to pick up and use. However, `pandas` has its own accessor operators, `loc` and `iloc`. For more advanced operations, these are the ones you're supposed to be using.

`pandas` indexing works in one of two paradigms. The first is **index-based selection**: selecting data based on its numerical position in the data. `iloc` follows this paradigm.

To select the first row of data in this `DataFrame`, we may use the following:

In [7]:
```
pols.iloc[0]
```

Out[7]:
```
coastal_ecological_area                                    NaN
collection_date                        18AUG2015:00:00:00.000
fiscal_year                                              2015
general_location                                  Chukchi Sea
latitude                                                   NaN
longitude                                                  NaN
matrix                                                Sediment
method                                              B&B SOP1006
nst_sample_id                             BA2015AKC-15-005bSED
nst_site                                           AKC_15-005
parameter                         1,6,7-Trimethylnaphthalene
parameter_name                                             NaN
qualifier                                       Below the MDL
region_name                                                NaN
result                                                   0.104
scientific_name                                       Sediment
source_file               AlaskaArcticStudy_Organics_Sediment.csv
specific_location                                    Peard Bay
state_name                                                 NaN
stratum                                                    NaN
study_name                                  Alaska Arctic Study
units                                                ng/dry g
Name: 0, dtype: object
```

Both `loc` and `iloc` are row-first, column-second. This is the opposite of what we do in native Python, which is column-first, row-second.

This means that it's marginally easier to retrieve rows, and marginally harder to get retrieve columns. To get a column with `iloc`, we can do the following (`We're selecting 3 to get the general_location column- Pandas indexing starts with 0 not 1`):

In [8]:
```
pols.iloc[:, 3]
```

```
Out[8]: 0         Chukchi Sea
        1         Chukchi Sea
        2         Chukchi Sea
        3         Chukchi Sea
        4         Chukchi Sea
        5         Chukchi Sea
        6         Chukchi Sea
        7         Chukchi Sea
        8         Chukchi Sea
        9         Chukchi Sea
        10        Chukchi Sea
        11        Chukchi Sea
        12        Chukchi Sea
        13        Chukchi Sea
        14        Chukchi Sea
        15        Chukchi Sea
        16        Chukchi Sea
        17        Chukchi Sea
        18        Chukchi Sea
        19        Chukchi Sea
        20        Chukchi Sea
        21        Chukchi Sea
        22        Chukchi Sea
        23        Chukchi Sea
        24        Chukchi Sea
        25        Chukchi Sea
        26        Chukchi Sea
        27        Chukchi Sea
        28        Chukchi Sea
        29        Chukchi Sea
                     ...
        769896      Tinian
        769897      Tinian
        769898      Tinian
        769899      Tinian
        769900      Tinian
        769901      Tinian
        769902      Tinian
        769903      Tinian
        769904      Tinian
        769905      Tinian
        769906      Tinian
        769907      Tinian
        769908      Tinian
        769909      Tinian
        769910      Tinian
        769911      Tinian
        769912      Tinian
        769913      Tinian
        769914      Tinian
        769915      Tinian
        769916      Tinian
        769917      Tinian
        769918      Tinian
        769919      Tinian
        769920      Tinian
        769921      Tinian
```

```
769922         Tinian
769923         Tinian
769924         Tinian
769925         Tinian
Name: general_location, Length: 769926, dtype: object
```

On its own the `:` operator, which also comes from native Python, means "everything". When combined with other selectors, however, it can be used to indicate a range of values. For example, to select the `country` column from just the first, second, and third row, we would do:

In [9]:
```
pols.iloc[:3, 3]
```

Out[9]:
```
0    Chukchi Sea
1    Chukchi Sea
2    Chukchi Sea
Name: general_location, dtype: object
```

Or, to select just the second and third entries, we would do:

In [10]:
```
pols.iloc[1:3, 3]
```

Out[10]:
```
1    Chukchi Sea
2    Chukchi Sea
Name: general_location, dtype: object
```

It's also possible to pass a list:

In [11]:
```
pols.iloc[[0, 1, 2], 3]
```

Out[11]:
```
0    Chukchi Sea
1    Chukchi Sea
2    Chukchi Sea
Name: general_location, dtype: object
```

Finally, it's worth knowing that negative numbers can be used in selection. This will start counting forwards from the *end* of the values. So for example here are the last five elements of the dataset.

In [12]:
```
pols.iloc[-5:]
```

Out[12]:

	coastal_ecological_area	collection_date	fiscal_year	general_location
769921	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769922	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769923	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769924	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769925	NaN	27AUG2013:00:00:00.000	2013.0	Tinian

5 rows × 22 columns

Label-based selection

The second paradigm for attribute selection is the one followed by the `loc` operator: **label-based selection**. In this paradigm it's the data index value, not its position, which matters.

For example, to get the first entry in `pols`, we would now do the following:

In [13]:
```
pols.loc[0, 'fiscal_year']
```

Out[13]:
```
2015.0
```

`iloc` is conceptually simpler than `loc` because it ignores the dataset's indices. When we use `iloc` we treat the dataset like a big matrix (a list of lists), one that we have to index into by position. `loc`, by contrast, uses the information in the indices to do its work. Since your dataset usually has meaningful indices, it's usually easier to do things using `loc` instead. For example, here's one operation that's much easier using `loc`:

In [14]:
```
pols.loc[:, ['fiscal_year', 'parameter', 'result', 'units']]
```

Out[14]:

	fiscal_year	parameter	result	units
0	2015.0	1,6,7-Trimethylnaphthalene	0.1040	ng/dry g
1	2015.0	1-Methyldibenzothiophene	0.0380	ng/dry g
2	2015.0	1-Methylfluorene	0.4470	ng/dry g
3	2015.0	1-Methylnaphthalene	0.5950	ng/dry g
4	2015.0	1-Methylphenanthrene	0.4600	ng/dry g
5	2015.0	18a-Oleanane	0.0000	ng/dry g
6	2015.0	2,6-Dimethylnaphthalene	0.3010	ng/dry g
7	2015.0	2-Methylanthracene	4.4500	ng/dry g
8	2015.0	2-Methylfluoranthene	0.2710	ng/dry g
9	2015.0	2-Methylnaphthalene	0.7470	ng/dry g
10	2015.0	2-Methylphenanthrene	0.6000	ng/dry g
11	2015.0	2/3-Methyldibenzothiophene	0.1470	ng/dry g
12	2015.0	3,6-Dimethylphenanthrene	0.1140	ng/dry g
13	2015.0	3-Methylphenanthrene	0.4270	ng/dry g
14	2015.0	4-Methyldibenzothiophene	0.2130	ng/dry g
15	2015.0	4/9-Methylphenanthrene	0.7260	ng/dry g
16	2015.0	Acenaphthene	0.0780	ng/dry g
17	2015.0	Acenaphthylene	0.0000	ng/dry g
18	2015.0	Anthracene	0.0000	ng/dry g
19	2015.0	Benz[a]anthracene	0.2770	ng/dry g
20	2015.0	Benzo(a)fluoranthene	0.0000	ng/dry g
21	2015.0	Benzo(b)fluorene	0.4010	ng/dry g
22	2015.0	Benzo[a]pyrene	0.2157	ng/dry g
23	2015.0	Benzo[b]fluoranthene	0.7361	ng/dry g
24	2015.0	Benzo[e]pyrene	0.5696	ng/dry g
25	2015.0	Benzo[g,h,i]perylene	0.6390	ng/dry g
26	2015.0	Benzo[k]fluoranthene	0.1222	ng/dry g
27	2015.0	Benzothiophene	0.0000	ng/dry g
28	2015.0	Biphenyl	0.3460	ng/dry g
29	2015.0	C1-Benzothiophene	0.0000	ng/dry g
...	...	...	...	...
769896	2013.0	PCB209	0.0000	ng/dry g

	fiscal_year	parameter	result	units
769897	2013.0	Monobutyltin	0.0000	ng Sn/dry g
769898	2013.0	1-Methylnaphthalene	0.1980	ng/dry g
769899	2013.0	Monobutyltin	0.0000	ng Sn/dry g
769900	2013.0	Perylene	0.0000	ng/dry g
769901	2013.0	1-Methylnaphthalene	0.0000	ng/dry g
769902	2013.0	PCB195_208	0.0000	ng/dry g
769903	2013.0	PCB74_61	0.0000	ng/dry g
769904	2013.0	Phenanthrene	0.3230	ng/dry g
769905	2013.0	PCB74_61	0.0000	ng/dry g
769906	2013.0	Phenanthrene	0.3660	ng/dry g
769907	2013.0	Monobutyltin	0.0000	ng Sn/dry g
769908	2013.0	Perylene	0.1780	ng/dry g
769909	2013.0	Monobutyltin	0.0000	ng Sn/dry g
769910	2013.0	Perylene	0.0000	ng/dry g
769911	2013.0	1-Methylnaphthalene	0.2330	ng/dry g
769912	2013.0	1-Methylnaphthalene	0.2000	ng/dry g
769913	2013.0	1-Methylnaphthalene	0.0000	ng/dry g
769914	2013.0	Phenanthrene	1.0920	ng/dry g
769915	2013.0	PCB206	0.0000	ng/dry g
769916	2013.0	Perylene	0.1090	ng/dry g
769917	2013.0	Phenanthrene	0.5820	ng/dry g
769918	2013.0	PCB31	0.0000	ng/dry g
769919	2013.0	1-Methylnaphthalene	0.1510	ng/dry g
769920	2013.0	Monobutyltin	0.0000	ng Sn/dry g
769921	2013.0	1,6,7-Trimethylnaphthalene	0.0000	ng/dry g
769922	2013.0	1-Methylnaphthalene	0.2430	ng/dry g
769923	2013.0	1-Methylnaphthalene	0.1710	ng/dry g
769924	2013.0	1-Methylnaphthalene	0.2060	ng/dry g
769925	2013.0	PCB8_5	0.0000	ng/dry g

769926 rows × 4 columns

When choosing or transitioning between `loc` and `iloc`, there is one "gotcha" worth keeping in mind, which is that the two methods use slightly different indexing schemes.

`iloc` uses the Python stdlib indexing scheme, where the first element of the range is included and the last one excluded. So 0:10 will select entries 0,...,9. `loc`, meanwhile, indexes inclusively. So 0:10 will select entries 0,...,10.

Why the change? Remember that loc can index any stdlib type: strings, for example. If we have a DataFrame with index values `Apples, ..., Potatoes, ...`, and we want to select "all the alphabetical fruit choices between Apples and Potatoes", then it's a heck of a lot more convenient to index `df.loc['Apples':'Potatoes']` than it is to index something like `df.loc['Apples', 'Potatoet]` (`t` coming after `s` in the alphabet).

This is particularly confusing when the `DataFrame` index is a simple numerical list, e.g. `0,...,1000`. In this case `df.iloc[0:1000]` will return 1000 entries, while `df.loc[0:1000]` return 1001 of them! To get 1000 elements using `loc`, you will need to go one lower and ask for `df.iloc[0:999]`. Earlier versions of this tutorial did not point this out explicitly, leading to a lot of user confusion on some of the related answers, so we've included this note here explaining this issue.

Otherwise, the semantics of using `loc` are the same as those for `iloc`.

Manipulating the index

Label-based selection derives its power from the labels in the index. Critically, the index we use is not immutable. We can manipulate the index in any way we see fit.

The `set_index` method can be used to do the job. Here is what happens when we `set_index` to the `result` field:

In [15]:
```
pols.set_index("result", inplace = True)
pols
```

Out[15]:

	coastal_ecological_area	collection_date	fiscal_year	general_location	
result					
0.1040	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	
0.0380	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	
0.4470	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	
0.5950	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	
0.4600	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	
0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	
0.3010	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	
4.4500	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	
0.2710	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	
0.7470	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	
0.6000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	
0.1470	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	
0.1140	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	
0.4270	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	
0.2130	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	
0.7260	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	
0.0780	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	
0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	

	coastal_ecological_area	collection_date	fiscal_year	general_location	I
result					
0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	I
0.2770	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	I
0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	I
0.4010	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	I
0.2157	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	I
0.7361	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	I
0.5696	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	I
0.6390	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	I
0.1222	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	I
0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	I
0.3460	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	I
0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Sea	I
...	...	...	...	...	.
0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian	I
0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian	I
0.1980	NaN	30AUG2013:00:00:00.000	2013.0	Tinian	I
0.0000	NaN	28AUG2013:00:00:00.000	2013.0	Tinian	I
0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian	I

result	coastal_ecological_area	collection_date	fiscal_year	general_location	I
0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian	I
0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian	I
0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian	I
0.3230	NaN	28AUG2013:00:00:00.000	2013.0	Tinian	I
0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian	I
0.3660	NaN	30AUG2013:00:00:00.000	2013.0	Tinian	I
0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian	I
0.1780	NaN	28AUG2013:00:00:00.000	2013.0	Tinian	I
0.0000	NaN	28AUG2013:00:00:00.000	2013.0	Tinian	I
0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian	I
0.2330	NaN	28AUG2013:00:00:00.000	2013.0	Tinian	I
0.2000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian	I
0.0000	NaN	28AUG2013:00:00:00.000	2013.0	Tinian	I
1.0920	NaN	27AUG2013:00:00:00.000	2013.0	Tinian	I
0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian	I
0.1090	NaN	27AUG2013:00:00:00.000	2013.0	Tinian	I
0.5820	NaN	27AUG2013:00:00:00.000	2013.0	Tinian	I

	coastal_ecological_area	collection_date	fiscal_year	general_location	I
result					
0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian	I
0.1510	NaN	30AUG2013:00:00:00.000	2013.0	Tinian	I
0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian	I
0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian	I
0.2430	NaN	30AUG2013:00:00:00.000	2013.0	Tinian	I
0.1710	NaN	27AUG2013:00:00:00.000	2013.0	Tinian	I
0.2060	NaN	30AUG2013:00:00:00.000	2013.0	Tinian	I
0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian	I

769926 rows × 21 columns

This index, however, is not so informative. If you're going to label the rows of your DataFrame, it would be good to label them in a manner which is more meaningful than the current one. Let's restore the original index

In [16]:
```
pols.reset_index(inplace = True)
pols
```

Out[16]:

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
0	0.1040	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
1	0.0380	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
2	0.4470	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
3	0.5950	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
4	0.4600	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
5	0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
6	0.3010	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
7	4.4500	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
8	0.2710	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
9	0.7470	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
10	0.6000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
11	0.1470	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
12	0.1140	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
13	0.4270	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
14	0.2130	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
15	0.7260	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
16	0.0780	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
17	0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
18	0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
19	0.2770	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
20	0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
21	0.4010	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
22	0.2157	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
23	0.7361	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
24	0.5696	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
25	0.6390	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
26	0.1222	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
27	0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
28	0.3460	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
29	0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
...	...	...	...	...	...
769896	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769897	0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769898	0.1980	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769899	0.0000	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769900	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769901	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
769902	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769903	0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769904	0.3230	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769905	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769906	0.3660	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769907	0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769908	0.1780	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769909	0.0000	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769910	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769911	0.2330	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769912	0.2000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769913	0.0000	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769914	1.0920	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769915	0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769916	0.1090	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769917	0.5820	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769918	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769919	0.1510	NaN	30AUG2013:00:00:00.000	2013.0	Tinian

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
769920	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769921	0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769922	0.2430	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769923	0.1710	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769924	0.2060	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769925	0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian

769926 rows × 22 columns

Data Evaluation

So far we've been indexing various strides of data, using structural properties of the `DataFrame` itself. To do *interesting* things with the data, however, we want to find some `answers`

Usually I begin by finding as much information about the data as I can. For example, what are the different general locations that are present in the dataset? This requires the numpy function `unique()`

In [17]:
```
pols.general_location.unique()
```

```
Out[17]: array(['Chukchi Sea', 'Beaufort Sea', 'Kachemak Bay- Western Subtidal',
                'Homer Harbor', 'Kachemak Bay - Western Flats', 'Port Graham',
                'Kachemak Bay-Eastern Flats', 'Kachemak Bay-Eastern Subtidal',
                'Kachemak Bay', 'Cook Inlet', 'Kachemak Bay South Bays', nan,
                'Nanwalek', 'Seldovia', 'Seldovia Bay', 'Barnes Sound',
                'North Miami', 'South Biscayne Bay', 'Manatee Bay',
                'Rickenbacker Causeway', 'Port of Miami', 'Miami Shores',
                'Miami Beach', 'North Miami Beach', 'North Bay', 'C-111 Canal',
                'Coral Gables Canal', 'Miami River', 'South Bay', 'Tamiani Canal',
                'North Canal', 'Mowry Canal', 'Military Canal',
                'Little Card Sound', 'Card Sound', 'Key Largo', 'Seybold Canal',
                'Mangrove Point', 'Snapper Creek Canal', 'Princeton Canal',
                'Florida City Canal', 'Charleston Channel', 'Dorchester Bay',
                'Fort Point', 'Massachusetts Bay', 'Weymouth Fore River',
                'Orient Heights', 'Upper Chelsea River', 'Quincy Bay',
                'Sculpin Lodge', 'Boston Wharves', 'Lower Chelsea River',
                'Mystic River', 'Nantasket Roads', 'Hull Bay', 'Northwest Harbor',
                'Channel Mouth', 'Snake Island', 'Hingham Bay', 'Boston Channel',
                'Reserved Channel', 'Chelsea Point', 'Charleston Harbor',
                'Wando River', 'Lower Cooper River', 'Ashley River',
                'Middle Cooper River', 'Upper Cooper River', 'Patriots Point',
                'Lower Wando River', 'Downtown Charleston- Charleston Aquarium',
                'Lower Harbor - Plum Island', 'Lower Harbor', 'Upper Wando River',
                'Wallace Creek', 'Romney Landfill', 'Palms Connector',
                'Shipyard Creek', 'Central Chesapeake Bay- Strata 36',
                'Central Chesapeake Bay- Strata 37',
                'Central Chesapeake Bay- Strata 33',
                'Lower Chesapeake Bay- Strata 41',
                'Lower Chesapeake Bay- Strata 42',
                'Central Chesapeake Bay- Strata 32',
                'Lower Chesapeake Bay- Strata 43',
                'Central Chesapeake Bay- Strata 31',
                'Central Chesapeake Bay- Strata 40',
                'Central Chesapeake Bay- Strata 35',
                'Central Chesapeake Bay- Strata 39',
                'Central Chesapeake Bay- Strata 34',
                'Central Chesapeake Bay- Strata 30',
                'Central Chesapeake Bay- Strata 29',
                'Central Chesapeake Bay- Strata 38',
                'Central Chesapeake Bay- Strata 28',
                'Central Chesapeake Bay- Strata 23',
                'Central Chesapeake Bay- Strata 27',
                'Upper Chesapeake Bay- Strata 16',
                'Central Chesapeake Bay- Strata 19',
                'Upper Chesapeake Bay- Strata 13',
                'Upper Chesapeake Bay- Strata 12',
                'Central Chesapeake Bay- Strata 18',
                'Upper Chesapeake Bay- Strata 17',
                'Central Chesapeake Bay- Strata 25',
                'Central Chesapeake Bay- Strata 21',
                'Central Chesapeake Bay- Strata 24',
                'Upper Chesapeake Bay- Strata 14',
                'Central Chesapeake Bay- Strata 26',
                'Upper Chesapeake Bay- Strata 15',
                'Central Chesapeake Bay- Strata 22',
                'Central Chesapeake Bay- Strata 20',
                'Lower Chesapeake Bay- Strata 58',
```

```
'Lower Chesapeake Bay- Strata 49',
'Lower Chesapeake Bay- Strata 50',
'Lower Chesapeake Bay- Strata 57',
'Lower Chesapeake Bay- Strata 44',
'Lower Chesapeake Bay- Strata 53',
'Lower Chesapeake Bay- Strata 51',
'Lower Chesapeake Bay- Strata 47',
'Lower Chesapeake Bay- Strata 54',
'Lower Chesapeake Bay- Strata 48',
'Lower Chesapeake Bay- Strata 55',
'Lower Chesapeake Bay- Strata 56',
'Lower Chesapeake Bay- Strata 52',
'Lower Chesapeake Bay- Strata 45',
'Lower Chesapeake Bay- Strata 46',
'Lower Chesapeake Bay- Strata 65',
'Lower Chesapeake Bay- Strata 64',
'Lower Chesapeake Bay- Strata 59',
'Lower Chesapeake Bay- Strata 63',
'Lower Chesapeake Bay- Strata 61',
'Lower Chesapeake Bay- Strata 60',
'Lower Chesapeake Bay- Strata 62',
'Upper Chesapeake Bay- Strata 8', 'Upper Chesapeake Bay- Strata 6',
'Upper Chesapeake Bay- Strata 4',
'Upper Chesapeake Bay- Strata 10',
'Upper Chesapeake Bay- Strata 5',
'Upper Chesapeake Bay- Strata 11',
'Upper Chesapeake Bay- Strata 1', 'Upper Chesapeake Bay- Strata 9',
'Upper Chesapeake Bay- Strata 3', 'Upper Chesapeake Bay- Strata 7',
'Upper Chesapeake Bay- Strata 2', 'Norfolk Harbor',
'Baltimore Harbor', 'Virginia Beach', 'South River', 'Norfolk',
'York River', 'Eastern Central Delaware Bay',
'North of Philadelphia', 'Bulkhead Shoal/Pea Patch Is.',
'Rehoboth', 'W. of Baker/Liston R.', 'E. of Baker/Liston R.',
'Cherry Island Range', 'Salem Cove/Ready Is.',
'Little Tinicum & Chester Is.', 'Central Delaware Bay', 'Bethany',
'East side - mouth of Del Bay', 'Phily & Camden Waterfront',
'Cape Henlopen - outside', 'Bombay Hook - small estuaries',
'Marcus Hook Bar', 'St. Jones River - small estuaries',
'Cherry Island Flats', 'Unimpounded wetlands',
'South of Philadelphia', 'South of Trenton',
'West side - mouth of Del Bay', 'Cape May - outside',
'Blackbird Creek - small estuaries', 'Impounded wetlands',
'Dry Tortugas', 'Lower New York Harbor', 'Lower East River',
'Sandy Hook Bay', 'Newark Bay/Arthur Kill', 'Western Raritan Bay',
'Western Long Island Sound', 'Upper East River',
'Lower Hudson River', 'Upper New York Harbor',
'Outer Bay/New York Bight', 'Southern Raritan Bay',
'Central Raritan Bay', 'Lower Raritan River',
'Upper Leadenwah Creek', 'Lower Leadenwah Creek',
'Middle Leadenwah Creek', 'Echo Bay', 'Larchmont Harbor',
'New Haven Harbor', 'Housatonic River', 'Milford Harbor',
'Pelham Harbor', 'Eastchester Bay', 'Centerport Harbor',
'Southport Harbor', 'Little Neck Bay', 'Norwalk Harbor',
'Bridgeport Harbor', 'Oyster Bay', 'Cold Spring Harbor',
'Connecticut Harbor', 'Branford Harbor', 'Thames River',
'Northport Harbor', 'Stamford Harbor', 'Manhassett Bay',
'Boston Harbor', 'Area Between Bays', 'Cape Cod Bay',
```

```
'Stellwagen Bank', 'Stellwagen Basin', 'Reference site', 'Outfall',
'Boston Harbor Deer Island', 'Neuse River', 'Newark Bay',
'Savannah River - South Channel', 'Port Wentworth',
'Savannah River- Lower Channel', 'Upper Savannah River',
'Savannah River- Ocean Terminal',
'Savannah River- Stevens Terminal',
'Savannah River- Suspension Bridge', 'Kings Island Turning Basin',
'Finger Canal', 'Savannah River- Front Channel',
'Marsh Island Turning Basin', 'Kings Island Channel',
'Savannah River- Wrecks Channel', 'Whiteball Channel',
'Marsh Island Channel', 'Back River', 'Savannah River',
'St Lucie - Lower Estuary', 'St Lucie - North Fork',
'St Lucie - Middle Estuary', 'St Lucie - South Fork',
'St Lucie - Convergence Zone', 'Turtle River', 'St Simon Sound',
'Purvis Creek', 'Academy Creek', 'Terry Creek', 'Dupree Creek',
'East River', 'Upper Rabbit Island',
'Winyah Bay- Sampit Point Channel', 'Winyah Bay- Turning Basin',
'Upper Rabbit Isle', 'Lower Rabbit Isle', 'Winyah Bay Marina',
'Winyah Bay - Steel Mill', 'Apalachicola Bay',
'East Chochawhatchee Bay', 'Garnier Bayou', 'Cinco Bayou',
'La Grange Bayou', "Tom's Bayou", 'Hand Cove', 'Boggy Bayou',
'Dons Bayou', 'Central Chochtawhatchee Bay', 'Rocky Bayou',
'West Chochtawhatchee Bay', 'Destin Harbor', 'Alaqua Bayou',
'Joes Bayou', 'Florida Bay', 'Central Galveston Bay- West',
'Trinity Bay - Offshore', 'Central Galveston Bay- East',
'Scott Bay', 'East Bay', 'Lower San Jacinto Bay', 'Texas City',
'Clear Lake', 'Lower Galveston Bay', 'Upper Galveston Bay - West',
'West Bay', 'Upper Houston Ship Channel',
'Trinity Bay - Nearshore', 'Upper San Jacinto Bay',
'Upper Galveston Bay - East', 'Tabbs Bay',
'Galveston Island - Nearshore', 'Bolivar Peninsula - Offshore',
'Galveston Bay - Entrance', 'Bolivar Peninsula - Nearshore',
'Bolivar Roads', 'Galveston Island - Offshore', 'Bayou Chico',
'Pensacola Bay', 'Bayou Grande', 'Inner Harbor Channel',
'Escambia Bay', 'Blackwater Bay', 'Bayou Channel', 'Inner Harbor',
'Escambia', 'Warrington', 'Escambia River Mouth', 'Bayou Texar',
'Lower Bay', 'Central Bay', 'Floridatown', 'Rookery Bay',
'Sabine Lake', 'Sabine-Neches Canal', 'Sabine River',
'Neches River', 'Sabine Pass', 'Gulf of Mexico/Sabine Pass',
'Lower Watson', 'West-East Bay', 'Middle Watson', 'Watson Bayou',
'St Andrews', 'St Andrew Bay', 'Upper Watson', 'Smak Bayou',
'Pearl Bayou', 'Massalina Bayou', 'Mouth of Pearl Bayou',
'East-East Bay', 'Mouth of Watson Bayou', 'Gulfport/Bear Creek',
'Northern Hillsborough Bay', 'Western Old Tampa Bay',
'St Petersburg Harbors', 'Anna Maria Sound', 'Manatee River',
'Middle Tampa Bay', 'Boca Ciega Bay', 'Charlotte Harbor',
'Cockroach Bay', 'Lower Tampa Bay', 'Terra Ceia Bay',
'Old Tampa Bay', 'Hillsborough Bay', 'Bay South', 'Offshore West',
'Watershed', 'Offshore East', 'Offshore Central', 'Bay Central',
'Bay North', 'Central Guanica Bay', 'East Guanica Bay',
'West Guanica Bay', 'Guanica Bay', 'Jobos Bay - Inner Bay',
'Jobos Bay- Outer Bay',
'Jobos Bay- National Estuarine Research Reserve',
'Jobos Bay- Center Bay', 'Jobos Bay- Offshore', 'Bahia Moltavia',
'Corral', 'Media Luna', 'El Palo', 'Playa Santa', 'Guanica',
'Isla Guayacan', 'Margarita', 'Isla Manueyes', 'Enrique',
'St. Thomas', 'St. Thomas East End Reserve', 'Southwest Vieques',
```

```
'Northwest Vieques', 'South Southwest Vieques',
'Northeast Vieques', 'North Vieques', 'North Northwest Vieques',
'North Northeast Vieques', 'South Southeast Vieques',
'South Vieques', 'Southeast Vieques', 'Center California',
'North California', 'Monterey Bay', 'San Francisco',
'Between Ascenion and Pioneer Canyons', 'Southeast of Pt Sur',
'Ascencion Canyon', 'Soquel Canyon', 'Carmel Canyon', 'Ano Nuevo',
'Ano Nuevo Canyon', 'Pioneer Canyon', 'Monterey Canyon',
'Between Pt Lobos and Pt Sur',
'Between Soquel and Cabrillo Canyon', 'Lucia Canyon',
'Bodega Canyon', 'Newport Bay', 'La Jolla Canyon', 'Dana Point',
'Santa Margarita River', 'Carlsbad Canyon', 'Del Mar Beach',
'Mission Bay', 'Puget Sound', 'Port Susan',
'Middle Everett Harbor', 'Outer Everett Harbor',
'Inner Everett Harbor', 'Port Gardner',
'Southeast Commencement Bay', 'Henderson Inlet', 'Port of Olympia',
'Drayton Passage', 'Case Inlet', 'East Passage', 'Eld Inlet',
'Colvos Passage', 'Nisqually Reach', 'Totten Inlet', 'Carr Inlet',
'Hale Passage', 'Pickering Passage', 'Gig Harbor',
'East Anderson Island', 'Middle Waterway', 'South Boundary Bay',
'Semiahmoo Bay', 'Drayton Harbor', 'Blair Waterway',
'South West Bainbridge Island', 'West Point', 'Birch Bay',
'Snohomish River Delta', 'Penn Cove', 'Skagit Bay',
'West Harbor Island', 'Mid Elliott Bay', 'Dyes Inlet',
'Port Orchard', 'Budd Inlet', 'Hylebos Waterway',
'Quartermaster Harbor', 'Central Basin', 'Middle Saratoga Passage',
'Northeast Commencement Bay', 'Thea Hoss Waterway', 'Keyport',
'Outer Commencement Bay', 'Possession Sound',
'Central Bellingham Bay', 'Bellingham Bay, South Bellingham',
'Port Townsend', 'South Admirality Inlet',
'North Saratoga Passage', 'West Boundary Bay', 'Sinclair Inlet',
'Eagle Harbor', 'Quilcene Bay', 'East Harbor Island',
'North Bellingham Bay', 'West Downtown Bellingham', 'Rich Passage',
'Hood Canal - Central', 'Port Gamble Bay', 'Port Ludlow',
'Outer Padilla Bay', 'Inner Padilla Bay',
'East Downtown Bellingham', 'March Point', 'Outer Fidalgo Bay',
'Samish Bay/Bellingham Bay', 'Inner Fidalgo Bay', 'Oak Harbor',
'Cherry Point', 'South Saratoga Passage', 'Shoreline Elliott Bay',
'Dabob Bay', 'Oaland Bay', 'Port Madison',
'Port Washington Narrows', 'Outer Elliott Bay', 'Port of Shelton',
'South Port Townsend', 'Duwamish', 'Hood Canal - North',
'Liberty Bay', 'Hood Canal - South',
'North West Bainbridge Island', 'San Diego Bay',
'San Francisco Bay', 'San Pedro Bay', 'Elkhorn Slough, Seal Point',
'Elkhorn Slough, Seal Bend', 'Tijuana River', 'Corpus Christi',
'Lower Laguna Madre', 'Matagorda Bay', 'Espiritu Santo',
'Copano Bay', 'Unakwit Inlet', 'Mesquite Bay', 'Gulf of Alaska',
'Nahku Bay', 'Prince William Sound', 'Brazos River', 'Aransas Bay',
'San Antonio Bay', 'Port Valdez', 'Resurrection Bay',
'Eastern Prince William Sound', 'Western Prince William Sound',
'Nushagek Bay', 'Ketchikan', 'Ninilchik', 'Lake Erie',
'Lake Ontario', 'Niagara River', 'Lake St. Clair', 'Green Bay',
'Lake Michigan', 'Saginaw Bay', 'Lake Huron', 'Traverse Bay',
'Lake Superior', 'Galveston Bay', 'Caillou Lake',
'Choctawhatchee Bay', 'St. Andrew Bay', 'Mobile Bay', 'Tampa Bay',
'Naples Bay', 'Breton Sound', 'Everglades', 'Terrebonne Bay',
'Lake Borgne', 'Calcasieu Lake', 'Mississippi Sound',
```

```
'Apalachee Bay', 'Barataria Bay', 'Lake Pontchartrain',
'Atchafalaya Bay', 'Vermilion Bay', 'Cedar Key',
'Joseph Harbor Bayou', 'Mississippi River', 'Panama City',
'Suwannee River', 'East Cote Blanche', "Barber's Point",
'Honolulu Hrb.', 'Hawaii', 'Kauai', 'Little Tanaga Island',
'Adak Island', 'Rat Island', 'Atak Island', 'Unalaska Island',
'Amchitka Island', 'Atka Island', 'Attu Island', 'Glacier Bay',
'Sitka', 'Norton Sound', 'Bristol Bay',
'Northern Prince William Sound', 'St. Paul Island', 'Kinak Bay',
'Kukak Bay', 'Takli Island', 'Kaflia Bay', 'Amalik Bay',
'Hudson River', 'Hudson/Raritan Estuary', 'New York Bight',
'Lake Eric', 'Florida Keys', 'Midway', 'French Frigate Shoals',
'Pearl and Hermes Atoll', 'Kure', 'Maro Reef', 'Laysan',
'Southeast shore of Vieques', 'Northwest shore of Vieques',
'San Pablo Bay', "Faga'alu north bay", "Faga'alu south bay",
"Faga'alu Inner Bay", "Faga'alu stream", "Faga'alu Channel",
'Brazil', 'Costa Rica', 'Colombia', 'Honduras', 'Panama',
'El Salvador', 'Ecuador', 'Nicaragua', 'Argentina', 'Chile',
'Mexico', 'Uruguay', 'Belize', 'Aruba', 'Trinidad', 'Jamaica',
'Venezuela', 'Cuba', 'Peru', 'Tinian'], dtype=object)
```

Well, that's quite a list! How many different places, exactly? This requires the built-in python function `len()`

```
In [18]: len(pols.general_location.unique())
```

```
Out[18]: 613
```

Dealing with missing data

To perform numerical assessments or manipulations, we must account for missing data. This can be done in a variety of ways. The simplest, perhaps, is just to drop the rows for which the value is NaN.

Let's find the range of fiscal years in this data set. To do this, we must first drop the NA values like this:

In [19]:
```
pols.dropna(subset = ['fiscal_year'], inplace = True)
pols
```

Out[19]:

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
0	0.1040	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
1	0.0380	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
2	0.4470	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
3	0.5950	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
4	0.4600	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
5	0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
6	0.3010	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
7	4.4500	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
8	0.2710	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
9	0.7470	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
10	0.6000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
11	0.1470	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
12	0.1140	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
13	0.4270	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
14	0.2130	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
15	0.7260	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
16	0.0780	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
17	0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
18	0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
19	0.2770	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
20	0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
21	0.4010	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
22	0.2157	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
23	0.7361	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
24	0.5696	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
25	0.6390	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
26	0.1222	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
27	0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
28	0.3460	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
29	0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
...	...	...	...	...	...
769896	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769897	0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769898	0.1980	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769899	0.0000	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769900	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769901	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
769902	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769903	0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769904	0.3230	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769905	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769906	0.3660	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769907	0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769908	0.1780	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769909	0.0000	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769910	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769911	0.2330	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769912	0.2000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769913	0.0000	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769914	1.0920	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769915	0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769916	0.1090	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769917	0.5820	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769918	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769919	0.1510	NaN	30AUG2013:00:00:00.000	2013.0	Tinian

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
769920	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769921	0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769922	0.2430	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769923	0.1710	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769924	0.2060	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769925	0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian

769562 rows × 22 columns

Next, we use the built-in python functions min() and max() to evaluate the date range as follows:

In [20]:
```
pols.fiscal_year.min()
```

Out[20]:
```
1986.0
```

In [21]:
```
pols.fiscal_year.max()
```

Out[21]:
```
2015.0
```

We could also sort the dataframe by year, then find the first and the last entry as follows:

In [22]:
```
result = pols.sort_values('fiscal_year')
print(result.fiscal_year.head(1))
print(result.fiscal_year.tail(1))
```

```
625885    1986.0
Name: fiscal_year, dtype: float64
0    2015.0
Name: fiscal_year, dtype: float64
```

Conditional selection

So far we've been indexing various strides of data, using structural properties of the `DataFrame` itself. To do *interesting* things with the data, however, we often need to ask questions based on conditions.

For example, suppose we're interested in finding out ways that pollutants change over time. We don't know how many locations have a range of dates, but we can start by asking where the first tests were done like this:

In [23]:
```
pols.fiscal_year == 1986
```

```
Out[23]: 0         False
         1         False
         2         False
         3         False
         4         False
         5         False
         6         False
         7         False
         8         False
         9         False
         10        False
         11        False
         12        False
         13        False
         14        False
         15        False
         16        False
         17        False
         18        False
         19        False
         20        False
         21        False
         22        False
         23        False
         24        False
         25        False
         26        False
         27        False
         28        False
         29        False
                   ...
         769896    False
         769897    False
         769898    False
         769899    False
         769900    False
         769901    False
         769902    False
         769903    False
         769904    False
         769905    False
         769906    False
         769907    False
         769908    False
         769909    False
         769910    False
         769911    False
         769912    False
         769913    False
         769914    False
         769915    False
         769916    False
         769917    False
         769918    False
         769919    False
         769920    False
         769921    False
```

```
769922    False
769923    False
769924    False
769925    False
Name: fiscal_year, Length: 769562, dtype: bool
```

This operation produced a `Series` of `True`/`False` booleans based on the `fiscal_year` of each record. This result can then be used inside of `loc` to select the relevant data:

In [24]:
```
pols.loc[pols.fiscal_year == 1986]
```

Out[24]:

	result	coastal_ecological_area	collection_date	fiscal_year	genera
438378	56.633333	NaN	29JAN1986:00:00:00.000	1986.0	Corpus
438384	0.510000	NaN	06MAR1986:00:00:00.000	1986.0	Matago
438392	3.840000	NaN	24JAN1986:00:00:00.000	1986.0	Lower Madre
438396	37.866666	NaN	28JAN1986:00:00:00.000	1986.0	Mesqu
438398	8.466667	NaN	01FEB1986:00:00:00.000	1986.0	Espiritu
438403	65.953334	NaN	31JAN1986:00:00:00.000	1986.0	Matago
438407	0.000000	NaN	31JAN1986:00:00:00.000	1986.0	Matago
438413	29.546667	NaN	25JAN1986:00:00:00.000	1986.0	Copano
438425	0.000000	NaN	28JAN1986:00:00:00.000	1986.0	Mesqu
438428	0.000000	NaN	01FEB1986:00:00:00.000	1986.0	Espiritu
438436	6.000000	NaN	02FEB1986:00:00:00.000	1986.0	Corpus
438437	44.283333	NaN	24JAN1986:00:00:00.000	1986.0	Lower Madre
438444	25.226667	NaN	31JAN1986:00:00:00.000	1986.0	Matago
438450	18.323333	NaN	31JAN1986:00:00:00.000	1986.0	Matago
438451	17.853333	NaN	30JAN1986:00:00:00.000	1986.0	Matago
438452	26.500000	NaN	06MAR1986:00:00:00.000	1986.0	Matago
438461	33.803333	NaN	01FEB1986:00:00:00.000	1986.0	Espiritu
438467	24.323333	NaN	29JAN1986:00:00:00.000	1986.0	Corpus
438470	31.076667	NaN	02FEB1986:00:00:00.000	1986.0	San Ar

	result	coastal_ecological_area	collection_date	fiscal_year	genera
438483	46.000000	NaN	06MAR1986:00:00:00.000	1986.0	Matago
438484	9.996667	NaN	28JAN1986:00:00:00.000	1986.0	Mesqu
438491	16.093333	NaN	30JAN1986:00:00:00.000	1986.0	Matago
438493	52.136668	NaN	28JAN1986:00:00:00.000	1986.0	Mesqu
438500	16.680000	NaN	01FEB1986:00:00:00.000	1986.0	Espiritu
438511	83.320002	NaN	01FEB1986:00:00:00.000	1986.0	Espiritu
438515	83.000000	NaN	02FEB1986:00:00:00.000	1986.0	Corpus
438519	64.239999	NaN	24JAN1986:00:00:00.000	1986.0	Lower Madre
438526	11.980000	NaN	01FEB1986:00:00:00.000	1986.0	Espiritu
438532	53.500000	NaN	24MAR1986:00:00:00.000	1986.0	Unakw
438535	0.000000	NaN	25JAN1986:00:00:00.000	1986.0	Copan
...	...	...	...	...	...
671805	11.166667	NaN	13MAY1986:00:00:00.000	1986.0	Barber
671808	9.700000	NaN	13MAY1986:00:00:00.000	1986.0	Barber
671810	5.733333	NaN	12MAY1986:00:00:00.000	1986.0	Honolu
671830	0.000000	NaN	13MAY1986:00:00:00.000	1986.0	Barber
671833	0.266667	NaN	13MAY1986:00:00:00.000	1986.0	Barber
671834	0.000000	NaN	12MAY1986:00:00:00.000	1986.0	Honolu
671843	0.100000	NaN	13MAY1986:00:00:00.000	1986.0	Barber

	result	coastal_ecological_area	collection_date	fiscal_year	genera
671844	0.000000	NaN	12MAY1986:00:00:00.000	1986.0	Honolu
671850	0.000000	NaN	13MAY1986:00:00:00.000	1986.0	Barber
671858	0.033333	NaN	12MAY1986:00:00:00.000	1986.0	Honolu
671862	0.000000	NaN	12MAY1986:00:00:00.000	1986.0	Honolu
671869	0.400000	NaN	12MAY1986:00:00:00.000	1986.0	Honolu
671876	0.000000	NaN	13MAY1986:00:00:00.000	1986.0	Barber
671880	0.000000	NaN	12MAY1986:00:00:00.000	1986.0	Honolu
671885	0.233333	NaN	12MAY1986:00:00:00.000	1986.0	Honolu
671914	50.533333	NaN	12MAY1986:00:00:00.000	1986.0	Honolu
671940	0.000000	NaN	12MAY1986:00:00:00.000	1986.0	Honolu
671949	0.000000	NaN	12MAY1986:00:00:00.000	1986.0	Honolu
671956	36.533019	NaN	12MAY1986:00:00:00.000	1986.0	Honolu
671962	0.000000	NaN	12MAY1986:00:00:00.000	1986.0	Honolu
671972	48.692810	NaN	13MAY1986:00:00:00.000	1986.0	Barber
671983	0.846667	NaN	13MAY1986:00:00:00.000	1986.0	Barber
671986	0.300000	NaN	13MAY1986:00:00:00.000	1986.0	Barber
671987	1.300000	NaN	13MAY1986:00:00:00.000	1986.0	Barber
671990	0.786667	NaN	12MAY1986:00:00:00.000	1986.0	Honolu
672000	0.873333	NaN	12MAY1986:00:00:00.000	1986.0	Honolu

	result	coastal_ecological_area	collection_date	fiscal_year	genera
672003	32.060000	NaN	12MAY1986:00:00:00.000	1986.0	Honolu
672013	26.000000	NaN	13MAY1986:00:00:00.000	1986.0	Barber
672015	0.166667	NaN	12MAY1986:00:00:00.000	1986.0	Honolu
672018	50.666667	NaN	13MAY1986:00:00:00.000	1986.0	Barber

4290 rows × 22 columns

We can see all of the locations that were tested in 1986 like this:

In [25]:
```
pols.loc[pols.fiscal_year == 1986].general_location.unique()
```

Out[25]:
```
array(['Corpus Christi', 'Matagorda Bay', 'Lower Laguna Madre',
       'Mesquite Bay', 'Espiritu Santo', 'Copano Bay', 'San Antonio Bay',
       'Unakwit Inlet', 'Aransas Bay', 'Port Valdez', 'St. Andrew Bay',
       'Calcasieu Lake', 'Caillou Lake', 'Mobile Bay', 'Rookery Bay',
       'Galveston Bay', 'Joseph Harbor Bayou', 'Tampa Bay',
       'Terrebonne Bay', 'Mississippi Sound', 'Barataria Bay',
       'Pensacola Bay', 'Choctawhatchee Bay', 'Breton Sound',
       'Lake Borgne', 'Apalachicola Bay', 'Everglades', 'Vermilion Bay',
       'Naples Bay', 'Cedar Key', 'Charlotte Harbor', 'Atchafalaya Bay',
       'Sabine Lake', 'East Cote Blanche', "Barber's Point",
       'Honolulu Hrb.'], dtype=object)
```

Similarly, we can view the test locations ten years later:

In [26]:
```
pols.loc[pols.fiscal_year == 1996].general_location.unique()
```

Out[26]:
```
array(['North Miami', 'Rickenbacker Causeway', 'Miami Shores',
       'Miami Beach', 'North Miami Beach', 'Coral Gables Canal',
       'North Canal', 'Mowry Canal', 'Military Canal',
       'Little Card Sound', 'Card Sound', 'Key Largo', 'Mangrove Point',
       'South Bay', 'Snapper Creek Canal', 'Florida Bay',
       'Central Galveston Bay- West', 'Trinity Bay - Offshore',
       'Central Galveston Bay- East', 'Scott Bay', 'East Bay',
       'Lower San Jacinto Bay', 'Texas City', 'Clear Lake',
       'Lower Galveston Bay', 'Upper Galveston Bay - West', 'West Bay',
       'Upper Houston Ship Channel', 'Trinity Bay - Nearshore',
       'Upper San Jacinto Bay', 'Upper Galveston Bay - East', 'Tabbs Bay',
       'Galveston Island - Nearshore', 'Bolivar Peninsula - Offshore',
       'Galveston Bay - Entrance', 'Bolivar Peninsula - Nearshore',
       'Bolivar Roads', 'Galveston Island - Offshore', 'Espiritu Santo',
       'Lower Laguna Madre', 'San Antonio Bay', 'Lake Michigan',
       'Saginaw Bay', 'Lake Huron', 'Green Bay', 'Lake Erie',
       'Traverse Bay', 'Mobile Bay', 'Tampa Bay', 'Choctawhatchee Bay',
       'Calcasieu Lake', 'Terrebonne Bay', 'Breton Sound', 'Lake Borgne',
       'Mississippi River', 'Mississippi Sound', 'Rookery Bay',
       'Naples Bay', 'Everglades', 'Pensacola Bay', 'Lake Pontchartrain',
       'Honolulu Hrb.', 'Hawaii', 'Port Valdez', 'Prince William Sound',
       'Cook Inlet', 'Gulf of Alaska'], dtype=object)
```

In [27]:
```
pols
```

Out[27]:

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
0	0.1040	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
1	0.0380	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
2	0.4470	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
3	0.5950	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
4	0.4600	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
5	0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
6	0.3010	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
7	4.4500	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
8	0.2710	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
9	0.7470	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
10	0.6000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
11	0.1470	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
12	0.1140	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
13	0.4270	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
14	0.2130	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
15	0.7260	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
16	0.0780	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
17	0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
18	0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
19	0.2770	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
20	0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
21	0.4010	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
22	0.2157	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
23	0.7361	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
24	0.5696	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
25	0.6390	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
26	0.1222	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
27	0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
28	0.3460	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
29	0.0000	NaN	18AUG2015:00:00:00.000	2015.0	Chukchi Se
...	...	...	...	...	...
769896	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769897	0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769898	0.1980	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769899	0.0000	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769900	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769901	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
769902	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769903	0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769904	0.3230	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769905	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769906	0.3660	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769907	0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769908	0.1780	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769909	0.0000	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769910	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769911	0.2330	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769912	0.2000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769913	0.0000	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
769914	1.0920	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769915	0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769916	0.1090	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769917	0.5820	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769918	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769919	0.1510	NaN	30AUG2013:00:00:00.000	2013.0	Tinian

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
769920	0.0000	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769921	0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769922	0.2430	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769923	0.1710	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769924	0.2060	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769925	0.0000	NaN	27AUG2013:00:00:00.000	2013.0	Tinian

769562 rows × 22 columns

If we look, we can see that Tampa Bay is in both of those lists, so we can view how pollutants in Tampa Bay change over time. First we will create a dataframe containing just the Tampa Bay tests like this:

In [28]:
```
tb = pols.loc[pols.general_location == 'Tampa Bay']
tb
```

Out[28]:

	result	coastal_ecological_area	collection_date	fiscal_year	genera
531964	8.310000	NaN	30JAN2012:00:00:00.000	2012.0	Tampa
531969	0.000000	NaN	30JAN2012:00:00:00.000	2012.0	Tampa
531974	0.000000	NaN	24JAN1996:00:00:00.000	1996.0	Tampa
531979	19.400000	NaN	08FEB2010:00:00:00.000	2010.0	Tampa
531983	21.800000	NaN	08FEB2006:00:00:00.000	2006.0	Tampa
531984	0.000000	NaN	08FEB2006:00:00:00.000	2006.0	Tampa
531992	4.023333	NaN	30JAN1991:00:00:00.000	1991.0	Tampa
531993	12.800000	NaN	14FEB2000:00:00:00.000	2000.0	Tampa
531996	11.083333	NaN	30JAN1991:00:00:00.000	1991.0	Tampa
531997	13.200000	NaN	16FEB2008:00:00:00.000	2008.0	Tampa
531998	10.473333	NaN	31JAN1991:00:00:00.000	1991.0	Tampa
531999	81.613333	NaN	31JAN1991:00:00:00.000	1991.0	Tampa
532000	2.400000	NaN	16FEB2008:00:00:00.000	2008.0	Tampa
532008	30.640000	NaN	29JAN1992:00:00:00.000	1992.0	Tampa
532017	18.100000	NaN	06FEB2010:00:00:00.000	2010.0	Tampa
532020	0.000000	NaN	02FEB2004:00:00:00.000	2004.0	Tampa
532036	0.000000	NaN	08FEB2010:00:00:00.000	2010.0	Tampa
532038	0.000000	NaN	30JAN2012:00:00:00.000	2012.0	Tampa
532046	0.000000	NaN	30JAN2002:00:00:00.000	2002.0	Tampa

	result	coastal_ecological_area	collection_date	fiscal_year	genera
532072	0.000000	NaN	15JAN1993:00:00:00.000	1993.0	Tampa
532078	0.000000	NaN	16FEB2008:00:00:00.000	2008.0	Tampa
532086	0.000000	NaN	28JAN1992:00:00:00.000	1992.0	Tampa
532091	0.000000	NaN	06FEB2010:00:00:00.000	2010.0	Tampa
532099	0.000000	NaN	24JAN1996:00:00:00.000	1996.0	Tampa
532101	0.000000	NaN	02FEB1998:00:00:00.000	1998.0	Tampa
532107	4.700000	NaN	07FEB2010:00:00:00.000	2010.0	Tampa
532117	0.000000	NaN	06FEB2010:00:00:00.000	2010.0	Tampa
532119	0.000000	NaN	16FEB2008:00:00:00.000	2008.0	Tampa
532121	0.000000	NaN	25JAN1996:00:00:00.000	1996.0	Tampa
532123	9.900000	NaN	31JAN2002:00:00:00.000	2002.0	Tampa
...	...	...	...	...	...
743145	15.200000	NaN	04NOV2010:00:00:00.000	2011.0	Tampa
743148	14.300000	NaN	04NOV2010:00:00:00.000	2011.0	Tampa
743162	9.800000	NaN	04NOV2010:00:00:00.000	2011.0	Tampa
743163	34.000000	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
743169	11.800000	NaN	05NOV2010:00:00:00.000	2011.0	Tampa
743171	6.420000	NaN	05NOV2010:00:00:00.000	2011.0	Tampa
743172	9.700000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa

	result	coastal_ecological_area	collection_date	fiscal_year	genera
743176	8.250000	NaN	05NOV2010:00:00:00.000	2011.0	Tampa
743177	28.500000	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
743178	8.620000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
743198	3.020000	NaN	05NOV2010:00:00:00.000	2011.0	Tampa
743203	3.010000	NaN	05NOV2010:00:00:00.000	2011.0	Tampa
743206	2.800000	NaN	05NOV2010:00:00:00.000	2011.0	Tampa
743207	2.800000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
743209	2.650000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
743210	3.780000	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
743212	3.600000	NaN	04NOV2010:00:00:00.000	2011.0	Tampa
743236	93.800000	NaN	04NOV2010:00:00:00.000	2011.0	Tampa
743259	86.800000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
743271	1.910000	NaN	05NOV2010:00:00:00.000	2011.0	Tampa
743277	1.090000	NaN	04NOV2010:00:00:00.000	2011.0	Tampa
743278	6.900000	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
743282	7.950000	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
743290	3.280000	NaN	05NOV2010:00:00:00.000	2011.0	Tampa
743298	3.350000	NaN	04NOV2010:00:00:00.000	2011.0	Tampa
743314	1.190000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa

	result	coastal_ecological_area	collection_date	fiscal_year	genera
743316	1.520000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
743320	1.440000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
743321	5.680000	NaN	04NOV2010:00:00:00.000	2011.0	Tampa
743334	1.880000	NaN	05NOV2010:00:00:00.000	2011.0	Tampa

19549 rows × 22 columns

Next we can sort by `result` which corresponds to pollutant concentration and then group by `parameter` like this:

In [29]:
```
tb.sort_values('result', ascending = False)
```

Out[29]:

	result	coastal_ecological_area	collection_date	fiscal_year	ge
625880	37600.000000	NaN	10FEB2006:00:00:00.000	2006.0	Tar
625485	19600.000000	NaN	10FEB2006:00:00:00.000	2006.0	Tar
625879	15666.666667	NaN	01MAR1987:00:00:00.000	1987.0	Tar
625888	15354.000000	NaN	25JAN1996:00:00:00.000	1996.0	Tar
625883	13600.000000	NaN	09FEB2006:00:00:00.000	2006.0	Tar
625866	11906.000000	NaN	NaN	1989.0	Tar
651307	10300.000000	NaN	08FEB2006:00:00:00.000	2006.0	Tar
654149	8820.000000	NaN	25JAN1998:00:00:00.000	1998.0	Tar
625547	8410.000000	NaN	25JAN1996:00:00:00.000	1996.0	Tar
652763	8309.000000	NaN	22FEB1988:00:00:00.000	1988.0	Tar
652039	8023.410000	NaN	15JAN1993:00:00:00.000	1993.0	Tar
655797	7774.000000	NaN	25JAN1996:00:00:00.000	1996.0	Tar

	result	coastal_ecological_area	collection_date	fiscal_year	ge
652155	7624.550000	NaN	29JAN1992:00:00:00.000	1992.0	Tar
654243	7590.000000	NaN	20JAN1994:00:00:00.000	1994.0	Tar
625862	7500.000000	NaN	28FEB1987:00:00:00.000	1987.0	Tar
625536	7466.666667	NaN	01MAR1987:00:00:00.000	1987.0	Tar
652929	7420.900000	NaN	30JAN1990:00:00:00.000	1990.0	Tar
625874	7410.000000	NaN	09FEB2006:00:00:00.000	2006.0	Tar
654234	7380.000000	NaN	29JAN1991:00:00:00.000	1991.0	Tar
625899	7286.000000	NaN	24JAN1996:00:00:00.000	1996.0	Tar
651909	7180.000000	NaN	30JAN1991:00:00:00.000	1991.0	Tar
625522	7170.000000	NaN	09FEB2006:00:00:00.000	2006.0	Tar
625567	6883.333333	NaN	NaN	1989.0	Tar
652795	6673.576667	NaN	NaN	1989.0	Tar
652049	6450.000000	NaN	01FEB2004:00:00:00.000	2004.0	Tar

	result	coastal_ecological_area	collection_date	fiscal_year	ge
630385	6300.000000	NaN	24JAN1996:00:00:00.000	1996.0	Tar
625877	6000.000000	NaN	24FEB1986:00:00:00.000	1986.0	Tar
654152	5980.000000	NaN	11FEB2000:00:00:00.000	2000.0	Tar
625878	5800.000000	NaN	24JAN1996:00:00:00.000	1996.0	Tar
654244	5730.000000	NaN	25JAN1998:00:00:00.000	1998.0	Tar
...	...	...	...	...	...
623456	0.000000	NaN	10FEB2006:00:00:00.000	2006.0	Tar
624394	0.000000	NaN	25JAN1996:00:00:00.000	1996.0	Tar
556781	0.000000	NaN	07FEB2010:00:00:00.000	2010.0	Tar
624401	0.000000	NaN	25JAN1996:00:00:00.000	1996.0	Tar
556509	0.000000	NaN	10FEB2006:00:00:00.000	2006.0	Tar
556120	0.000000	NaN	06FEB2010:00:00:00.000	2010.0	Tar
556211	0.000000	NaN	30JAN2012:00:00:00.000	2012.0	Tar
556217	0.000000	NaN	08FEB2010:00:00:00.000	2010.0	Tar
556254	0.000000	NaN	26JAN1998:00:00:00.000	1998.0	Tar
556260	0.000000	NaN	16JAN1993:00:00:00.000	1993.0	Tar
556265	0.000000	NaN	30JAN2012:00:00:00.000	2012.0	Tar

	result	coastal_ecological_area	collection_date	fiscal_year	ge
556278	0.000000	NaN	20JAN1994:00:00:00.000	1994.0	Tar
556298	0.000000	NaN	06FEB2010:00:00:00.000	2010.0	Tar
556447	0.000000	NaN	06FEB2010:00:00:00.000	2010.0	Tar
556464	0.000000	NaN	30JAN2012:00:00:00.000	2012.0	Tar
556498	0.000000	NaN	08FEB2010:00:00:00.000	2010.0	Tar
556521	0.000000	NaN	17FEB2008:00:00:00.000	2008.0	Tar
556757	0.000000	NaN	31JAN2002:00:00:00.000	2002.0	Tar
556538	0.000000	NaN	09FEB2006:00:00:00.000	2006.0	Tar
556552	0.000000	NaN	08FEB2010:00:00:00.000	2010.0	Tar
556556	0.000000	NaN	17FEB2008:00:00:00.000	2008.0	Tar
556560	0.000000	NaN	30JAN2012:00:00:00.000	2012.0	Tar
556563	0.000000	NaN	07FEB2010:00:00:00.000	2010.0	Tar
556568	0.000000	NaN	NaN	1989.0	Tar
556574	0.000000	NaN	16FEB2008:00:00:00.000	2008.0	Tar
556580	0.000000	NaN	30JAN2012:00:00:00.000	2012.0	Tar
556604	0.000000	NaN	20JAN1994:00:00:00.000	1994.0	Tar
556681	0.000000	NaN	25JAN1998:00:00:00.000	1998.0	Tar
556712	0.000000	NaN	08FEB2010:00:00:00.000	2010.0	Tar
633212	0.000000	NaN	25JAN1996:00:00:00.000	1996.0	Tar

19549 rows × 22 columns

In [30]:
```
top5 = tb.groupby(['parameter', 'units']).head(5)
top5
```

Out[30]:

	result	coastal_ecological_area	collection_date	fiscal_year	genera
531964	8.310000	NaN	30JAN2012:00:00:00.000	2012.0	Tampa
531969	0.000000	NaN	30JAN2012:00:00:00.000	2012.0	Tampa
531974	0.000000	NaN	24JAN1996:00:00:00.000	1996.0	Tampa
531979	19.400000	NaN	08FEB2010:00:00:00.000	2010.0	Tampa
531983	21.800000	NaN	08FEB2006:00:00:00.000	2006.0	Tampa
531984	0.000000	NaN	08FEB2006:00:00:00.000	2006.0	Tampa
531992	4.023333	NaN	30JAN1991:00:00:00.000	1991.0	Tampa
531993	12.800000	NaN	14FEB2000:00:00:00.000	2000.0	Tampa
531996	11.083333	NaN	30JAN1991:00:00:00.000	1991.0	Tampa
531997	13.200000	NaN	16FEB2008:00:00:00.000	2008.0	Tampa
531998	10.473333	NaN	31JAN1991:00:00:00.000	1991.0	Tampa
531999	81.613333	NaN	31JAN1991:00:00:00.000	1991.0	Tampa
532000	2.400000	NaN	16FEB2008:00:00:00.000	2008.0	Tampa
532008	30.640000	NaN	29JAN1992:00:00:00.000	1992.0	Tampa
532017	18.100000	NaN	06FEB2010:00:00:00.000	2010.0	Tampa
532020	0.000000	NaN	02FEB2004:00:00:00.000	2004.0	Tampa
532036	0.000000	NaN	08FEB2010:00:00:00.000	2010.0	Tampa
532038	0.000000	NaN	30JAN2012:00:00:00.000	2012.0	Tampa
532046	0.000000	NaN	30JAN2002:00:00:00.000	2002.0	Tampa

	result	coastal_ecological_area	collection_date	fiscal_year	genera
532072	0.000000	NaN	15JAN1993:00:00:00.000	1993.0	Tampa
532078	0.000000	NaN	16FEB2008:00:00:00.000	2008.0	Tampa
532086	0.000000	NaN	28JAN1992:00:00:00.000	1992.0	Tampa
532091	0.000000	NaN	06FEB2010:00:00:00.000	2010.0	Tampa
532099	0.000000	NaN	24JAN1996:00:00:00.000	1996.0	Tampa
532101	0.000000	NaN	02FEB1998:00:00:00.000	1998.0	Tampa
532107	4.700000	NaN	07FEB2010:00:00:00.000	2010.0	Tampa
532117	0.000000	NaN	06FEB2010:00:00:00.000	2010.0	Tampa
532119	0.000000	NaN	16FEB2008:00:00:00.000	2008.0	Tampa
532121	0.000000	NaN	25JAN1996:00:00:00.000	1996.0	Tampa
532123	9.900000	NaN	31JAN2002:00:00:00.000	2002.0	Tampa
...	...	...	...	...	...
730608	0.010000	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
730619	0.010200	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
730651	0.006270	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
730652	0.006640	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
730653	0.011700	NaN	02MAY2010:00:00:00.000	2010.0	Tampa

	result	coastal_ecological_area	collection_date	fiscal_year	genera
730705	0.013300	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
730706	0.014000	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
730714	0.021900	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
730715	0.014300	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
730729	0.012800	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
730750	54.400000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
730759	32.800000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
730760	50.400000	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
730779	40.200000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
730813	57.400000	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
730910	0.350000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
730969	1.420000	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
730970	2.520000	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
731010	0.420000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
731117	0.440000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
731143	1.510000	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
731144	0.430000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa

	result	coastal_ecological_area	collection_date	fiscal_year	genera
731145	0.970000	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
731225	0.330000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
731226	0.440000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
736174	0.000000	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
736175	0.000000	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
736210	0.000000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
736211	0.000000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
736212	0.000000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa

1593 rows × 22 columns

Well, that isn't very interesting. Let's drop the zeros and try again. We create a new dataframe for tampa bay based on these criteria using & as follows:

In [31]:
```
tb3 = pols.loc[(pols.general_location == 'Tampa Bay') & (pols.result > 0)]
tb3
```

Out[31]:

	result	coastal_ecological_area	collection_date	fiscal_year	gener
531964	8.310000	NaN	30JAN2012:00:00:00.000	2012.0	Tampa
531979	19.400000	NaN	08FEB2010:00:00:00.000	2010.0	Tampa
531983	21.800000	NaN	08FEB2006:00:00:00.000	2006.0	Tampa
531992	4.023333	NaN	30JAN1991:00:00:00.000	1991.0	Tampa
531993	12.800000	NaN	14FEB2000:00:00:00.000	2000.0	Tampa
531996	11.083333	NaN	30JAN1991:00:00:00.000	1991.0	Tampa
531997	13.200000	NaN	16FEB2008:00:00:00.000	2008.0	Tampa
531998	10.473333	NaN	31JAN1991:00:00:00.000	1991.0	Tampa
531999	81.613333	NaN	31JAN1991:00:00:00.000	1991.0	Tampa
532000	2.400000	NaN	16FEB2008:00:00:00.000	2008.0	Tampa
532008	30.640000	NaN	29JAN1992:00:00:00.000	1992.0	Tampa
532017	18.100000	NaN	06FEB2010:00:00:00.000	2010.0	Tampa
532107	4.700000	NaN	07FEB2010:00:00:00.000	2010.0	Tampa
532123	9.900000	NaN	31JAN2002:00:00:00.000	2002.0	Tampa
532125	10.800000	NaN	11FEB2000:00:00:00.000	2000.0	Tampa
532126	5.580000	NaN	31JAN1990:00:00:00.000	1990.0	Tampa
532127	18.600000	NaN	16FEB2008:00:00:00.000	2008.0	Tampa
532128	8.200000	NaN	30JAN2002:00:00:00.000	2002.0	Tampa
532129	136.580000	NaN	21JAN1994:00:00:00.000	1994.0	Tampa

	result	coastal_ecological_area	collection_date	fiscal_year	gener
532156	31.300000	NaN	09FEB2006:00:00:00.000	2006.0	Tampa
532160	180.100000	NaN	25JAN1996:00:00:00.000	1996.0	Tampa
532161	55.546667	NaN	01FEB1990:00:00:00.000	1990.0	Tampa
532163	16.700000	NaN	06FEB2010:00:00:00.000	2010.0	Tampa
532166	158.703333	NaN	29JAN1991:00:00:00.000	1991.0	Tampa
532167	68.500000	NaN	13FEB2000:00:00:00.000	2000.0	Tampa
532191	11.700000	NaN	09FEB2006:00:00:00.000	2006.0	Tampa
532198	14.000000	NaN	17FEB2008:00:00:00.000	2008.0	Tampa
532201	12.800000	NaN	03FEB2004:00:00:00.000	2004.0	Tampa
532204	51.832000	NaN	30JAN2012:00:00:00.000	2012.0	Tampa
532207	77.900000	NaN	13FEB2000:00:00:00.000	2000.0	Tampa
...	...	...	...	...	...
743145	15.200000	NaN	04NOV2010:00:00:00.000	2011.0	Tampa
743148	14.300000	NaN	04NOV2010:00:00:00.000	2011.0	Tampa
743162	9.800000	NaN	04NOV2010:00:00:00.000	2011.0	Tampa
743163	34.000000	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
743169	11.800000	NaN	05NOV2010:00:00:00.000	2011.0	Tampa
743171	6.420000	NaN	05NOV2010:00:00:00.000	2011.0	Tampa
743172	9.700000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa

	result	coastal_ecological_area	collection_date	fiscal_year	gener
743176	8.250000	NaN	05NOV2010:00:00:00.000	2011.0	Tampa
743177	28.500000	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
743178	8.620000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
743198	3.020000	NaN	05NOV2010:00:00:00.000	2011.0	Tampa
743203	3.010000	NaN	05NOV2010:00:00:00.000	2011.0	Tampa
743206	2.800000	NaN	05NOV2010:00:00:00.000	2011.0	Tampa
743207	2.800000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
743209	2.650000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
743210	3.780000	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
743212	3.600000	NaN	04NOV2010:00:00:00.000	2011.0	Tampa
743236	93.800000	NaN	04NOV2010:00:00:00.000	2011.0	Tampa
743259	86.800000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
743271	1.910000	NaN	05NOV2010:00:00:00.000	2011.0	Tampa
743277	1.090000	NaN	04NOV2010:00:00:00.000	2011.0	Tampa
743278	6.900000	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
743282	7.950000	NaN	02MAY2010:00:00:00.000	2010.0	Tampa
743290	3.280000	NaN	05NOV2010:00:00:00.000	2011.0	Tampa
743298	3.350000	NaN	04NOV2010:00:00:00.000	2011.0	Tampa
743314	1.190000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa

	result	coastal_ecological_area	collection_date	fiscal_year	gener
743316	1.520000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
743320	1.440000	NaN	01MAY2010:00:00:00.000	2010.0	Tampa
743321	5.680000	NaN	04NOV2010:00:00:00.000	2011.0	Tampa
743334	1.880000	NaN	05NOV2010:00:00:00.000	2011.0	Tampa

13941 rows × 22 columns

Let's slice out the columns we are interested in like this:

In [32]:
```
tb3.iloc[0]
```

Out[32]:
```
result                                                        8.31
coastal_ecological_area                                        NaN
collection_date                                30JAN2012:00:00:00.000
fiscal_year                                                   2012
general_location                                         Tampa Bay
latitude                                                       NaN
longitude                                                      NaN
matrix                                                      Oyster
method                                                    PAH-2010
nst_sample_id                                          MW2012TBHBCV
nst_site                                                       TBHB
parameter                                  C4-Fluoranthenes_Pyrenes
parameter_name                                                 NaN
qualifier                                                      NaN
region_name                                                    NaN
scientific_name                                 Crassostrea virginica
source_file                MusselWatch_GulfCoast_Organics_Tissue.csv
specific_location                                  Hillsborough Bay
state_name                                                     NaN
stratum                                                        NaN
study_name                                             Mussel Watch
units                                                      ng/dry g
Name: 531964, dtype: object
```

In [33]:
```
tb4 = tb3.iloc[: , [0,3,4,11,21]]
tb4
```

Out[33]:

	result	fiscal_year	general_location	parameter	units
531964	8.310000	2012.0	Tampa Bay	C4-Fluoranthenes_Pyrenes	ng/dry g
531979	19.400000	2010.0	Tampa Bay	C4-Naphthalenes	ng/dry g
531983	21.800000	2006.0	Tampa Bay	C3-Naphthalenes	ng/dry g
531992	4.023333	1991.0	Tampa Bay	C4-Phenanthrenes_Anthracenes	ng/dry g
531993	12.800000	2000.0	Tampa Bay	C3-Naphthalenes	ng/dry g
531996	11.083333	1991.0	Tampa Bay	C4-Phenanthrenes_Anthracenes	ng/dry g
531997	13.200000	2008.0	Tampa Bay	C4-Phenanthrenes_Anthracenes	ng/dry g
531998	10.473333	1991.0	Tampa Bay	C3-Naphthalenes	ng/dry g
531999	81.613333	1991.0	Tampa Bay	C4-Naphthalenes	ng/dry g
532000	2.400000	2008.0	Tampa Bay	C3-Naphthobenzothiophene	ng/dry g
532008	30.640000	1992.0	Tampa Bay	C3-Naphthalenes	ng/dry g
532017	18.100000	2010.0	Tampa Bay	C3-Fluorenes	ng/dry g
532107	4.700000	2010.0	Tampa Bay	C4-Fluoranthenes_Pyrenes	ng/dry g
532123	9.900000	2002.0	Tampa Bay	C3-Phenanthrenes_Anthracenes	ng/dry g
532125	10.800000	2000.0	Tampa Bay	C3-Phenanthrenes_Anthracenes	ng/dry g
532126	5.580000	1990.0	Tampa Bay	C3-Phenanthrenes_Anthracenes	ng/dry g
532127	18.600000	2008.0	Tampa Bay	C3-Phenanthrenes_Anthracenes	ng/dry g
532128	8.200000	2002.0	Tampa Bay	C3-Phenanthrenes_Anthracenes	ng/dry g
532129	136.580000	1994.0	Tampa Bay	C3-Phenanthrenes_Anthracenes	ng/dry g

	result	fiscal_year	general_location	parameter	units
532156	31.300000	2006.0	Tampa Bay	C3-Fluorenes	ng/dry g
532160	180.100000	1996.0	Tampa Bay	C3-Fluorenes	ng/dry g
532161	55.546667	1990.0	Tampa Bay	C3-Fluorenes	ng/dry g
532163	16.700000	2010.0	Tampa Bay	C4-Phenanthrenes_Anthracenes	ng/dry g
532166	158.703333	1991.0	Tampa Bay	C4-Phenanthrenes_Anthracenes	ng/dry g
532167	68.500000	2000.0	Tampa Bay	C4-Phenanthrenes_Anthracenes	ng/dry g
532191	11.700000	2006.0	Tampa Bay	C3-Naphthalenes	ng/dry g
532198	14.000000	2008.0	Tampa Bay	C4-Naphthalenes	ng/dry g
532201	12.800000	2004.0	Tampa Bay	C4-Naphthalenes	ng/dry g
532204	51.832000	2012.0	Tampa Bay	C4-Naphthalenes	ng/dry g
532207	77.900000	2000.0	Tampa Bay	C3-Fluorenes	ng/dry g
...	...	...	...	...	...
743145	15.200000	2011.0	Tampa Bay	Phenanthrene	ng/dry g
743148	14.300000	2011.0	Tampa Bay	Phenanthrene	ng/dry g
743162	9.800000	2011.0	Tampa Bay	Phenanthrene	ng/dry g
743163	34.000000	2010.0	Tampa Bay	Phenanthrene	ng/dry g
743169	11.800000	2011.0	Tampa Bay	Phenanthrene	ng/dry g
743171	6.420000	2011.0	Tampa Bay	Phenanthrene	ng/dry g
743172	9.700000	2010.0	Tampa Bay	Phenanthrene	ng/dry g

	result	fiscal_year	general_location	parameter	units
743176	8.250000	2011.0	Tampa Bay	Phenanthrene	ng/dry g
743177	28.500000	2010.0	Tampa Bay	Phenanthrene	ng/dry g
743178	8.620000	2010.0	Tampa Bay	Phenanthrene	ng/dry g
743198	3.020000	2011.0	Tampa Bay	Perylene	ng/dry g
743203	3.010000	2011.0	Tampa Bay	Perylene	ng/dry g
743206	2.800000	2011.0	Tampa Bay	Perylene	ng/dry g
743207	2.800000	2010.0	Tampa Bay	Perylene	ng/dry g
743209	2.650000	2010.0	Tampa Bay	Perylene	ng/dry g
743210	3.780000	2010.0	Tampa Bay	Perylene	ng/dry g
743212	3.600000	2011.0	Tampa Bay	Perylene	ng/dry g
743236	93.800000	2011.0	Tampa Bay	Sample percent wet weight	Percent
743259	86.800000	2010.0	Tampa Bay	Sample percent wet weight	Percent
743271	1.910000	2011.0	Tampa Bay	Anthracene	ng/dry g
743277	1.090000	2011.0	Tampa Bay	Anthracene	ng/dry g
743278	6.900000	2010.0	Tampa Bay	Anthracene	ng/dry g
743282	7.950000	2010.0	Tampa Bay	Anthracene	ng/dry g
743290	3.280000	2011.0	Tampa Bay	Anthracene	ng/dry g
743298	3.350000	2011.0	Tampa Bay	Anthracene	ng/dry g
743314	1.190000	2010.0	Tampa Bay	Anthracene	ng/dry g
743316	1.520000	2010.0	Tampa Bay	Anthracene	ng/dry g

	result	fiscal_year	general_location	parameter	units
743320	1.440000	2010.0	Tampa Bay	Anthracene	ng/dry g
743321	5.680000	2011.0	Tampa Bay	Anthracene	ng/dry g
743334	1.880000	2011.0	Tampa Bay	Anthracene	ng/dry g

13941 rows × 5 columns

Then we sort and group as before

In [34]:
```
tb4.sort_values('result', ascending = False)
```

Out[34]:

	result	fiscal_year	general_location	parameter	units
625880	37600.000000	2006.0	Tampa Bay	Aluminum	micrograms per dry gram
625485	19600.000000	2006.0	Tampa Bay	Iron	micrograms per dry gram
625879	15666.666667	1987.0	Tampa Bay	Aluminum	micrograms per dry gram
625888	15354.000000	1996.0	Tampa Bay	Aluminum	micrograms per dry gram
625883	13600.000000	2006.0	Tampa Bay	Aluminum	micrograms per dry gram
625866	11906.000000	1989.0	Tampa Bay	Aluminum	micrograms per dry gram
651307	10300.000000	2006.0	Tampa Bay	Zinc	micrograms per dry gram
654149	8820.000000	1998.0	Tampa Bay	Zinc	micrograms per dry gram
625547	8410.000000	1996.0	Tampa Bay	Iron	micrograms per dry gram
652763	8309.000000	1988.0	Tampa Bay	Zinc	micrograms per dry gram
652039	8023.410000	1993.0	Tampa Bay	Zinc	micrograms per dry gram
655797	7774.000000	1996.0	Tampa Bay	Zinc	micrograms per dry gram
652155	7624.550000	1992.0	Tampa Bay	Zinc	micrograms per dry gram
654243	7590.000000	1994.0	Tampa Bay	Zinc	micrograms per dry gram
625862	7500.000000	1987.0	Tampa Bay	Aluminum	micrograms per dry gram
625536	7466.666667	1987.0	Tampa Bay	Iron	micrograms per dry gram
652929	7420.900000	1990.0	Tampa Bay	Zinc	micrograms per dry gram
625874	7410.000000	2006.0	Tampa Bay	Aluminum	micrograms per dry gram
654234	7380.000000	1991.0	Tampa Bay	Zinc	micrograms per dry gram

	result	fiscal_year	general_location	parameter	units
625899	7286.000000	1996.0	Tampa Bay	Aluminum	micrograms per dry gram
651909	7180.000000	1991.0	Tampa Bay	Zinc	micrograms per dry gram
625522	7170.000000	2006.0	Tampa Bay	Iron	micrograms per dry gram
625567	6883.333333	1989.0	Tampa Bay	Iron	micrograms per dry gram
652795	6673.576667	1989.0	Tampa Bay	Zinc	micrograms per dry gram
652049	6450.000000	2004.0	Tampa Bay	Zinc	micrograms per dry gram
630385	6300.000000	1996.0	Tampa Bay	Clostridium perfringens - dry	CFU/g
625877	6000.000000	1986.0	Tampa Bay	Aluminum	micrograms per dry gram
654152	5980.000000	2000.0	Tampa Bay	Zinc	micrograms per dry gram
625878	5800.000000	1996.0	Tampa Bay	Aluminum	micrograms per dry gram
654244	5730.000000	1998.0	Tampa Bay	Zinc	micrograms per dry gram
...	...	...	...	...	...
730651	0.006270	2010.0	Tampa Bay	Pristane [i-C19]	micrograms per dry gram
627277	0.005667	1987.0	Tampa Bay	Silver	micrograms per dry gram
728241	0.004430	2010.0	Tampa Bay	n-Undecane	micrograms per dry gram
730384	0.004090	2010.0	Tampa Bay	Norpristane	micrograms per dry gram
631878	0.004000	1988.0	Tampa Bay	PCB187	ng/dry g
565261	0.003333	1991.0	Tampa Bay	Hexachlorobenzene	ng/dry g
638132	0.003333	1986.0	Tampa Bay	Aldrin	ng/dry g
729748	0.003320	2010.0	Tampa Bay	n-Nonane	micrograms per dry gram

	result	fiscal_year	general_location	parameter	units
730388	0.003200	2010.0	Tampa Bay	2,6,10-Trimethyldodecane (i-C15)	micrograms per dry gram
728242	0.003130	2010.0	Tampa Bay	n-Undecane	micrograms per dry gram
730212	0.003060	2010.0	Tampa Bay	n-Nonane	micrograms per dry gram
730501	0.003010	2010.0	Tampa Bay	Norpristane	micrograms per dry gram
730554	0.003010	2010.0	Tampa Bay	2,6,10-Trimethyldodecane (i-C15)	micrograms per dry gram
730553	0.002210	2010.0	Tampa Bay	2,6,10-Trimethyldodecane (i-C15)	micrograms per dry gram
729744	0.002130	2010.0	Tampa Bay	n-Nonane	micrograms per dry gram
730424	0.002130	2010.0	Tampa Bay	Norpristane	micrograms per dry gram
730475	0.002040	2010.0	Tampa Bay	2,6,10-Trimethyldodecane (i-C15)	micrograms per dry gram
728161	0.002040	2010.0	Tampa Bay	n-Decane	micrograms per dry gram
729746	0.002010	2010.0	Tampa Bay	n-Nonane	micrograms per dry gram
728159	0.002010	2010.0	Tampa Bay	n-Decane	micrograms per dry gram
728314	0.001110	2010.0	Tampa Bay	n-Decane	micrograms per dry gram
729086	0.001070	2010.0	Tampa Bay	n-Decane	micrograms per dry gram
730465	0.001040	2010.0	Tampa Bay	2,6,10-Trimethyldodecane (i-C15)	micrograms per dry gram
729088	0.001040	2010.0	Tampa Bay	n-Decane	micrograms per dry gram
730211	0.001040	2010.0	Tampa Bay	n-Nonane	micrograms per dry gram

	result	fiscal_year	general_location	parameter	units
625456	0.001040	2006.0	Tampa Bay	Methyl Mercury	micrograms per dry gram
625465	0.000920	2006.0	Tampa Bay	Methyl Mercury	micrograms per dry gram
625451	0.000380	2006.0	Tampa Bay	Methyl Mercury	micrograms per dry gram
625482	0.000140	2006.0	Tampa Bay	Methyl Mercury	micrograms per dry gram
625481	0.000140	2006.0	Tampa Bay	Methyl Mercury	micrograms per dry gram

13941 rows × 5 columns

In [35]:
```
top5 = tb4.groupby(['parameter', 'units']).head(5)
top5
```

Out[35]:

	result	fiscal_year	general_location	parameter	unit
531964	8.310000	2012.0	Tampa Bay	C4-Fluoranthenes_Pyrenes	ng/dry g
531979	19.400000	2010.0	Tampa Bay	C4-Naphthalenes	ng/dry g
531983	21.800000	2006.0	Tampa Bay	C3-Naphthalenes	ng/dry g
531992	4.023333	1991.0	Tampa Bay	C4-Phenanthrenes_Anthracenes	ng/dry g
531993	12.800000	2000.0	Tampa Bay	C3-Naphthalenes	ng/dry g
531996	11.083333	1991.0	Tampa Bay	C4-Phenanthrenes_Anthracenes	ng/dry g
531997	13.200000	2008.0	Tampa Bay	C4-Phenanthrenes_Anthracenes	ng/dry g
531998	10.473333	1991.0	Tampa Bay	C3-Naphthalenes	ng/dry g
531999	81.613333	1991.0	Tampa Bay	C4-Naphthalenes	ng/dry g
532000	2.400000	2008.0	Tampa Bay	C3-Naphthobenzothiophene	ng/dry g
532008	30.640000	1992.0	Tampa Bay	C3-Naphthalenes	ng/dry g
532017	18.100000	2010.0	Tampa Bay	C3-Fluorenes	ng/dry g
532107	4.700000	2010.0	Tampa Bay	C4-Fluoranthenes_Pyrenes	ng/dry g
532123	9.900000	2002.0	Tampa Bay	C3-Phenanthrenes_Anthracenes	ng/dry g
532125	10.800000	2000.0	Tampa Bay	C3-Phenanthrenes_Anthracenes	ng/dry g
532126	5.580000	1990.0	Tampa Bay	C3-Phenanthrenes_Anthracenes	ng/dry g
532127	18.600000	2008.0	Tampa Bay	C3-Phenanthrenes_Anthracenes	ng/dry g
532128	8.200000	2002.0	Tampa Bay	C3-Phenanthrenes_Anthracenes	ng/dry g
532156	31.300000	2006.0	Tampa Bay	C3-Fluorenes	ng/dry g
532160	180.100000	1996.0	Tampa Bay	C3-Fluorenes	ng/dry g
532161	55.546667	1990.0	Tampa Bay	C3-Fluorenes	ng/dry g
532163	16.700000	2010.0	Tampa Bay	C4-Phenanthrenes_Anthracenes	ng/dry g
532166	158.703333	1991.0	Tampa Bay	C4-Phenanthrenes_Anthracenes	ng/dry g
532191	11.700000	2006.0	Tampa Bay	C3-Naphthalenes	ng/dry g
532198	14.000000	2008.0	Tampa Bay	C4-Naphthalenes	ng/dry g

	result	fiscal_year	general_location	parameter	units
532201	12.800000	2004.0	Tampa Bay	C4-Naphthalenes	ng/dry g
532204	51.832000	2012.0	Tampa Bay	C4-Naphthalenes	ng/dry g
532207	77.900000	2000.0	Tampa Bay	C3-Fluorenes	ng/dry g
532250	0.950000	1994.0	Tampa Bay	Chlorpyrifos	ng/dry g or L
532272	6.026000	2012.0	Tampa Bay	C3-Fluoranthenes_Pyrenes	ng/dry g
...	...	...	...	...	...
730651	0.006270	2010.0	Tampa Bay	Pristane [i-C19]	micrograms per dry gram
730652	0.006640	2010.0	Tampa Bay	Pristane [i-C19]	micrograms per dry gram
730653	0.011700	2010.0	Tampa Bay	Pristane [i-C19]	micrograms per dry gram
730705	0.013300	2010.0	Tampa Bay	Phytane [i-C20]	micrograms per dry gram
730706	0.014000	2010.0	Tampa Bay	Phytane [i-C20]	micrograms per dry gram
730714	0.021900	2010.0	Tampa Bay	Phytane [i-C20]	micrograms per dry gram
730715	0.014300	2010.0	Tampa Bay	Phytane [i-C20]	micrograms per dry gram
730729	0.012800	2010.0	Tampa Bay	Phytane [i-C20]	micrograms per dry gram
730750	54.400000	2010.0	Tampa Bay	TEH	ng/dry g
730759	32.800000	2010.0	Tampa Bay	TEH	ng/dry g
730760	50.400000	2010.0	Tampa Bay	TEH	ng/dry g
730779	40.200000	2010.0	Tampa Bay	TEH	ng/dry g
730813	57.400000	2010.0	Tampa Bay	TEH	ng/dry g
730910	0.350000	2010.0	Tampa Bay	2/4-Methylphenanthrene	ng/dry g
730969	1.420000	2010.0	Tampa Bay	2/4-Methylphenanthrene	ng/dry g

	result	fiscal_year	general_location	parameter	unit
730970	2.520000	2010.0	Tampa Bay	2/4-Methylphenanthrene	ng/dry g
731010	0.420000	2010.0	Tampa Bay	9-Methylphenanthrene	ng/dry g
731117	0.440000	2010.0	Tampa Bay	9-Methylphenanthrene	ng/dry g
731143	1.510000	2010.0	Tampa Bay	9-Methylphenanthrene	ng/dry g
731144	0.430000	2010.0	Tampa Bay	9-Methylphenanthrene	ng/dry g
731145	0.970000	2010.0	Tampa Bay	9-Methylphenanthrene	ng/dry g
731225	0.330000	2010.0	Tampa Bay	2/4-Methylphenanthrene	ng/dry g
731226	0.440000	2010.0	Tampa Bay	2/4-Methylphenanthrene	ng/dry g
731803	1.561000	2010.0	Tampa Bay	Decalin	ng/dry g
731867	1.790000	2010.0	Tampa Bay	C4-Naphthobenzothiophenes	ng/dry g
739723	9.140000	2011.0	Tampa Bay	C1-Benzothiophene	ng/dry g
740552	7.530000	2011.0	Tampa Bay	C1-Benzothiophene	ng/dry g
740736	6.890000	2011.0	Tampa Bay	C1-Benzothiophene	ng/dry g
742960	0.657000	2011.0	Tampa Bay	Retene	ng/dry g
743056	0.162000	2010.0	Tampa Bay	Retene	ng/dry g

1146 rows × 5 columns

This is still not very clear. It's time to introduce a very fun pandas function- `pivot_table`- which is a fun way to get an overview of your data really quickly. Let's see which pollutants changed the most through the years of the tampa bay study (`the default function in pivot_table is the mean but we could also aggregate the results by other functions`):

In [36]:
```
tb4.pivot_table(values = 'result', columns = 'fiscal_year', index = 'paramete
r')
```

Out[36]:

fiscal_year	1986.0	1987.0	1988.0	1989.0	1990
parameter					
1,2,3,4-Tetrachlorobenzene	NaN	NaN	NaN	NaN	NaN
1,2,4,5-Tetrachlorobenzene	NaN	NaN	NaN	NaN	NaN
1,6,7-Trimethylnaphthalene	NaN	NaN	NaN	10.837500	9.416111
1-Methyldibenzothiophene	NaN	NaN	NaN	NaN	NaN
1-Methylfluorene	NaN	NaN	NaN	NaN	NaN
1-Methylnaphthalene	NaN	NaN	NaN	16.230417	16.640000
1-Methylphenanthrene	NaN	NaN	NaN	33.412917	8.705000
18a-Oleanane	NaN	NaN	NaN	NaN	NaN
2,3,7,8-Tetrachlorodibenzofuran	NaN	NaN	NaN	NaN	NaN
2,4'-DDD	1.970476	1.308889	1.904167	2.767833	2.221667
2,4'-DDE	0.619524	1.064444	0.314867	2.329792	1.615556
2,4'-DDT	0.800000	0.480000	1.006389	1.641375	1.408333
2,6,10-Trimethyldodecane (i-C15)	NaN	NaN	NaN	NaN	NaN
2,6,10-Trimethyltridecane (i-C16)	NaN	NaN	NaN	NaN	NaN
2,6-Dimethylnaphthalene	NaN	NaN	NaN	14.495000	14.356667
2-Methylanthracene	NaN	NaN	NaN	NaN	NaN
2-Methylfluoranthene	NaN	NaN	NaN	NaN	NaN
2-Methylnaphthalene	NaN	NaN	NaN	37.876667	17.763889
2-Methylphenanthrene	NaN	NaN	NaN	NaN	NaN
2/3-Methyldibenzothiophene	NaN	NaN	NaN	NaN	NaN
2/4-Methylphenanthrene	NaN	NaN	NaN	NaN	NaN
3,6-Dimethylphenanthrene	NaN	NaN	NaN	NaN	NaN

fiscal_year	1986.0	1987.0	1988.0	1989.0	1990
parameter					
3-Methylphenanthrene	NaN	NaN	NaN	NaN	NaN
4,4'-DDD	4.368333	19.816667	19.189722	10.539167	9.703889
4,4'-DDE	12.489583	13.092083	15.456400	32.572375	21.535000
4,4'-DDT	1.166667	1.429048	2.259667	2.127000	2.656667
4-Methyldibenzothiophene	NaN	NaN	NaN	NaN	NaN
4/9-Methylphenanthrene	NaN	NaN	NaN	NaN	NaN
9-Methylphenanthrene	NaN	NaN	NaN	NaN	NaN
Acenaphthene	NaN	NaN	NaN	5.710833	6.293333
...	...	...	...	...	...
Tributyltin	NaN	NaN	NaN	287.000000	130.50000
Trichlorobiphenyls	3.209524	7.914583	NaN	NaN	NaN
Zinc	1023.083333	703.458333	2576.356389	2161.379417	3208.61722
n-Decane	NaN	NaN	NaN	NaN	NaN
n-Docosane	NaN	NaN	NaN	NaN	NaN
n-Dodecane	NaN	NaN	NaN	NaN	NaN
n-Dotriacontane	NaN	NaN	NaN	NaN	NaN
n-Eicosane	NaN	NaN	NaN	NaN	NaN
n-Heneicosane	NaN	NaN	NaN	NaN	NaN
n-Hentriacontane	NaN	NaN	NaN	NaN	NaN
n-Heptacosane	NaN	NaN	NaN	NaN	NaN
n-Heptadecane	NaN	NaN	NaN	NaN	NaN
n-Hexacosane	NaN	NaN	NaN	NaN	NaN
n-Hexadecane	NaN	NaN	NaN	NaN	NaN
n-Nonacosane	NaN	NaN	NaN	NaN	NaN
n-Nonadecane	NaN	NaN	NaN	NaN	NaN
n-Nonane	NaN	NaN	NaN	NaN	NaN
n-Octacosane	NaN	NaN	NaN	NaN	NaN
n-Octadecane	NaN	NaN	NaN	NaN	NaN
n-Pentacosane	NaN	NaN	NaN	NaN	NaN
n-Pentadecane	NaN	NaN	NaN	NaN	NaN
n-Pentatriacontane	NaN	NaN	NaN	NaN	NaN

fiscal_year	1986.0	1987.0	1988.0	1989.0	1990
parameter					
n-Tetracosane	NaN	NaN	NaN	NaN	NaN
n-Tetradecane	NaN	NaN	NaN	NaN	NaN
n-Tetratriacontane	NaN	NaN	NaN	NaN	NaN
n-Triacontane	NaN	NaN	NaN	NaN	NaN
n-Tricosane	NaN	NaN	NaN	NaN	NaN
n-Tridecane	NaN	NaN	NaN	NaN	NaN
n-Tritriacontane	NaN	NaN	NaN	NaN	NaN
n-Undecane	NaN	NaN	NaN	NaN	NaN

243 rows × 19 columns

Well, that was sort of fun... but I want to be able to work with the data not just look at it. Let's turn the pivot table into a dataframe:

In [37]:
```
df = pd.DataFrame(tb4.pivot_table(values = 'result', columns = 'fiscal_year',
index = 'parameter' ))
df
```

Out[37]:

fiscal_year	1986.0	1987.0	1988.0	1989.0	1990
parameter					
1,2,3,4-Tetrachlorobenzene	NaN	NaN	NaN	NaN	NaN
1,2,4,5-Tetrachlorobenzene	NaN	NaN	NaN	NaN	NaN
1,6,7-Trimethylnaphthalene	NaN	NaN	NaN	10.837500	9.416111
1-Methyldibenzothiophene	NaN	NaN	NaN	NaN	NaN
1-Methylfluorene	NaN	NaN	NaN	NaN	NaN
1-Methylnaphthalene	NaN	NaN	NaN	16.230417	16.640000
1-Methylphenanthrene	NaN	NaN	NaN	33.412917	8.705000
18a-Oleanane	NaN	NaN	NaN	NaN	NaN
2,3,7,8-Tetrachlorodibenzofuran	NaN	NaN	NaN	NaN	NaN
2,4'-DDD	1.970476	1.308889	1.904167	2.767833	2.221667
2,4'-DDE	0.619524	1.064444	0.314867	2.329792	1.615556
2,4'-DDT	0.800000	0.480000	1.006389	1.641375	1.408333
2,6,10-Trimethyldodecane (i-C15)	NaN	NaN	NaN	NaN	NaN
2,6,10-Trimethyltridecane (i-C16)	NaN	NaN	NaN	NaN	NaN
2,6-Dimethylnaphthalene	NaN	NaN	NaN	14.495000	14.356667
2-Methylanthracene	NaN	NaN	NaN	NaN	NaN
2-Methylfluoranthene	NaN	NaN	NaN	NaN	NaN
2-Methylnaphthalene	NaN	NaN	NaN	37.876667	17.763889
2-Methylphenanthrene	NaN	NaN	NaN	NaN	NaN
2/3-Methyldibenzothiophene	NaN	NaN	NaN	NaN	NaN
2/4-Methylphenanthrene	NaN	NaN	NaN	NaN	NaN
3,6-Dimethylphenanthrene	NaN	NaN	NaN	NaN	NaN

fiscal_year	1986.0	1987.0	1988.0	1989.0	1990
parameter					
3-Methylphenanthrene	NaN	NaN	NaN	NaN	NaN
4,4'-DDD	4.368333	19.816667	19.189722	10.539167	9.703889
4,4'-DDE	12.489583	13.092083	15.456400	32.572375	21.535000
4,4'-DDT	1.166667	1.429048	2.259667	2.127000	2.656667
4-Methyldibenzothiophene	NaN	NaN	NaN	NaN	NaN
4/9-Methylphenanthrene	NaN	NaN	NaN	NaN	NaN
9-Methylphenanthrene	NaN	NaN	NaN	NaN	NaN
Acenaphthene	NaN	NaN	NaN	5.710833	6.293333
...	...	...	...	...	...
Tributyltin	NaN	NaN	NaN	287.000000	130.50000
Trichlorobiphenyls	3.209524	7.914583	NaN	NaN	NaN
Zinc	1023.083333	703.458333	2576.356389	2161.379417	3208.61722
n-Decane	NaN	NaN	NaN	NaN	NaN
n-Docosane	NaN	NaN	NaN	NaN	NaN
n-Dodecane	NaN	NaN	NaN	NaN	NaN
n-Dotriacontane	NaN	NaN	NaN	NaN	NaN
n-Eicosane	NaN	NaN	NaN	NaN	NaN
n-Heneicosane	NaN	NaN	NaN	NaN	NaN
n-Hentriacontane	NaN	NaN	NaN	NaN	NaN
n-Heptacosane	NaN	NaN	NaN	NaN	NaN
n-Heptadecane	NaN	NaN	NaN	NaN	NaN
n-Hexacosane	NaN	NaN	NaN	NaN	NaN
n-Hexadecane	NaN	NaN	NaN	NaN	NaN
n-Nonacosane	NaN	NaN	NaN	NaN	NaN
n-Nonadecane	NaN	NaN	NaN	NaN	NaN
n-Nonane	NaN	NaN	NaN	NaN	NaN
n-Octacosane	NaN	NaN	NaN	NaN	NaN
n-Octadecane	NaN	NaN	NaN	NaN	NaN
n-Pentacosane	NaN	NaN	NaN	NaN	NaN
n-Pentadecane	NaN	NaN	NaN	NaN	NaN
n-Pentatriacontane	NaN	NaN	NaN	NaN	NaN

fiscal_year	1986.0	1987.0	1988.0	1989.0	1990
parameter					
n-Tetracosane	NaN	NaN	NaN	NaN	NaN
n-Tetradecane	NaN	NaN	NaN	NaN	NaN
n-Tetratriacontane	NaN	NaN	NaN	NaN	NaN
n-Triacontane	NaN	NaN	NaN	NaN	NaN
n-Tricosane	NaN	NaN	NaN	NaN	NaN
n-Tridecane	NaN	NaN	NaN	NaN	NaN
n-Tritriacontane	NaN	NaN	NaN	NaN	NaN
n-Undecane	NaN	NaN	NaN	NaN	NaN

243 rows × 19 columns

Let's find out which pollutants changed the most in the Tampa Bay data

In [38]:
```
## we first append the dataframe
## with the minimim and maximum across all the columns
df['small'] = df.min(axis = 1)
df['large'] = df.max(axis = 1)
## axis = 1 means across columns
## the default is axis = 0
df.info()
## df.info tells you about the DataFrame
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 243 entries, 1,2,3,4-Tetrachlorobenzene to n-Undecane
Data columns (total 21 columns):
1986.0    44 non-null float64
1987.0    43 non-null float64
1988.0    53 non-null float64
1989.0    81 non-null float64
1990.0    97 non-null float64
1991.0    97 non-null float64
1992.0    96 non-null float64
1993.0    91 non-null float64
1994.0    102 non-null float64
1996.0    130 non-null float64
1998.0    111 non-null float64
2000.0    107 non-null float64
2002.0    104 non-null float64
2004.0    123 non-null float64
2006.0    164 non-null float64
2008.0    122 non-null float64
2010.0    174 non-null float64
2011.0    68 non-null float64
2012.0    106 non-null float64
small     243 non-null float64
large     243 non-null float64
dtypes: float64(21)
memory usage: 41.8+ KB
```

now we can take the difference between the smallest and largest values

In [39]:
```
df['change'] = df.large - df.small
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 243 entries, 1,2,3,4-Tetrachlorobenzene to n-Undecane
Data columns (total 22 columns):
1986.0    44 non-null float64
1987.0    43 non-null float64
1988.0    53 non-null float64
1989.0    81 non-null float64
1990.0    97 non-null float64
1991.0    97 non-null float64
1992.0    96 non-null float64
1993.0    91 non-null float64
1994.0    102 non-null float64
1996.0    130 non-null float64
1998.0    111 non-null float64
2000.0    107 non-null float64
2002.0    104 non-null float64
2004.0    123 non-null float64
2006.0    164 non-null float64
2008.0    122 non-null float64
2010.0    174 non-null float64
2011.0    68 non-null float64
2012.0    106 non-null float64
small     243 non-null float64
large     243 non-null float64
change    243 non-null float64
dtypes: float64(22)
memory usage: 43.7+ KB
```

In [40]:
```
df['change'] = df['change'].apply(pd.to_numeric, errors = 'coerce')
df
```

Out[40]:

fiscal_year	1986.0	1987.0	1988.0	1989.0	1990
parameter					
1,2,3,4-Tetrachlorobenzene	NaN	NaN	NaN	NaN	NaN
1,2,4,5-Tetrachlorobenzene	NaN	NaN	NaN	NaN	NaN
1,6,7-Trimethylnaphthalene	NaN	NaN	NaN	10.837500	9.416111
1-Methyldibenzothiophene	NaN	NaN	NaN	NaN	NaN
1-Methylfluorene	NaN	NaN	NaN	NaN	NaN
1-Methylnaphthalene	NaN	NaN	NaN	16.230417	16.640000
1-Methylphenanthrene	NaN	NaN	NaN	33.412917	8.705000
18a-Oleanane	NaN	NaN	NaN	NaN	NaN
2,3,7,8-Tetrachlorodibenzofuran	NaN	NaN	NaN	NaN	NaN
2,4'-DDD	1.970476	1.308889	1.904167	2.767833	2.221667
2,4'-DDE	0.619524	1.064444	0.314867	2.329792	1.615556
2,4'-DDT	0.800000	0.480000	1.006389	1.641375	1.408333
2,6,10-Trimethyldodecane (i-C15)	NaN	NaN	NaN	NaN	NaN
2,6,10-Trimethyltridecane (i-C16)	NaN	NaN	NaN	NaN	NaN
2,6-Dimethylnaphthalene	NaN	NaN	NaN	14.495000	14.356667
2-Methylanthracene	NaN	NaN	NaN	NaN	NaN
2-Methylfluoranthene	NaN	NaN	NaN	NaN	NaN
2-Methylnaphthalene	NaN	NaN	NaN	37.876667	17.763889
2-Methylphenanthrene	NaN	NaN	NaN	NaN	NaN
2/3-Methyldibenzothiophene	NaN	NaN	NaN	NaN	NaN
2/4-Methylphenanthrene	NaN	NaN	NaN	NaN	NaN
3,6-Dimethylphenanthrene	NaN	NaN	NaN	NaN	NaN

fiscal_year	1986.0	1987.0	1988.0	1989.0	1990
parameter					
3-Methylphenanthrene	NaN	NaN	NaN	NaN	NaN
4,4'-DDD	4.368333	19.816667	19.189722	10.539167	9.703889
4,4'-DDE	12.489583	13.092083	15.456400	32.572375	21.535000
4,4'-DDT	1.166667	1.429048	2.259667	2.127000	2.656667
4-Methyldibenzothiophene	NaN	NaN	NaN	NaN	NaN
4/9-Methylphenanthrene	NaN	NaN	NaN	NaN	NaN
9-Methylphenanthrene	NaN	NaN	NaN	NaN	NaN
Acenaphthene	NaN	NaN	NaN	5.710833	6.293333
...	...	...	...	...	...
Tributyltin	NaN	NaN	NaN	287.000000	130.50000
Trichlorobiphenyls	3.209524	7.914583	NaN	NaN	NaN
Zinc	1023.083333	703.458333	2576.356389	2161.379417	3208.61722
n-Decane	NaN	NaN	NaN	NaN	NaN
n-Docosane	NaN	NaN	NaN	NaN	NaN
n-Dodecane	NaN	NaN	NaN	NaN	NaN
n-Dotriacontane	NaN	NaN	NaN	NaN	NaN
n-Eicosane	NaN	NaN	NaN	NaN	NaN
n-Heneicosane	NaN	NaN	NaN	NaN	NaN
n-Hentriacontane	NaN	NaN	NaN	NaN	NaN
n-Heptacosane	NaN	NaN	NaN	NaN	NaN
n-Heptadecane	NaN	NaN	NaN	NaN	NaN
n-Hexacosane	NaN	NaN	NaN	NaN	NaN
n-Hexadecane	NaN	NaN	NaN	NaN	NaN
n-Nonacosane	NaN	NaN	NaN	NaN	NaN
n-Nonadecane	NaN	NaN	NaN	NaN	NaN
n-Nonane	NaN	NaN	NaN	NaN	NaN
n-Octacosane	NaN	NaN	NaN	NaN	NaN
n-Octadecane	NaN	NaN	NaN	NaN	NaN
n-Pentacosane	NaN	NaN	NaN	NaN	NaN
n-Pentadecane	NaN	NaN	NaN	NaN	NaN
n-Pentatriacontane	NaN	NaN	NaN	NaN	NaN

fiscal_year	1986.0	1987.0	1988.0	1989.0	1990
parameter					
n-Tetracosane	NaN	NaN	NaN	NaN	NaN
n-Tetradecane	NaN	NaN	NaN	NaN	NaN
n-Tetratriacontane	NaN	NaN	NaN	NaN	NaN
n-Triacontane	NaN	NaN	NaN	NaN	NaN
n-Tricosane	NaN	NaN	NaN	NaN	NaN
n-Tridecane	NaN	NaN	NaN	NaN	NaN
n-Tritriacontane	NaN	NaN	NaN	NaN	NaN
n-Undecane	NaN	NaN	NaN	NaN	NaN

243 rows × 22 columns

In [41]:
```
df2 = df.sort_values('change', ascending = False)
df2
```

Out[41]:

fiscal_year	1986.0	1987.0	1988.0	1989.0	
parameter					
Aluminum	1930.416667	3649.916667	3390.666667	7978.333333	NaN
Zinc	1023.083333	703.458333	2576.356389	2161.379417	3208.
Iron	820.000000	1806.500000	630.111111	1310.978333	303.3
Clostridium perfringens - dry	NaN	NaN	NaN	NaN	NaN
Clostridium perfringens - wet	NaN	NaN	NaN	NaN	NaN
C3-Phenanthrenes_Anthracenes	NaN	NaN	NaN	NaN	54.62
C1-Chrysenes	NaN	NaN	NaN	NaN	2.550
Fluoranthene	NaN	NaN	NaN	142.300833	49.01
C1-Fluoranthenes_Pyrenes	NaN	NaN	NaN	NaN	35.55
Tributyltin	NaN	NaN	NaN	287.000000	130.5
C4-Phenanthrenes_Anthracenes	NaN	NaN	NaN	NaN	61.28
C2-Chrysenes	NaN	NaN	NaN	NaN	1.650
C2-Phenanthrenes_Anthracenes	NaN	NaN	NaN	NaN	63.16
Pyrene	NaN	NaN	NaN	146.133333	35.34
C3-Fluorenes	NaN	NaN	NaN	NaN	40.87
C3-Dibenzothiophenes	NaN	NaN	NaN	NaN	40.88
Copper	37.195833	33.625000	81.838278	71.787875	96.22
Chrysene	NaN	NaN	NaN	91.241250	19.67
Benzo[b]fluoranthene	NaN	NaN	NaN	49.271667	5.316
C1-Phenanthrenes_Anthracenes	NaN	NaN	NaN	NaN	36.79
C1-Naphthalenes	NaN	NaN	NaN	NaN	34.40
C3-Naphthalenes	NaN	NaN	NaN	NaN	55.48
C2-Fluorenes	NaN	NaN	NaN	NaN	25.59
C2-Dibenzothiophenes	NaN	NaN	NaN	NaN	32.97
C4-Naphthalenes	NaN	NaN	NaN	NaN	46.25
Dibutyltin	NaN	NaN	NaN	74.166667	47.00
Phenanthrene	NaN	NaN	NaN	53.645833	20.76

fiscal_year	1986.0	1987.0	1988.0	1989.0	
parameter					
C2-Naphthalenes	NaN	NaN	NaN	NaN	49.04
Benz[a]anthracene	NaN	NaN	NaN	52.060833	6.663
2-Methylnaphthalene	NaN	NaN	NaN	37.876667	17.76
...	...	...	...	...	...
BDE 1 [2-MonoBDE]	NaN	NaN	NaN	NaN	NaN
2,3,7,8-Tetrachlorodibenzofuran	NaN	NaN	NaN	NaN	NaN
BDE 126 [3,3',4,4',5-PentaBDE]	NaN	NaN	NaN	NaN	NaN
PCB158	NaN	NaN	NaN	NaN	NaN
Methyl Mercury	NaN	NaN	NaN	NaN	NaN
PCB174	NaN	NaN	NaN	NaN	NaN
PCB194	NaN	NaN	NaN	NaN	NaN
Decalin	NaN	NaN	NaN	NaN	NaN
PCB199	NaN	NaN	NaN	NaN	NaN
Clostridium perfringens - Count 2	NaN	NaN	NaN	NaN	NaN
C1-Benzothiophene	NaN	NaN	NaN	NaN	NaN
BDE 8 [2,4'-DiBDE]	NaN	NaN	NaN	NaN	NaN
BDE 3 [4-MonoBDE]	NaN	NaN	NaN	NaN	NaN
BDE 2 [3-MonoBDE]	NaN	NaN	NaN	NaN	NaN
BDE 153 [2,2',4,4',5,5'-HexaBDE]	NaN	NaN	NaN	NaN	NaN
BDE 119 [2,3',4,4',6-PentaBDE]	NaN	NaN	NaN	NaN	NaN
2,6,10-Trimethyldodecane (i-C15)	NaN	NaN	NaN	NaN	NaN
BDE 118 [2,3',4,4',5-PentaBDE]	NaN	NaN	NaN	NaN	NaN
Phytane [i-C20]	NaN	NaN	NaN	NaN	NaN
Pristane [i-C19]	NaN	NaN	NaN	NaN	NaN
BDE 11 [3,3'-DiBDE]	NaN	NaN	NaN	NaN	NaN
RETENE	NaN	NaN	NaN	NaN	NaN

fiscal_year	1986.0	1987.0	1988.0	1989.0	
parameter					
Norpristane	NaN	NaN	NaN	NaN	NaN
4/9-Methylphenanthrene	NaN	NaN	NaN	NaN	NaN
3,6-Dimethylphenanthrene	NaN	NaN	NaN	NaN	NaN
2-Methylphenanthrene	NaN	NaN	NaN	NaN	NaN
TEH	NaN	NaN	NaN	NaN	NaN
2-Methylfluoranthene	NaN	NaN	NaN	NaN	NaN
2,6,10-Trimethyltridecane (i-C16)	NaN	NaN	NaN	NaN	NaN
n-Undecane	NaN	NaN	NaN	NaN	NaN

243 rows × 22 columns

Suppose we wanted to look at specific types of pollutants- for example, all of the PCBs. We create a new dataframe containing these columns. First we get rid of the missing data here

In [43]:
```
pols.dropna(subset = ['parameter'], inplace = True)
```

Now we search for parameters that contain the letters `PCB`, and we omit the zero values.

In [44]:
```
pcbs= pols.loc[pols['parameter'].str.contains("PCB") & (pols.result > 0)]
pcbs
```

Out[44]:

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
8923	1.52	NaN	10AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
8928	0.55	NaN	10AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
8929	0.13	NaN	09AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
8937	0.62	NaN	10AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
8943	0.01	NaN	06AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
8947	0.03	NaN	09AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
8949	0.04	NaN	09AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
8953	0.55	NaN	10AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
8954	0.15	NaN	10AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
8968	0.55	NaN	10AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
8981	1.37	NaN	10AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
9000	0.13	NaN	05AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
9001	0.14	NaN	09AUG2009:00:00:00.000	2009.0	Kachemak I South Bays

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
9002	0.14	NaN	09AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
9003	0.38	NaN	10AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
9006	0.32	NaN	10AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
9038	0.03	NaN	10AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
9041	0.45	NaN	10AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
9283	0.05	NaN	09AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
9284	0.66	NaN	09AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
9285	0.01	NaN	10AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
9287	0.81	NaN	09AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
9290	1.12	NaN	10AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
9295	0.50	NaN	10AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
9297	0.02	NaN	06AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
9299	2.43	NaN	10AUG2009:00:00:00.000	2009.0	Kachemak I South Bays

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
9300	0.91	NaN	10AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
9309	0.70	NaN	10AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
9314	0.07	NaN	09AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
9316	0.09	NaN	09AUG2009:00:00:00.000	2009.0	Kachemak I South Bays
...	...	...	...	...	...
767660	0.25	NaN	NaN	1992.0	Costa Rica
767661	0.30	NaN	NaN	1992.0	Costa Rica
767662	0.25	NaN	NaN	1992.0	Costa Rica
767669	0.40	NaN	NaN	1992.0	Costa Rica
767676	0.25	NaN	NaN	1992.0	Costa Rica
767685	0.50	NaN	NaN	1992.0	Costa Rica
767687	0.50	NaN	NaN	1992.0	Costa Rica
767688	0.25	NaN	NaN	1992.0	Costa Rica
768675	0.56	NaN	27AUG2013:00:00:00.000	2013.0	Tinian

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
768891	0.15	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
768916	0.14	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
768923	0.12	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
768926	1.04	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
768945	0.35	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
768967	0.13	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
768968	0.14	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
768975	0.65	NaN	28AUG2013:00:00:00.000	2013.0	Tinian
768985	0.13	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
768996	0.29	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769006	0.28	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769010	0.70	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769066	0.12	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769109	0.11	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769168	0.19	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769172	0.03	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769197	0.03	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769223	0.04	NaN	27AUG2013:00:00:00.000	2013.0	Tinian
769248	0.03	NaN	30AUG2013:00:00:00.000	2013.0	Tinian

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
769270	0.03	NaN	30AUG2013:00:00:00.000	2013.0	Tinian
769336	0.11	NaN	30AUG2013:00:00:00.000	2013.0	Tinian

59605 rows × 22 columns

Now we can do a pivot table (`stored as a data frame`) to evaluate individual PCBs and the test years, to start to look at changes over time

In [45]:
```
pcbDF = pd.DataFrame(pcbs.pivot_table(values = 'result', columns = 'parameter'
, index = 'fiscal_year' ))
pcbDF
```

Out[45]:

parameter	PCB 126	PCB101_90	PCB105	PCB110_77	PCB112	PCB118	PCB126	
fiscal_year								
1988.0	NaN	6.669840	6.865758	NaN	NaN	7.997774	NaN	1
1989.0	NaN	10.205556	3.635030	NaN	NaN	7.858801	NaN	1
1990.0	NaN	7.293068	5.155409	NaN	NaN	5.897367	NaN	1
1991.0	NaN	13.581287	9.126319	NaN	NaN	8.477709	59.787500	4
1992.0	NaN	11.719558	5.640506	16.375269	NaN	10.503778	1.841795	4
1993.0	NaN	8.845867	6.587232	18.666667	NaN	10.441948	3.560552	3
1994.0	NaN	5.591576	3.965080	NaN	77.0	5.165531	NaN	2
1995.0	NaN	9.599819	3.041290	NaN	NaN	7.552485	NaN	2
1996.0	NaN	5.327842	2.166108	NaN	NaN	4.838550	NaN	1
1997.0	NaN	3.986236	1.172645	NaN	NaN	5.423298	49.526342	1
1998.0	0.3455	5.366655	3.221071	NaN	NaN	4.049341	100.566667	1
1999.0	NaN	4.582415	1.402955	NaN	NaN	3.114884	NaN	1
2000.0	NaN	5.109532	1.563924	NaN	NaN	3.694891	NaN	1
2001.0	NaN	2.905493	0.853721	NaN	NaN	2.877624	NaN	0
2002.0	NaN	4.595043	1.052469	NaN	NaN	3.712070	NaN	1
2003.0	NaN	7.949357	2.141667	NaN	NaN	5.974296	NaN	1
2004.0	NaN	13.549103	6.949356	0.288667	NaN	19.225878	NaN	6
2005.0	NaN	2.185189	0.524713	1.469554	NaN	1.079053	NaN	0
2006.0	NaN	8.513827	1.284000	5.010270	NaN	3.925789	NaN	1
2007.0	NaN	2.856069	0.777848	3.543238	NaN	2.886723	NaN	1
2008.0	NaN	7.821532	1.728947	7.868554	NaN	4.267190	NaN	1
2009.0	NaN	3.617727	0.823974	2.756301	NaN	2.635833	NaN	1
2010.0	NaN	3.283134	1.744483	3.004872	NaN	2.376275	NaN	0
2011.0	NaN	2.001429	3.331667	4.593750	NaN	4.104000	NaN	0
2012.0	NaN	3.087308	1.453333	5.378889	NaN	3.826875	NaN	0
2013.0	NaN	1.213750	1.790000	2.640000	NaN	2.363333	NaN	1
2014.0	NaN	1.033200	NaN	0.687273	NaN	0.723636	NaN	0

27 rows × 44 columns

let's see when the cummulative PCB values rise and fall. First we take the sum across the dataFrame Columns like this:

In [46]:
```
pcbDF['all'] = pcbDF.sum(axis = 1)
pcbDF
```

Out[46]:

parameter	PCB 126	PCB101_90	PCB105	PCB110_77	PCB112	PCB118	PCB126	
fiscal_year								
1988.0	NaN	6.669840	6.865758	NaN	NaN	7.997774	NaN	1
1989.0	NaN	10.205556	3.635030	NaN	NaN	7.858801	NaN	1
1990.0	NaN	7.293068	5.155409	NaN	NaN	5.897367	NaN	1
1991.0	NaN	13.581287	9.126319	NaN	NaN	8.477709	59.787500	4
1992.0	NaN	11.719558	5.640506	16.375269	NaN	10.503778	1.841795	4
1993.0	NaN	8.845867	6.587232	18.666667	NaN	10.441948	3.560552	3
1994.0	NaN	5.591576	3.965080	NaN	77.0	5.165531	NaN	2
1995.0	NaN	9.599819	3.041290	NaN	NaN	7.552485	NaN	2
1996.0	NaN	5.327842	2.166108	NaN	NaN	4.838550	NaN	1
1997.0	NaN	3.986236	1.172645	NaN	NaN	5.423298	49.526342	1
1998.0	0.3455	5.366655	3.221071	NaN	NaN	4.049341	100.566667	1
1999.0	NaN	4.582415	1.402955	NaN	NaN	3.114884	NaN	1
2000.0	NaN	5.109532	1.563924	NaN	NaN	3.694891	NaN	1
2001.0	NaN	2.905493	0.853721	NaN	NaN	2.877624	NaN	0
2002.0	NaN	4.595043	1.052469	NaN	NaN	3.712070	NaN	1
2003.0	NaN	7.949357	2.141667	NaN	NaN	5.974296	NaN	1
2004.0	NaN	13.549103	6.949356	0.288667	NaN	19.225878	NaN	6
2005.0	NaN	2.185189	0.524713	1.469554	NaN	1.079053	NaN	0
2006.0	NaN	8.513827	1.284000	5.010270	NaN	3.925789	NaN	1
2007.0	NaN	2.856069	0.777848	3.543238	NaN	2.886723	NaN	1
2008.0	NaN	7.821532	1.728947	7.868554	NaN	4.267190	NaN	1
2009.0	NaN	3.617727	0.823974	2.756301	NaN	2.635833	NaN	1
2010.0	NaN	3.283134	1.744483	3.004872	NaN	2.376275	NaN	0
2011.0	NaN	2.001429	3.331667	4.593750	NaN	4.104000	NaN	0
2012.0	NaN	3.087308	1.453333	5.378889	NaN	3.826875	NaN	0
2013.0	NaN	1.213750	1.790000	2.640000	NaN	2.363333	NaN	1
2014.0	NaN	1.033200	NaN	0.687273	NaN	0.723636	NaN	0

27 rows × 45 columns

Next, we sort by the `sum` value that we just stored in the `all` column

In [47]:
```
pcbDF.sort_values('all', ascending = False)
```

Out[47]:

parameter	PCB 126	PCB101_90	PCB105	PCB110_77	PCB112	PCB118	PCB126	
fiscal_year								
2004.0	NaN	13.549103	6.949356	0.288667	NaN	19.225878	NaN	6
1991.0	NaN	13.581287	9.126319	NaN	NaN	8.477709	59.787500	4
1998.0	0.3455	5.366655	3.221071	NaN	NaN	4.049341	100.566667	1
1992.0	NaN	11.719558	5.640506	16.375269	NaN	10.503778	1.841795	4
2008.0	NaN	7.821532	1.728947	7.868554	NaN	4.267190	NaN	1
1997.0	NaN	3.986236	1.172645	NaN	NaN	5.423298	49.526342	1
1994.0	NaN	5.591576	3.965080	NaN	77.0	5.165531	NaN	2
2006.0	NaN	8.513827	1.284000	5.010270	NaN	3.925789	NaN	1
1993.0	NaN	8.845867	6.587232	18.666667	NaN	10.441948	3.560552	3
1995.0	NaN	9.599819	3.041290	NaN	NaN	7.552485	NaN	2
2009.0	NaN	3.617727	0.823974	2.756301	NaN	2.635833	NaN	1
2003.0	NaN	7.949357	2.141667	NaN	NaN	5.974296	NaN	1
2012.0	NaN	3.087308	1.453333	5.378889	NaN	3.826875	NaN	0
2013.0	NaN	1.213750	1.790000	2.640000	NaN	2.363333	NaN	1
2007.0	NaN	2.856069	0.777848	3.543238	NaN	2.886723	NaN	1
1989.0	NaN	10.205556	3.635030	NaN	NaN	7.858801	NaN	1
1988.0	NaN	6.669840	6.865758	NaN	NaN	7.997774	NaN	1
2010.0	NaN	3.283134	1.744483	3.004872	NaN	2.376275	NaN	0
1990.0	NaN	7.293068	5.155409	NaN	NaN	5.897367	NaN	1
2005.0	NaN	2.185189	0.524713	1.469554	NaN	1.079053	NaN	0
2011.0	NaN	2.001429	3.331667	4.593750	NaN	4.104000	NaN	0
2000.0	NaN	5.109532	1.563924	NaN	NaN	3.694891	NaN	1
1996.0	NaN	5.327842	2.166108	NaN	NaN	4.838550	NaN	1
2002.0	NaN	4.595043	1.052469	NaN	NaN	3.712070	NaN	1
1999.0	NaN	4.582415	1.402955	NaN	NaN	3.114884	NaN	1
2001.0	NaN	2.905493	0.853721	NaN	NaN	2.877624	NaN	0
2014.0	NaN	1.033200	NaN	0.687273	NaN	0.723636	NaN	0

27 rows × 45 columns

Let's try to determine WHY 2004 (the most PCBs) was more polluted than the previous decade. Were more tests done? Different types of tests? Different areas tested? We will go back to our pcb dataframe to look at this information more closely.

In [49]:
```
oh494 = pcbs.loc[(pcbs.fiscal_year == 2004) | (pcbs.fiscal_year == 1994)]
## the pipe operater denotes "or"
oh494
```

Out[49]:

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
129681	0.030	NaN	07JUN2004:00:00:00.000	2004.0	Cape Cod
129685	0.430	NaN	08JUN2004:00:00:00.000	2004.0	Cape Cod
129687	0.050	NaN	11JUN2004:00:00:00.000	2004.0	Area Betwe Bays
129689	2.090	NaN	08JUN2004:00:00:00.000	2004.0	Boston Har
129690	0.210	NaN	08JUN2004:00:00:00.000	2004.0	Stellwagen
129692	2.510	NaN	09JUN2004:00:00:00.000	2004.0	Boston Har
129694	0.270	NaN	09JUN2004:00:00:00.000	2004.0	Boston Har
129697	0.540	NaN	10JUN2004:00:00:00.000	2004.0	Stellwagen
129699	0.470	NaN	09JUN2004:00:00:00.000	2004.0	Boston Har
129701	0.430	NaN	08JUN2004:00:00:00.000	2004.0	Massachus Bay
129702	0.020	NaN	10JUN2004:00:00:00.000	2004.0	Area Betwe Bays
129711	0.500	NaN	11JUN2004:00:00:00.000	2004.0	Area Betwe Bays
129712	1.430	NaN	08JUN2004:00:00:00.000	2004.0	Boston Har Deer Island
129715	0.080	NaN	08JUN2004:00:00:00.000	2004.0	Massachus Bay
129717	0.400	NaN	10JUN2004:00:00:00.000	2004.0	Stellwagen
129723	0.160	NaN	10JUN2004:00:00:00.000	2004.0	Area Betwe Bays
129725	0.610	NaN	10JUN2004:00:00:00.000	2004.0	Stellwagen
129726	0.820	NaN	09JUN2004:00:00:00.000	2004.0	Boston Har
129727	0.080	NaN	08JUN2004:00:00:00.000	2004.0	Stellwagen

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
129728	0.520	NaN	10JUN2004:00:00:00.000	2004.0	Outfall
129730	0.010	NaN	09JUN2004:00:00:00.000	2004.0	Stellwagen
129731	0.510	NaN	09JUN2004:00:00:00.000	2004.0	Boston Har
129732	0.060	NaN	08JUN2004:00:00:00.000	2004.0	Cape Cod
129733	0.100	NaN	11JUN2004:00:00:00.000	2004.0	Cape Cod
129741	0.450	NaN	09JUN2004:00:00:00.000	2004.0	Stellwagen
129744	1.310	NaN	09JUN2004:00:00:00.000	2004.0	Boston Har
129745	2.890	NaN	08JUN2004:00:00:00.000	2004.0	Boston Har
129746	0.030	NaN	10JUN2004:00:00:00.000	2004.0	Area Betwe Bays
129747	0.280	NaN	09JUN2004:00:00:00.000	2004.0	Massachus Bay
129748	0.180	NaN	10JUN2004:00:00:00.000	2004.0	Massachus Bay
...	...	...	...	...	...
723982	11.104	NaN	NaN	2004.0	New York E
723983	13.910	NaN	23NOV2003:00:00:00.000	2004.0	Hudson/Ra Estuary
723984	0.210	NaN	23NOV2003:00:00:00.000	2004.0	Hudson/Ra Estuary
723986	1.650	NaN	24NOV2003:00:00:00.000	2004.0	Hudson/Ra Estuary
723988	1.280	NaN	24NOV2003:00:00:00.000	2004.0	Hudson/Ra Estuary
723991	6.880	NaN	24NOV2003:00:00:00.000	2004.0	Hudson/Ra Estuary
723995	17.841	NaN	NaN	2004.0	New York E

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
723997	25.420	NaN	24NOV2003:00:00:00.000	2004.0	Hudson/Ra Estuary
723998	35.020	NaN	23NOV2003:00:00:00.000	2004.0	Hudson/Ra Estuary
723999	9.978	NaN	NaN	2004.0	Hudson/Ra Estuary
724001	15.600	NaN	23NOV2003:00:00:00.000	2004.0	Hudson/Ra Estuary
724007	11.722	NaN	NaN	2004.0	Hudson/Ra Estuary
724008	21.590	NaN	24NOV2003:00:00:00.000	2004.0	Hudson/Ra Estuary
724011	6.335	NaN	NaN	2004.0	New York E
724013	3.943	NaN	NaN	2004.0	Hudson/Ra Estuary
724016	1.279	NaN	NaN	2004.0	New York E
724020	21.510	NaN	23NOV2003:00:00:00.000	2004.0	Hudson/Ra Estuary
724022	17.820	NaN	24NOV2003:00:00:00.000	2004.0	Hudson/Ra Estuary
724023	17.669	NaN	NaN	2004.0	Hudson/Ra Estuary
724024	21.280	NaN	23NOV2003:00:00:00.000	2004.0	Hudson/Ra Estuary
724025	44.970	NaN	23NOV2003:00:00:00.000	2004.0	Hudson/Ra Estuary
724028	22.013	NaN	NaN	2004.0	Hudson/Ra Estuary
724030	45.480	NaN	24NOV2003:00:00:00.000	2004.0	Hudson/Ra Estuary
724032	20.208	NaN	NaN	2004.0	Hudson/Ra Estuary
724034	42.163	NaN	NaN	2004.0	Hudson/Ra Estuary
724040	5.922	NaN	NaN	2004.0	Hudson/Ra Estuary

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
724041	16.920	NaN	23NOV2003:00:00:00.000	2004.0	Hudson/Ra Estuary
724042	10.075	NaN	NaN	2004.0	Hudson/Ra Estuary
724044	0.091	NaN	NaN	2004.0	Hudson/Ra Estuary
724045	1.313	NaN	NaN	2004.0	Hudson/Ra Estuary

5247 rows × 22 columns

How many tests were from 2004?

```
In [50]: len(oh494.loc[oh494.fiscal_year==2004])
```

```
Out[50]: 3509
```

How many tests were from 1994?

```
In [51]: len(oh494.loc[oh494.fiscal_year==1994])
```

```
Out[51]: 1738
```

That's quite a difference- more than twice the number of tests What locations were tested in each of those years?

In [52]:
```
oh494[oh494.fiscal_year==2004].general_location.unique()
```

Out[52]:
```
array(['Cape Cod Bay', 'Area Between Bays', 'Boston Harbor',
       'Stellwagen Basin', 'Massachusetts Bay',
       'Boston Harbor Deer Island', 'Outfall', 'Stellwagen Bank',
       'Reference site', 'St Lucie - South Fork',
       'St Lucie - Middle Estuary', 'St Lucie - Convergence Zone',
       'St Lucie - North Fork', 'St Lucie - Lower Estuary',
       'Soquel Canyon', 'Pioneer Canyon', 'Between Pt Lobos and Pt Sur',
       'Ano Nuevo Canyon', 'Ascencion Canyon',
       'Between Ascenion and Pioneer Canyons', 'Monterey Canyon',
       'Between Soquel and Cabrillo Canyon', 'Ano Nuevo',
       'Southeast of Pt Sur', 'Carmel Canyon', 'San Antonio Bay',
       'Lower Laguna Madre', 'Espiritu Santo', 'Lake Michigan',
       'Green Bay', 'Saginaw Bay', 'Lake Huron', 'Traverse Bay',
       'Florida Bay', 'Pensacola Bay', 'Tampa Bay', 'Choctawhatchee Bay',
       'Mobile Bay', 'Lake Pontchartrain', 'Terrebonne Bay',
       'Lake Borgne', 'Mississippi Sound', 'Rookery Bay', 'Breton Sound',
       'Everglades', 'Naples Bay', 'Honolulu Hrb.', 'Hawaii',
       'Hudson/Raritan Estuary', 'New York Bight'], dtype=object)
```

In [53]:
```
oh494[oh494.fiscal_year==1994].general_location.unique()
```

Out[53]:
```
array(['Port Wentworth', 'Kings Island Channel',
       'Marsh Island Turning Basin', 'Savannah River- Stevens Terminal',
       'Upper Savannah River', 'Savannah River', 'Marsh Island Channel',
       'Savannah River- Front Channel', 'Savannah River- Wrecks Channel',
       'Savannah River - South Channel',
       'Savannah River- Suspension Bridge', 'Kings Island Turning Basin',
       'Savannah River- Lower Channel', 'Back River', 'Whiteball Channel',
       'Savannah River- Ocean Terminal', 'Finger Canal', 'Academy Creek',
       'Turtle River', 'East River', 'St Simon Sound', 'Purvis Creek',
       'Apalachicola Bay', 'Garnier Bayou', "Tom's Bayou",
       'East Chochawhatchee Bay', 'Boggy Bayou', 'Hand Cove',
       'Rocky Bayou', 'Destin Harbor', 'Cinco Bayou',
       'West Chochtawhatchee Bay', 'La Grange Bayou', 'Dons Bayou',
       'Central Chochtawhatchee Bay', 'Newport Bay', 'Del Mar Beach',
       'Dana Point', 'La Jolla Canyon', 'Santa Margarita River',
       'Aransas Bay', 'San Antonio Bay', 'Espiritu Santo',
       'Lower Laguna Madre', 'Matagorda Bay', 'Mesquite Bay',
       'Corpus Christi', 'Brazos River', 'Lake Erie', 'Lake Ontario',
       'Niagara River', 'Lake Michigan', 'Green Bay', 'Pensacola Bay',
       'Tampa Bay', 'Mississippi River', 'Choctawhatchee Bay',
       'Caillou Lake', 'Cedar Key', 'Mississippi Sound', 'Galveston Bay',
       'Vermilion Bay', 'Sabine Lake', 'Calcasieu Lake', 'Lake Borgne',
       'Breton Sound', 'Florida Bay', 'Mobile Bay', 'Terrebonne Bay',
       'Apalachee Bay', 'Panama City', "Barber's Point", 'Honolulu Hrb.',
       'Bristol Bay', 'Norton Sound'], dtype=object)
```

In [ ]:
```
Lake Michigan was tested in both of those years, lets evaluate PCBs there in m
ore detail
```

In [54]:
```
LM = oh494.loc[oh494.general_location == 'Lake Michigan']
LM
```

Out[54]:

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
506467	8.927	NaN	08SEP2004:00:00:00.000	2004.0	Lake Michi
506670	2.083	NaN	09SEP2004:00:00:00.000	2004.0	Lake Michi
506952	2.664	NaN	15AUG1994:00:00:00.000	1994.0	Lake Michi
507038	1.114	NaN	07SEP2004:00:00:00.000	2004.0	Lake Michi
507380	10.772	NaN	08SEP2004:00:00:00.000	2004.0	Lake Michi
507517	5.632	NaN	08SEP2004:00:00:00.000	2004.0	Lake Michi
507526	11.959	NaN	22AUG1994:00:00:00.000	1994.0	Lake Michi
507679	4.404	NaN	16AUG1994:00:00:00.000	1994.0	Lake Michi
507794	1.344	NaN	08SEP2004:00:00:00.000	2004.0	Lake Michi
508477	44.335	NaN	18AUG1994:00:00:00.000	1994.0	Lake Michi
508491	6.289	NaN	18AUG1994:00:00:00.000	1994.0	Lake Michi
508545	16.713	NaN	16AUG1994:00:00:00.000	1994.0	Lake Michi
508850	9.162	NaN	15AUG1994:00:00:00.000	1994.0	Lake Michi

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
508870	0.912	NaN	16AUG1994:00:00:00.000	1994.0	Lake Michi
509081	4.448	NaN	18AUG1994:00:00:00.000	1994.0	Lake Michi
509138	3.965	NaN	16AUG1994:00:00:00.000	1994.0	Lake Michi
509150	4.484	NaN	09SEP2004:00:00:00.000	2004.0	Lake Michi
509187	1.174	NaN	08SEP2004:00:00:00.000	2004.0	Lake Michi
509196	2.597	NaN	22AUG1994:00:00:00.000	1994.0	Lake Michi
509532	1.458	NaN	16AUG1994:00:00:00.000	1994.0	Lake Michi
509790	1.055	NaN	08SEP2004:00:00:00.000	2004.0	Lake Michi
509899	1.631	NaN	22AUG1994:00:00:00.000	1994.0	Lake Michi
510201	2.072	NaN	08SEP2004:00:00:00.000	2004.0	Lake Michi
510229	1.236	NaN	15AUG1994:00:00:00.000	1994.0	Lake Michi
510620	6.712	NaN	07SEP2004:00:00:00.000	2004.0	Lake Michi
510927	31.941	NaN	09SEP2004:00:00:00.000	2004.0	Lake Michi

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
511820	1.235	NaN	07SEP2004:00:00:00.000	2004.0	Lake Michi
512070	1.830	NaN	08SEP2004:00:00:00.000	2004.0	Lake Michi
512074	1.996	NaN	08SEP2004:00:00:00.000	2004.0	Lake Michi
512114	8.283	NaN	07SEP2004:00:00:00.000	2004.0	Lake Michi
...	...	...	...	...	...
518668	20.319	NaN	08SEP2004:00:00:00.000	2004.0	Lake Michi
518672	10.280	NaN	08SEP2004:00:00:00.000	2004.0	Lake Michi
518681	19.770	NaN	09SEP2004:00:00:00.000	2004.0	Lake Michi
518714	9.770	NaN	08SEP2004:00:00:00.000	2004.0	Lake Michi
518748	1.611	NaN	16AUG1994:00:00:00.000	1994.0	Lake Michi
518849	13.871	NaN	08SEP2004:00:00:00.000	2004.0	Lake Michi
518857	21.585	NaN	16AUG1994:00:00:00.000	1994.0	Lake Michi
518941	7.130	NaN	16AUG1994:00:00:00.000	1994.0	Lake Michi
518991	10.230	NaN	07SEP2004:00:00:00.000	2004.0	Lake Michi

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
519224	25.829	NaN	07SEP2004:00:00:00.000	2004.0	Lake Michi
519248	96.689	NaN	18AUG1994:00:00:00.000	1994.0	Lake Michi
519256	44.563	NaN	08SEP2004:00:00:00.000	2004.0	Lake Michi
519357	29.013	NaN	22AUG1994:00:00:00.000	1994.0	Lake Michi
519358	35.405	NaN	15AUG1994:00:00:00.000	1994.0	Lake Michi
519364	0.873	NaN	07SEP2004:00:00:00.000	2004.0	Lake Michi
519462	51.360	NaN	18AUG1994:00:00:00.000	1994.0	Lake Michi
519500	1.930	NaN	08SEP2004:00:00:00.000	2004.0	Lake Michi
519542	15.662	NaN	22AUG1994:00:00:00.000	1994.0	Lake Michi
519575	5.801	NaN	16AUG1994:00:00:00.000	1994.0	Lake Michi
519602	7.978	NaN	15AUG1994:00:00:00.000	1994.0	Lake Michi
519640	1.469	NaN	08SEP2004:00:00:00.000	2004.0	Lake Michi
519696	14.628	NaN	16AUG1994:00:00:00.000	1994.0	Lake Michi

	result	coastal_ecological_area	collection_date	fiscal_year	general_lo
519879	5.007	NaN	07SEP2004:00:00:00.000	2004.0	Lake Michi
519939	81.978	NaN	18AUG1994:00:00:00.000	1994.0	Lake Michi
520004	7.057	NaN	08SEP2004:00:00:00.000	2004.0	Lake Michi
520007	8.316	NaN	22AUG1994:00:00:00.000	1994.0	Lake Michi
520076	43.724	NaN	16AUG1994:00:00:00.000	1994.0	Lake Michi
520244	4.309	NaN	07SEP2004:00:00:00.000	2004.0	Lake Michi
520378	23.470	NaN	09SEP2004:00:00:00.000	2004.0	Lake Michi
520492	6.452	NaN	22AUG1994:00:00:00.000	1994.0	Lake Michi

168 rows × 22 columns

Let's do another pivot table to look at the PCB changes in Lake Michigan

In [56]:
```
LM.iloc[0]
```

Out[56]:
```
result                                                    8.927
coastal_ecological_area                                     NaN
collection_date                             08SEP2004:00:00:00.000
fiscal_year                                                2004
general_location                                   Lake Michigan
latitude                                                    NaN
longitude                                                   NaN
matrix                                        Zebra Mussel Tissue
method                                                  ECDDUAL.M
nst_sample_id                                        MW2004LMHMDS
nst_site                                                 LMHM
parameter                                               PCB187
parameter_name                                              NaN
qualifier                                                   NaN
region_name                                                 NaN
scientific_name                                 Dreissena species
source_file                MusselWatch_GreatLakes_Organics_Tissue.csv
specific_location                                  Hammod Marina
state_name                                                  NaN
stratum                                                     NaN
study_name                                         Mussel Watch
units                                                  ng/dry g
Name: 506467, dtype: object
```

In [59]:
```
LM2 = LM.pivot_table(values = 'result', columns = 'fiscal_year', index = 'para
meter' )
LM2
```

Out[59]:

fiscal_year	1994.0	2004.0
parameter		
PCB101_90	67.83140	16.1100
PCB105	32.63360	8.4136
PCB118	44.31820	25.9658
PCB128	11.87900	5.4352
PCB138_160	45.16620	34.5640
PCB153_132_168	32.38780	48.4056
PCB170_190	11.98425	8.3340
PCB18	18.10100	12.2850
PCB180	24.46320	13.9148
PCB187	17.31460	12.7968
PCB195_208	2.91720	2.1438
PCB206	2.32620	1.5336
PCB209	4.85425	0.8898
PCB28	21.31740	24.2220
PCB44	32.14400	31.1454
PCB52	52.88620	45.8236
PCB66	31.72000	20.4586
PCB8_5	4.45600	NaN

which PCB's changed the most in Lake Michigan during that decade?

In [74]:
```
LM2.diff(axis = 1)
```

Out[74]:

fiscal_year	1994.0	2004.0
parameter		
PCB101_90	NaN	-51.72140
PCB105	NaN	-24.22000
PCB118	NaN	-18.35240
PCB128	NaN	-6.44380
PCB138_160	NaN	-10.60220
PCB153_132_168	NaN	16.01780
PCB170_190	NaN	-3.65025
PCB18	NaN	-5.81600
PCB180	NaN	-10.54840
PCB187	NaN	-4.51780
PCB195_208	NaN	-0.77340
PCB206	NaN	-0.79260
PCB209	NaN	-3.96445
PCB28	NaN	2.90460
PCB44	NaN	-0.99860
PCB52	NaN	-7.06260
PCB66	NaN	-11.26140
PCB8_5	NaN	NaN

well, that's great, but how do I work with that data? Let's make a new DataFrame

In [95]:
```
LMfnl = LM2.diff(axis = 1)
LMfnl
```

Out[95]:

fiscal_year	1994.0	2004.0
parameter		
PCB101_90	NaN	-51.72140
PCB105	NaN	-24.22000
PCB118	NaN	-18.35240
PCB128	NaN	-6.44380
PCB138_160	NaN	-10.60220
PCB153_132_168	NaN	16.01780
PCB170_190	NaN	-3.65025
PCB18	NaN	-5.81600
PCB180	NaN	-10.54840
PCB187	NaN	-4.51780
PCB195_208	NaN	-0.77340
PCB206	NaN	-0.79260
PCB209	NaN	-3.96445
PCB28	NaN	2.90460
PCB44	NaN	-0.99860
PCB52	NaN	-7.06260
PCB66	NaN	-11.26140
PCB8_5	NaN	NaN

is this a dataframe? Let's find out

In [78]:
```
LMfnl.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 18 entries, PCB101_90 to PCB8_5
Data columns (total 2 columns):
1994.0    0 non-null float64
2004.0    17 non-null float64
dtypes: float64(2)
memory usage: 432.0+ bytes
```

Now, let's see which PCBs are responsible for the biggest changes

In [97]:
```
LMfnl.sort_values(2004.0)
```

Out[97]:

fiscal_year	1994.0	2004.0
parameter		
PCB101_90	NaN	-51.72140
PCB105	NaN	-24.22000
PCB118	NaN	-18.35240
PCB66	NaN	-11.26140
PCB138_160	NaN	-10.60220
PCB180	NaN	-10.54840
PCB52	NaN	-7.06260
PCB128	NaN	-6.44380
PCB18	NaN	-5.81600
PCB187	NaN	-4.51780
PCB209	NaN	-3.96445
PCB170_190	NaN	-3.65025
PCB44	NaN	-0.99860
PCB206	NaN	-0.79260
PCB195_208	NaN	-0.77340
PCB28	NaN	2.90460
PCB153_132_168	NaN	16.01780
PCB8_5	NaN	NaN

well, we have a conclusion: PCB101_90 decreased from 1994 to 2004, while PCB153_132_168 is responsible for the biggest increase.

I hope you enjoyed reading- come back if you need help with the companion exercises. Everything you need to know is right here!