```
In [1]: import pandas as pd
        import numpy as np
        from IPython.core.display import display, HTML
        display(HTML("<style>.container { width:100% !important; }</style>"))
```

The Department of Human Services provided us with seven spreadsheets that each have information about substantiated complaints against assisted living, residential care, and nursing facilities in Oregon. The purpose of this notebook is to mung them, standardizing the column names, removing unnecessary columns, and cleaning some fields. The second purpose of this notebook is to get initial ownership date for facilities from the owner_history table and assign it to each facility.

Complaints mung

Start with the 10-year data that do not have narratives.

Import, clean, concat.

```
In [2]: #Five years of detailed complaint data for all four kinds of facilities (Resid
        ential Care, Assisted Living, Nursing, and Adult Foster Home)
        detailed = pd.read_excel('../../data/raw/Oregonian Abuse records 5 years May 2
        016.xlsx', header=3)
        #Ten years of non-detailed complaints for Nursing Facilities
        NF_complaints = pd.read_excel('../../data/raw/Copy of Oregonian Data Request F
        acility Abuse Records April 2016   Reviewed.xlsx',sheetname='NF Complaints')
        #Ten years of non-detailed complaints for Assisted Living Facilities
        ALF_complaints = pd.read_excel('../../data/raw/Copy of Oregonian Data Request
         Facility Abuse Records April 2016   Reviewed.xlsx',sheetname='ALF Complaints'
        )
        #Ten years of non-detailed complaints for Residential Care Facilities
        RCF_complaints = pd.read_excel('../../data/raw/Copy of Oregonian Data Request
         Facility Abuse Records April 2016   Reviewed.xlsx',sheetname='RCF Complaints'
        )
```

```
In [3]: #NF has an inconsistently named column
        NF_complaints.rename(columns={'Abuse_CbcAbuse': 'CbcAbuse'}, inplace=True)
```

```
In [4]: ten_year_complaints = pd.concat([RCF_complaints,ALF_complaints,NF_complaints],
        ignore_index=True).reset_index().drop('index',1)
```

```
In [5]: ten_year_complaints.rename(columns={'Abuse_Number':'abuse_number', 'Facility I
        D':'facility_id','Incident Date':'incident_date','Fac Type': 'facility_type',
        'FineAssessed':'fine',
        'Investigation Results':'results_1','FacilityInvestResultsAbuse':'results_2',
        'FacilityInvestResultsRule':'results_3','OutcomeCode':'outcome_code',
        'CbcAbuse':'abuse_type'}, inplace=True)
```

```
In [6]: ten_year_complaints = ten_year_complaints[['abuse_number','facility_id','incid
        ent_date','results_1','fine',
                                            'results_2','results_3','outcome_co
        de','abuse_type']][ten_year_complaints['abuse_number'].notnull()]
```

There are 52 complaints that have been mislabelled as unsubstantiated.

```
In [7]: sub_comps = pd.read_excel('../../data/raw/52 mislabelled as unsubstantiated.xl
        sx', header=None, names=['abuse_number'])
```

```
In [8]: miss_comps = sub_comps.merge(ten_year_complaints, how = 'left', left_on='abuse
        _number',right_on='abuse_number')#.count()
```

This dataset contains unsubstantiated complaints, which we don't need. There are three columns that indicate substantiation. A DHS person explained that if any one of them has the word 'substantiated,' then the complaint was substantiated.

```
In [9]: ten_year_complaints = ten_year_complaints[(ten_year_complaints['results_1']==
        'Substantiated')|
                            (ten_year_complaints['results_2']=='Substantiated')|
                            (ten_year_complaints['results_3']=='Substantiated')]
```

```
In [10]: ten_year_complaints = pd.concat([ten_year_complaints,miss_comps]).reset_index
         ().drop('index',1)
```

```
In [11]: ten_year_ready = ten_year_complaints[['abuse_number','facility_id','incident_d
         ate','outcome_code','abuse_type', 'fine']].reset_index().drop('index',1)
```

Now we prepare the five-year, detailed data

The 'detailed' data is a five-year set of substantiated complaints against all facilities, including adult foster homes, which we don't want.

```
In [12]: detailed.rename(columns={'Abuse_Number':'abuse_number','Facility ID':'facility
         _id',
                        'Incident Date':'incident_date','Investigation Results':'resul
         ts_1',
                        'Facility Invest Results Abuse':'results_2','Facility Invest R
         esults Rule':'results_3',
                        'Outcome Code':'outcome_code','Action Notes':'action_notes','Ou
         tcome Notes':'outcome_notes',
                        'Cbc Abuse Indicator':'abuse_type', 'Facility Type':'facility_t
         ype', 'Fine Assessed':'fine'}, inplace=True)
```

Drop Adult Foster Homes and select columns.

```
In [13]: five_year_complaints = detailed[['abuse_number','facility_id','facility_type',
         'incident_date','outcome_code',
                        'action_notes','outcome_notes','fine','abuse_type']][det
         ailed['facility_type']!='AFH']
```

No longer need the facility_type field.

```
In [14]: five_year_ready = five_year_complaints.drop('facility_type',1)
```

There are thousands of complaints that appear in both datasets. If a complaint is a duplicate, we want to keep the one that is in the five-year set, because that one has richer data. To do this, we will add a 'source' column to each dataframe, value '1' for the five-year data and '2' for the ten-year data. We will then sort based on that column, then de-duplicate on the abuse_number field, telling pandas to keep the first instance of the duplicate that it finds.

```
In [15]: five_year_ready['source']=1
```

```
In [16]: ten_year_ready['source']=2
```

```
In [17]: five_ten_concat = pd.concat([five_year_ready,ten_year_ready])
```

Set abuse_numbers to uppercase (three abuse numbers in ten-year data have lowercase)

```
In [18]: five_ten_concat['abuse_number'] = five_ten_concat['abuse_number'].apply(lambda
         x:x.upper())
```

```
In [19]: five_ten_concat = five_ten_concat.sort_values('source')
```

```
In [20]: complaints = five_ten_concat.drop_duplicates(subset='abuse_number', keep='firs
         t').reset_index().drop('index',1)
```

Add a 'year' column based on incident date.

```
In [21]: complaints['year']=complaints['incident_date'].dt.year.astype(int)
```

```
In [22]: complaints.count()
```

```
Out[22]: abuse_number    13705
         abuse_type      12478
         action_notes     6574
         facility_id     13705
         fine            10343
         incident_date   13705
         outcome_code    13704
         outcome_notes    6544
         source          13705
         year            13705
         dtype: int64
```

```
In [23]: complaints['abuse_type'].fillna('',inplace=True)
```

Clean the abuse_type column

```
In [24]: complaints['abuse_type'] = complaints['abuse_type'].apply(lambda x: x.upper())
```

```
In [25]: complaints["abuse_type"] = complaints["abuse_type"].apply(dict([
             ('0', ''),
             ('1', ''),
             ('2', ''),
             ('363', ''),
             ('I', ''),
             ('A', 'A'),
             ('L', 'L'),
         ]).get).fillna('')
```

Join with scraped complaints — 3/29/2017

Complaints were scraped from https://apps.state.or.us/cf2/spd/facility_complaints/ (https://apps.state.or.us/cf2/spd/facility_complaints/) using the script in ..scraper/DHS_scraper.py

```
In [26]: scraped_comp_3_29 = pd.read_csv('../../data/scraped/scraped_complaints_3_29.csv')
```

Set all abuse numbers to upper case.

```python
In [27]: scraped_comp_3_29['abuse_number'] = scraped_comp_3_29['abuse_number'].apply(la
         mbda x: x.upper())
```

```python
In [28]: scraped_comp_3_29 = scraped_comp_3_29.drop_duplicates(subset='abuse_number').d
         rop(['fac_type','inv_comp_date','city_name'],1)
```

```python
In [29]: merged = complaints.merge(scraped_comp_3_29, how = 'left',on = 'abuse_number')
```

```python
In [30]: merged['outcome_code'] = merged['outcome_code'].fillna(0)
```

Add a column that tells us if the complaint has an equivalent online, based on the present of the online name.

```python
In [31]: merged['public'] = np.where(merged['online_fac_name'].notnull(),'online','offl
         ine')
```

Join to a lookup table for the code number

```python
In [32]: codes = pd.read_excel('../../data/raw/OLRO Outcome Codes.xlsx', header=3)
         codes.rename(columns = {'Code':'outcome_code','Display Text':'outcome'}, inpla
         ce = True)
         codes['outcome_code'] = codes['outcome_code'].astype(str)
         codes = codes.drop('Definition',1)
```

```python
In [33]: merged['outcome_code'] = merged['outcome_code'].astype(int).astype(str)
```

```python
In [34]: merged = merged.merge(codes, how = 'left')
```

```python
In [36]: merged['online_fac_name'].fillna('',inplace=True)
```

Join with facilities

First, prep the facilities.

```python
In [37]: facilities = pd.read_csv('../../data/raw/APD_FacilityRecords.csv')
```

```python
In [38]: facilities.rename(columns={'FACID':'facid','Facility ID':'facility_id','FAC_CC
         MUNumber':'fac_ccmunumber','FAC_Type':'facility_type',
                                     'FAC_Capacity':'fac_capacity','Facility Name':'facil
         ity_name','Facility Address':'street',
                                     'Other Service':'other_service','Owner':'owner','Ope
         rator':'operator'}, inplace=True)
```

Select the columns we need and drop the one duplicate in here.

```
In [39]: facilities = facilities[['facility_id','fac_ccmunumber','facility_type','fac_c
         apacity','facility_name']].drop_duplicates(subset='facility_id', keep='last')
```

Churchill Estates Residential Care has blank facility_type and capacity fields. The facility is an RCF.

```
In [40]: facilities.loc[318,'facility_type']='RCF'
         facilities.loc[318,'fac_capacity']=108
```

Left join facilities to complaints.

This eliminates complaints without facilities.

```
In [41]: merged_comp_fac = facilities.merge(merged, on = 'facility_id',how = 'left')
```

The analysis is only of complaints in 2005 or later.

```
In [42]: merged_comp_fac = merged_comp_fac[['abuse_number','facility_id','facility_typ
         e','facility_name','abuse_type','fine','action_notes','incident_date','outcom
         e','outcome_notes',
                                            'year','online_fac_name','public']][merged_
         comp_fac['year']>2004]
```

merged_comp_fac has all the complaints we need for the complaints analysis.

Aggregate data by facility

```
In [54]: complaint_pivot = merged_comp_fac.pivot_table(values='abuse_number',index='fac
         ility_id',columns='public', aggfunc='count').reset_index()
```

Next, left join the facilities to the pivot table.

```
In [56]: fac_pivot_merge = facilities.merge(complaint_pivot, how='left',on='facility_i
         d')
```

Add our own outcome code

In [57]:
```python
merged_comp_fac["omg_outcome"] = merged_comp_fac["outcome"].apply(dict([
    ('No Negative Outcome', 'Potential harm'),
    ('Exposed to Potential Harm', 'Potential harm'),

    ('Fall Without Injury', 'Fall, no injury'),

    ('Left facility without assistance without injury', 'Left facility without
attendant, no injury'),

    ('Loss of Dignity', 'Loss of Dignity'),

    ('Fall with Injury', 'Fracture or other injury'),
    ('Injury During Self-Transfer', 'Fracture or other injury'),
    ('Fall Resulting In Fractured Bone(s)', 'Fracture or other injury'),
    ('Fall Resulting In Fractured Hip', 'Fracture or other injury'),
    ('Transfer Resulting In Skin Injury or Bruise', 'Fracture or other injury'
),
    ('Fractured Bone', 'Fracture or other injury'),
    ('Fractured Hip', 'Fracture or other injury'),
    ('Burned', 'Fracture or other injury'),
    ('Transfer Resulting In Fractured Hip', 'Fracture or other injury'),
    ('Transfer Resulting In Fracture Bone(s)', 'Fracture or other injury'),
    ('Left Facility Without Assistance With Injury', 'Fracture or other injur
y'),
    ('Bruised', 'Fracture or other injury'),
    ('Skin Injury', 'Fracture or other injury'),

    ('Negative Behavior Escalated, Affected Other Resident(s)', 'Failure to ad
dress resident aggression'),

    ('Medical Condition Developed or Worsened', 'Medical condition developed o
r worsened'),
    ('Decubitus Ulcer(s) Developed', 'Medical condition developed or worsened'
),
    ('Decubitus Ulcer(s) Worsened', 'Medical condition developed or worsened'
),
    ('Urinary Tract Infection Worsened', 'Medical condition developed or worse
ned'),
    ('Transfer To Hospital For Treatment', 'Medical condition developed or wor
sened'),

    ('Received Incorrect or Wrong Dose of Medication(s)', 'Medication error'),
    ('The resident did not receive an ordered medication', 'Medication error'
),

    ('Loss of Resident Property', 'Loss of property, theft or financial exploi
tation'),
    ('Loss of Medication', 'Loss of property, theft or financial exploitation'
),
    ('Financially Exploited', 'Loss of property, theft or financial exploitati
on'),

    ('Unreasonable Discomfort', 'Unreasonable discomfort or continued pain'),
    ('Pain And Suffering Continued', 'Unreasonable discomfort or continued pai
n'),
```

```
        ('Undesirable Weight Loss', 'Weight loss'),

        ('Poor Continuity Of Care', 'Inadequate care'),
        ('Failed To Have Quality of Life Maintained or Enhanced', 'Inadequate car
e'),
        ('Failed to Receive Needed Services', 'Inadequate care'),
        ('Denied Choice In Treatment', 'Inadequate care'),

        ('Incontinence', 'Inadequate hygiene'),
        ('Inadequate Hygiene', 'Inadequate hygiene'),

        ('Physically Abused', 'Physical abuse'),
        ('Corporally Punished', 'Physical abuse'),

        ('Verbally Abused', 'Verbal or emotional abuse'),
        ('Mentally or Emotionally Abused', 'Verbal or emotional abuse'),

        ('Involuntarily Secluded', 'Involuntary seclusion'),

        ('Raped', 'Sexual abuse'),
        ('Sexually Abused', 'Sexual abuse'),

        ('Deceased', 'Death'),
        ('Facility was understaffed with no negative outcome', 'Staffing issues'),
        ('Unable to timely assess adequacy of staffing', 'Staffing issues'),

        ('Improperly Transferred Out of Facility, Denied Readmission or Inappropri
ate Move Within Facility', 'Denied readmission or moved improperly'),
]).get).fillna('')
```

Export the facility and complaints data for munging

```
In [58]: merged_comp_fac.to_csv('../../data/processed/complaints-3-29-scrape.csv',index
         =False)
```

```
In [59]: fac_pivot_merge.to_csv('../../data/processed/facilities-3-29-scrape.csv',index
         =False)
```

DONE