**Adding a new filter to faceted search**

1. Open file: …\puerto\_rico\filters\filter\_SQL\_queries.py

Add the desired SQL query filter function(s) – this is the hardest part

1. Open file: …\puerto\_rico\faceted\_search\faceted\_search\_filter\_classes.py

Add the desired filter on PID lists class using the abstract base class “PidFilter”:

**class** NewFilter(PidFilter):  
 \_\_metaclass\_\_ = abc.ABCMeta  
  
 **def** \_\_init\_\_(self, category):  
 super(NewFilter, self).\_\_init\_\_(**"FilterName"**)  
 self.desired\_category = category  
  
 **def** apply(self, pids\_list):  
 *"""Input: List of project IDs.  
 Output: Returns list of project IDs after filtering by this filter's category type."""* ans = []

**'''For every category of this filter, add if statement and call the respective SQL filter function'''**  
 **if** self.desired\_category == **"category1"**:  
 ans = sql\_fltr.respective\_sql\_fltr\_function(pids\_list)  
 **elif** self.desired\_category == **" category2"**:  
 ans = sql\_fltr. respective\_sql\_fltr\_function(pids\_list)

...  
 **return** ans  
  
 **def** get\_cat(self):  
 **return** str(self.desired\_category)  
  
 **def** get\_kind\_and\_cat(self):  
 **return** str(self.get\_kind() + **" "** + self.get\_cat())

1. Open file: …\puerto\_rico\faceted\_search\faceted\_search\_filter\_instances.py

Add instances of the new filter for each of its categories to the FilterSystem class’ \_\_init\_\_ function:

**def** \_\_init\_\_(self):  
 filter\_category1 = cl.NewFilter(**"category1"**)  
 filter\_category2 = cl.NewFilter(**"category2"**)

...

...  
  
 self.filters\_tuple = (filter\_category1, filter\_category2, ..., ...)   
 self.filters\_dict = {**"FilterName"**: (filter\_category1, filter\_category2, ...),

**"..."**: (...)}