U5VGKQ2SY : so essentially this function returns False when entry is in exclude, regardless of anything else that happens.
U2BS4M1RV : If include isn't an empty list then the entry is included if not in exclude. If include has entries and the entry matches one of the items in include then the entry is included if not in exclude.
U2BS4M1RV : So, if whitelist then only include entry if it is in whilelist (and also not in blacklist).
If only blacklist include everything not in blacklist.

U5VGKQ2SY : brb
U2BS4M1RV : The first if is necessary as well, because without it the code would see if entry is in an empty list, which it never will be and the lack of whitelist means nothing gets through.
U2BS4M1RV : It might be better if I changed those variables names to whiltelist and blacklist.
U5VGKQ2SY : <@U2BS4M1RV> Check this out:
```
def _query_filter(entry: str, include: list = None, exclude: list = None):
    if include:
        if entry in include and entry not in exclude:
            return True
    else:
        if entry not in exclude:
            return True
    return False

print("Entry: 'test', include is ['this', 'is', 'a', 'test'], exclude is ['whatever']: "
      + str(_query_filter('test', ['this', 'is', 'a', 'test'], ['whatever'])))
print("Entry: 'test', include is [], exclude is ['whatever']: "
      + str(_query_filter('test', [], ['whatever'])))
```
output:
```
Entry: 'test', include is ['this', 'is', 'a', 'test'], exclude is ['whatever']: True
Entry: 'test', include is [], exclude is ['whatever']: True
```

U5VGKQ2SY : `if include` has no value b/c the very next line you are checking for the exact same thing that the `else` checks
U5VGKQ2SY : as it is written, `_query_filter()` returns True if `entry` is NOT in `exclude` and False for anything else
U2BS4M1RV : I think you misunderstand the meaning of include.
U5VGKQ2SY :
```
def _query_filter(entry: str, include: list = None, exclude: list = None):
    if include:
        if entry in include and entry not in exclude:
            return True
    else:
        if entry not in exclude:
            return True
    return False

print("Entry: 'test', include is ['this', 'is', 'a', 'test'], exclude is ['whatever']: "
      + str(_query_filter('test', ['this', 'is', 'a', 'test'], ['whatever'])))
print("Entry: 'test', include is [], exclude is ['whatever']: "
      + str(_query_filter('test', [], ['whatever'])))
print("Entry: 'test', include is ['this', 'is', 'a', 'test'], exclude is ['whatever', 'test']: "
      + str(_query_filter('test', ['this', 'is', 'a', 'test'], ['whatever', 'test'])))
print("Entry: 'test', include is [], exclude is ['whatever', 'test']: "
      + str(_query_filter('test', [], ['whatever', 'test'])))
```
output:
```
Entry: 'test', include is ['this', 'is', 'a', 'test'], exclude is ['whatever']: True
Entry: 'test', include is [], exclude is ['whatever']: True
Entry: 'test', include is ['this', 'is', 'a', 'test'], exclude is ['whatever', 'test']: False
Entry: 'test', include is [], exclude is ['whatever', 'test']: False
```

```

U2BS4M1RV : Here is one set of my tests for it. The first test shows what the raw data is.
```
    def test_counts_query_unfiltered(self):
        counts = pypihole.counts_query(self.test_log)
        self.assertEqual(counts['unifi'], 2)
        self.assertEqual(counts['openvpn'], 2)
        self.assertEqual(counts['<http://docs.google.com|docs.google.com>'], 1)

    def test_counts_query_include(self):
        counts = pypihole.counts_query(self.test_log, ['unifi'])
        self.assertEqual(counts['unifi'], 2)
        self.assertEqual(counts['openvpn'], 0)

    def test_counts_query_exclude(self):
        counts = pypihole.counts_query(self.test_log, exclude=['openvpn'])
        self.assertEqual(counts['unifi'], 2)
        self.assertEqual(counts['openvpn'], 0)
        self.assertEqual(counts['<http://docs.google.com|docs.google.com>'], 1)

    def test_counts_query_include_and_exclude(self):
        counts = pypihole.counts_query(self.test_log,
                                       include=['unifi', '<http://docs.google.com|docs.google.com>'],
                                       exclude=['openvpn'])
        self.assertEqual(counts['unifi'], 2)
        self.assertEqual(counts['openvpn'], 0)
        self.assertEqual(counts['<http://docs.google.com|docs.google.com>'], 1)
```

U2BS4M1RV : All of the tests pass as it is currently.
U2BS4M1RV : Include is a whitelist, but only used if it isn't an empty list. If include is an empty list then it is ignored.
U2BS4M1RV : Your example is the intended result.
U2BS4M1RV : It works similarly to the include used on some routers, or like grep in a way. If include isn't specified everything is returned True. If include is populated then it acts as a whiltelist. And exclude works as a blacklist.
U5VGKQ2SY : output:
```
'Entry: 'test' NOT IN exclude:
Entry: 'test', include is ['this', 'is', 'a', 'test'], exclude is ['whatever']: True
Entry: 'test', include is [], exclude is ['whatever']: True
Entry: 'test', include is ['this', 'is', 'a', 'test'], exclude is ['whatever']: True
Entry: 'test', include is [], exclude is ['whatever']: True
Entry: 'test' IN exclude:
Entry: 'test', include is ['this', 'is', 'a', 'test'], exclude is ['whatever', 'test']: False
Entry: 'test', include is [], exclude is ['whatever', 'test']: False
Entry: 'test', include is ['this', 'is', 'a', 'test'], exclude is ['whatever', 'test']: False
Entry: 'test', include is [], exclude is ['whatever', 'test']: False
```

U5VGKQ2SY : &gt; If include isn't specified everything is returned TrueThat's not what is happening.

U2BS4M1RV : Sorry, that should say if include isn't specified, and exclude doesn't trigger, then it is returned True.
U5VGKQ2SY : but I could be wrong. Probably best to wait for someone else to chime in
U2BS4M1RV : There is no problem with that code, it tests perfectly.
```