**22 july**

Changed one club name as per hive.

Updated every file.

15 July 2025

Re-ran with updated club data.

# Comparing club positions

With and without 2019 filter. Without 2019 filter seemed to be most accurate.

Code standardized club names (lowercase, remove whitespace etc) from both athena and campusgroups.

Then, created a club map—with Kiersten’s help was able to map everything correctly. Clubs that are only in athena and not campusgroup are NaN—which shouldn’t matter as clubs in both are mapped and we only need the map to merge—so if not in one for sure—merge not affected.

Merged and got only\_in\_athena, only\_in\_campusgroup and in\_both.

Final dataset: club\_map.xlsx, club\_membership\_exclusive 22 july updated club.xlsx

**22 july**

only\_in\_athena = 130048

only\_in\_campusgroups= 78549

in\_both = 5207

15 July 2025

Re-ran with updated club data.

Final dataset: club\_membership\_exclusive 15 july updated club.xlsx

without 2019 filter & updated 15 july club mapping

only\_in\_athena = 130079

only\_in\_campusgroups= 78579

in\_both = 5176

# Comparing officer positions:

Ran with and without 2019 filter.

Created table with list of all positions in athena and campusgroups. Saw that in athena, “Member” and nans were to be marked FALSE for officer flag.

athena\_filtd['officer\_flag'] = (

(athena\_filtd['Student Activity Participation'] != 'Member') &

(~pd.isna(athena\_filtd['Student Activity Participation']))

).astype(str)

In campusgroups, only blanks/nans were to be marked as FALSE for officer flag.

campusgroups\_filtd['officer\_flag'] = pd.notna(campusgroups\_filtd['officerPosition']).astype(str)

grouping and creating concatenated positions column:

campusgroup\_grouped = campusgroups\_filtd.groupby(['UNI', 'common\_mapped\_name']).agg({

'officer\_flag': 'max', # if they were an officer at least once → 1

'officerPosition': lambda x: ', '.join(sorted(set(x.dropna())))

}).reset\_index()

athena\_grouped = athena\_filtd.groupby(['UNI', 'common\_mapped\_name']).agg({

'officer\_flag': 'max',

'Student Activity Participation': lambda x: ', '.join(

sorted(set(str(val) for val in x if str(val) != 'Member' and pd.notna(val)))

)

}).reset\_index()

Created officer\_flag for both datasets. campusgroup already has an officer flag but it doesn't seem to be entirely correct.

Above, officer\_position and student\_Activity\_participation are getting rewritten with all positions that that student held in that club separated by a comma.

'''

No 2019 filter

only in athena(only BU)= 76396

only in campusgroups = 77824

in\_both=5079

'''

'''

with 2019 filter

only\_athena (only BU)= 191

only\_campusgroups= 81453

in\_both = 1378

'''

Final datasets: with 2019 filter club\_officer\_exclusive.xlsx, club\_officer\_exclusive no year filter.xlsx

15 July 2025:

* Changed officer\_flag for campus group:
* campusgroups\_filtd['officer\_flag'] = (campusgroups\_filtd['officer']==1).astype(str)
* rerunning with updated clubs.
* Updated:

def aggregate\_position(pos, custom, role):

# Combine all three into one Series

combined = pd.concat([pos, custom, role]).dropna().astype(str)

combined = combined[combined.str.strip() != ''] # Remove blanks

if combined.empty:

return 'Unknown'

else:

return ', '.join(sorted(set(combined)))

campusgroup\_grouped = campusgroups\_filtd.groupby(['UNI', 'common\_mapped\_name']).agg({

'officer\_flag': 'max',

'officerPosition': lambda x: list(x),

'officerCustomPosition': lambda x: list(x),

'officerRole': lambda x: list(x)

}).reset\_index()

# Apply the aggregation logic row-wise

campusgroup\_grouped['aggregated\_officer\_position'] = campusgroup\_grouped.apply(

lambda row: aggregate\_position(

pd.Series(row['officerPosition']),

pd.Series(row['officerCustomPosition']),

pd.Series(row['officerRole'])

) if row['officer\_flag'] == 1 else '',

axis=1

)

Final dataset: club\_officer\_exclusive updated 22 july.xlsx

22 July

only in athena(only BU)= 76534

only in campusgroups = 77773

in\_both= 5130

15 july

No 2019 filter, updated club map, updated officer flag for campusgroup.

only in athena(only BU)= 76565

only in campusgroups = 77803

in\_both= 5100

# Club final dataset:

Used the ‘merged dataset from comparing officer position.py--- since this already ahd what we needed. Wrote it to csv.

Created two new columns

1. ‘merged\_final\_flag’ which was TRUE if either of the athena officer flag or campusgroup flag were true (=OR(D2,F2)).
2. Merged\_final\_position which merged the position concated columns from athena and campusgroup if they weren’t blank (=TEXTJOIN(", ", TRUE, E2, G2)) (Note that if there were multiple positions that that student held in the same club—it was already merged in the python code.)

Final dataset: final club dataset merged.csv

Left\_only= athena only. Right\_only=campusgroup only.

15 july (officer updates and club map update)

Used python to created final\_aggregated position. – manually removed “Unknown, “ and “, Unknown”.

Manually mapped final officer flag (=OR(D2,F2) or something).

Final dataset: final club dataset merged 22 july.csv