D:\NHRI\AMR\amr-main\explore\top6_pct_by_sr.py

```python
import pandas as pd
import numpy as np
import seaborn as sns
from pathlib import Path
from db_tools import ezfuncs
from cod_prep.downloaders import *
from cod_prep.utils import *
from cod_prep.claude.configurator import Configurator
from amr_prep.utils.amr_io import get_amr_results
from amr_prep.utils.amr_io import AmrResult
from mcod_prep.compile_burden import summarize_draws
from explore.number_plugging.rtr import *
import warnings
warnings.filterwarnings("ignore")

CONF = Configurator()
LSV_ID = CONF.get_id('location_set_version')
CSV_ID = CONF.get_id('cause_set_version')

lh = get_current_location_hierarchy(location_set_version_id=LSV_ID)
ch = get_current_cause_hierarchy(cause_set_version_id=CSV_ID)

print('Find top 6 pathogen percent against total by super region for both counterfactuals')
supregs = lh.loc[lh['level'] == 1, ['location_id', 'location_name']].drop_duplicates()
supregs_ids = supregs['location_id'].tolist()
supregs_dict = dict(zip(supregs.location_id, supregs.location_name))

df = get_results_wrapper('fatal', 1, 1, syndrome='all', abx_class='all_resistant',
    counterfactual='no_infection')
top6 = df.loc[(df['pathogen'] != 'all'), ]
    .sort_values(by='amr_mean', ascending=False).head(6)['pathogen'].unique().tolist()

df_f = get_results_wrapper('fatal', 1, 1, location_id=supregs_ids,
    pathogen=top6, syndrome='all', abx_class='all_resistant', draws=True)
total_f = get_results_wrapper('fatal', 1, 1, location_id=supregs_ids,
    syndrome='all', pathogen='all', abx_class='all_resistant', draws=True)

df_nf = get_results_wrapper('nonfatal', 2, 1, location_id=supregs_ids,
    pathogen=top6, syndrome='all', abx_class='all_resistant', draws=True)
total_nf = get_results_wrapper('nonfatal', 2, 1, location_id=supregs_ids,
    syndrome='all', pathogen='all', abx_class='all_resistant', draws=True)

dffs = []
dfnfs = []
for pathogen in top6:
    print("-------" + pathogen.capitalize() + "--------")
    for location_id, location_name in supregs_dict.items():
        print("----" + location_name + '-----')
        print('Deaths')
        df_f1 = df_f.loc[(df_f['pathogen'] == pathogen) & (df_f['location_id'] == location_id), ]
        total_f1 = total_f.loc[(total_f['location_id'] == location_id), ]
        df_f1, pct_f = aggregate_summarize_draws(df_f1, to_aggregate='pathogen', get_pct=True, denominator=total_f1)
        print(pathogen + ' attributable AMR death percentage out of total in :' + location_name)
        print_out_cf_lower_mean_upper(pct_f, 'deaths')
        print()
        pct_f['pathogen'] = pathogen
        dffs.append(pct_f)

        print('DALYS')
        df_nf1 = df_nf.loc[(df_nf['pathogen'] == pathogen) & (df_nf['location_id'] == location_id)]
        total_nf1 = total_nf.loc[(total_nf['location_id'] == location_id), ]
        df_nf1, pct_nf = aggregate_summarize_draws(df_nf1, to_aggregate='pathogen', get_pct=True, denominator=total_nf1)
        print(pathogen + ' attributable AMR DALYs percentage out of total in :' + location_name)
        print_out_cf_lower_mean_upper(pct_nf, 'DALYs')
        pct_nf['pathogen'] = pathogen
        dfnfs.append(pct_nf)

    print()

dff = pd.concat(dffs)
dfnf = pd.concat(dfnfs)

dff.to_csv('FILEPATH', index=False)
dfnf.to_csv('FILEPATH', index=False)
```