```
from models import db, User, Query, Story
from sent_analysis import subjectize, polarize, parse_async
from flask import session
from sqlalchemy import exc
from psycopg2.errors import UniqueViolation


"""Helper Functions"""


def order_pol():
    """Loops over session results, filters out stories with no SA results,
    then orders by polarity"""
    results = parse_async(session['results'])
    for story in results:
        score = polarize(story, parsed=True)
        if not score:
            story['pol'] = None
        else:
            story['pol'] = score['article_res']['result']
    session['results'] = [story for story in results if story['pol']]
    results = session['results']
    not_negative = [
        story for story in results if story['pol'][0] is not "-"]
    negative = [
        story for story in results if story['pol'][0] is "-"]
    ordered_neg = sorted(negative,
                         key=lambda story: story['pol'])
    ordered_not_neg = sorted(not_negative,
                             key=lambda story: story['pol'], reverse=True)
    ordered = ordered_not_neg + ordered_neg
```

```
    return ordered



def order_sub():
    """Loops over session results, filters out stories with no SA results,
    then orders by subjectivity"""
    results = parse_async(session['results'])
    for story in results:
        score = subjectize(story, parsed=True)
        if not score:
            story['sub'] = None
        else:
            story['sub'] = str(score['measure'])
    session['results'] = [story for story in results if story['sub']]
    results = session['results']
    ordered = sorted(results,
                     key=lambda story: story['sub'], reverse=True)
    return ordered


"""Conversions form SqlAlchemy object to dictionary"""
def db_query_to_dict(query):
    """Converts Query SqlAlchemy object to dict"""
    dict = {}
    dict['name'] = query.name
    dict['id'] = query.id
    dict['user_id'] = query.user_id
    dict['keyword'] = query.keyword
    dict['source'] = query.source
    dict['quantity'] = query.quantity
```

```
    dict['date_from'] = query.date_from
    dict['date_to'] = query.date_to
    dict['language'] = query.language
    dict['sa'] = query.sa
    dict['sort_by'] = query.sort_by
    if "query" in session:
        session.pop("query")
    session["query"] = dict
    return dict


def db_story_to_dict(story):
    """Converts Story SqlAlchemy object to dict"""
    dict = {}
    dict['id'] = story.id
    dict['user_id'] = story.user_id
    dict['headline'] = story.headline
    dict['source'] = story.source
    dict['content'] = story.content
    dict['author'] = story.author
    dict['description'] = story.description
    dict['url'] = story.url
    dict['image'] = story.image
    dict['published_at'] = story.published_at
    return dict


def db_user_to_dict(user):
    """Converts User SqlAlchemy object to dict"""
    dict = {}
    dict['id'] = user.id
```

```
    dict['username'] = user.username
    dict['password'] = user.password
    dict['email'] = user.email
    dict['first_name'] = user.first_name
    dict['last_name'] = user.last_name
    return dict

"""Conversions from dictionary to SqlAlchemy object"""

def dict_query_to_db(user_id, dict):
    """Adds saved query to associated user in db"""
    if dict['sort_by'] == "subjectivity" or dict['sort_by'] == "polarity":
        dict['sa'] = dict['sort_by']
        dict['sort_by'] = 'relevancy'
    dict['type'] = "Detailed Search"
    query = Query(user_id = user_id,
        name = dict['name'],
        source = dict['source'],
        quantity = dict['quantity'],
        date_from = dict['date_from'],
        date_to = dict['date_to'],
        language = dict['language'],
        sort_by = dict['sort_by'],
        sa = dict['sa'],
        type = dict['type']
        )

    return query
```

```
def dict_story_to_db(user_id, dict):
    """Adds saved story to associated user in db"""
    story = Story(user_id = user_id,
    headline = dict['headline'],
    source = dict['source'],
    content = dict['content'],
    author = dict['author'],
    description = dict['description'],
    url = dict['url'],
    image = dict['image'],
    published_at = dict['published_at'],
    sub = dict['sub'],
    pol = dict['pol']
    )
    return story


"""Conversions from form data to dict"""
def form_query_to_dict(form):
    """Converts form data to dict"""
    query = {}
    query['keyword'] = form.keyword.data
    query['source'] = form.source.data
    query['quantity'] = form.quantity.data
    query['date_from'] = form.date_from.data
    query['date_to'] = form.date_to.data
    query['language'] = form.language.data
    if form.sort_by.data == "subjectivity" or form.sort_by.data == "polarity":
        query['sa'] = form.sort_by.data
        query['sort_by'] = 'relevancy'
```

```
else:
    query['sort_by'] = form.sort_by.data
    query['sa'] = None

if form.default.data:
    query['name'] = form.name.data
    query['default'] = True
elif form.saved_query.data:
    query['name'] = form.name.data
session['query'] = query
return query
```