The Basic SQL Queries of SQLAlchemy

WE'LL COVER THE FOLLOWING

- Wrapping Up

SQLAlchemy provides all the queries you'll probably ever need. We'll be spending a little time just looking at a few of the basic ones though, such as a couple simple SELECTs, a JOINed SELECT and using the LIKE query. You'll also learn where to go for information on other types of queries. For now, let's look at some code:

```
# queries.py
from sqlalchemy import create_engine
from sqlalchemy.orm import sessionmaker
# from table_def import Album, Artist

engine = create_engine('sqlite:///mymusic.db', echo=True)

# create a Session
Session = sessionmaker(bind=engine)
session = Session()

# how to do a SELECT * (i.e. all)
res = session.query(Artist).all()
for artist in res:
    print(artist.name)

# how to SELECT the first result
res = session.query(Artist).filter(Artist.name=="Newsboys").first()

# how to sort the results (ORDER_BY)
res = session.query(Album).order_by(Album.title).all()
for album in res:
    print(album.title)

# how to do a JOINed query
qry = session.query(Artist, Album)
qry = qry.filter(Artist.id==Album.artist_id)
artist = qry.filter(Album.title=="Step Up to the Microphone").first()
album = qry.filter(Album.title=="Step Up to the Microphone").first()

# how to use LIKE in a query
res = session.query(Album).filter(Album.publisher.like("S%a%")).all()
res = session.query(Album).filter(Album.publisher.like("S%a%")).all()
for item in res:
    print(item.publisher)
```

The first query we run will grab all the artists in the database (a SELECT *) and print out each of their name fields. Next you'll see how to just do a query for a specific artist and return just the first result. The third query shows how do a SELECT* on the Album table and order the results by album title. The fourth query is the same query (a query on a JOIN) we used in our editing section except that we've broken it down to better fit PEP8 standards regarding line length. Another reason to break down long queries is that they become more readable and easier to fix later on if you messed something up. The last query uses LIKE, which allows us to pattern match or look for something that's "like" a specified string. In this case, we wanted to find any records that had a Publisher that started with a capital "S", some character, an "a" and then anything else. So this will match the publishers Sparrow and Star, for example.

SQLAlchemy also supports IN, IS NULL, NOT, AND, OR and all the other filtering keywords that most DBAs use. SQLAlchemy also supports literal SQL, scalars, etc, etc.

Wrapping Up

At this point you should know SQLAlchemy well enough to get started using it confidently. The project also has excellent documentation that you should be able to use to answer just about anything you need to know. If you get stuck, the SQLAlchemy users group / mailing list is very responsive to new users and even the main developers are there to help you figure things out.