Skills SQLAlchemy

Setup

Make a virtual environment, and install the requirements.

```
$ virtualenv env
$ source env/bin/activate
$ pip install -r requirements.txt
```

Reminder: in Windows, use the command 'virtualenv env –always-copy' to create a virtual environment.

```
$ createdb cars
$ psql cars < database.sql
```

Part 1: Fill in *model.py*

Using SQLAlchemy, fill in the columns for the classes already defined in ***model.py***. The column names and datatypes should be the same as those in the ***cars*** database. (Since you already have tables, there is no need to run the command `db.create_all()` at any point during this assignment.)

Be sure to include a **relationship** between the two tables, using a foreign keys between the two tables.

Helper commands

- To open the database: `psql cars`
- To see a list of the tables: `\dt` (must be inside ***psql***)
- To inspect the schema for each table: `\d TABLENAME` (must be inside ***psql***)

Hint

You will know if you have the right answer if you can run the interactive python terminal in ***model.py*** and the only output you see is `Connected to DB`, i.e.

```
$ python -i model.py
Connected to DB.
>>>
```

If you have the incorrect answer, you'll see something like this:

```
$ python -i model.py
...
sqlalchemy.exc.ArgumentError: Mapper Mapper|Model|models could
not assemble any primary key columns for mapped table 'models'
```

Part 2: SQLAlchemy Queries

Please compose the following queries, using an interactive python terminal to test them as you go along. Add your SQLAlchemy queries to the file called ***query.py***.

1. Get the brand with the `id` of 8.
   - Use SQLAlchemy's `.get()`, not `.filter()` nor `.filter_by()`.
2. Get all models with the name Corvette and the **brand_name** Chevrolet.
3. Get all models that are older than 1960.
4. Get all brands that were founded after 1920.
5. Get all models with names that begin with "Cor."
6. Get all brands that were founded in 1903 and that are not yet discontinued.
7. Get all brands that are either 1) discontinued (at any time) or 2) founded before 1950.
8. Get any model whose brand_name is not Chevrolet.

Fill in the stubbed out functions found in ***query.py***. Important: You may only run a single query in each function.

1. Fill in `get_model_info` so that it takes a year as input, and *prints* each model's `name`, `brand_name` and brand `headquarters` for each car model from that year.
2. Fill in `get_brands_summary` so that it takes nothing as input and prints each brand name, and all of that brand's models. (Feel free to format with newlines (`\n`) and/or tabs (`\t`) to create helpful and readable output.)

Helper commands

- To open an interactive terminal in order to test queries: `python -i model.py`

Part 2.5: Discussion questions

Please add your answers as comments in ***query.py***.

1. What is the returned value and datatype of `Brand.query.filter_by(name='Ford')`?

2. In your own words, what is an association table, and what *type* of relationship (many to one, many to many, one to one, etc.) does an association table manage?

Part 3

Please compose the following python functions and add them to ***query.py***.

1. Design a function in python that takes in any string as parameter, and returns *a list of objects* that are brands whose name *contains or is equal to* the input string.
2. Design a function that takes in a start year and end year (two integers), and returns a **list of objects** that are models with years that fall between the start year (inclusive) and end year (exclusive).