Queries: filtering

Often times you don't want to retrieve all the entries from a table but select only those that satisfy some criterion. Criteria are usually based on the values of the table's columns. To filter a query, SQLAlchemy provides the `.filter()` method.

For example, to select books from a specific year from the `Book` table we use the following command:

```
Book.query.filter(Book.year = 2020).all()
```

Notice the additional `.all()` method. `.filter()` returns a `Query` object that needs to be further refined. This can be done by using several additional methods like `.all()` that returns a list of all results, `.count()` that counts the number of fetched entries, or `.first()` that returns only one result, namely the first one.

```
Book.query.filter(Book.year = 2020).first()
```

Multiple criteria may be specified as comma separated and the interpretation of a comma is a Boolean `and`:

```
Review.query.filter(Review.stars <= 3, Review.book_id = 1).all()
```

This query will return all entries in the `Review` table that have fewer than 3 stars for the book with `id = 1`.

Note: there is also the `.filter_by()` method that uses only a simple attribute-value test for filtering.

Instructions

1.
Use the `filter` method to fetch **all** the readers from the `Reader` table with `Adams` surname, and assign the result in the variable called `adams`.

Stuck? Get a hint

2.
Use the `filter` method to fetch all the books from the year 2019 or earlier, but then assign only the **first** result to the variable called `book_pre2019`.

```
from app import Book, Reader, Review, Annotation

#select books from the year 2020
book_2020 = Book.query.filter(Book.year == 2020).all()
print("All the suggested books in the year 2020:")
[print(book) for book in book_2020]
```

```
#instead of all books suggested in 2020, fetch only the first one
book_2020_first = Book.query.filter(Book.year == 2020).first()
print("\nThe first book fetched from the year 2020: ", book_2020_first)

#you can specify multiple criteria for filtering
rev_3_boook13 = Review.query.filter(Review.stars <= 3, Review.book_id ==
13).all()
print("\nThe review of 3 stars or lower written for a book with id = 13: ",
rev_3_boook13)

#Checkpoint 1: fetching all the readers with "Adams" surname
adams = Reader.query.filter(Reader.surname == "Adams").all()
[print(person) for person in adams]

#Checkpoint 2: fetching the first book dating prior to the year 2019
book_pre2019 = Book.query.filter(Book.year <= 2019).all()
print(book_pre2019)
```