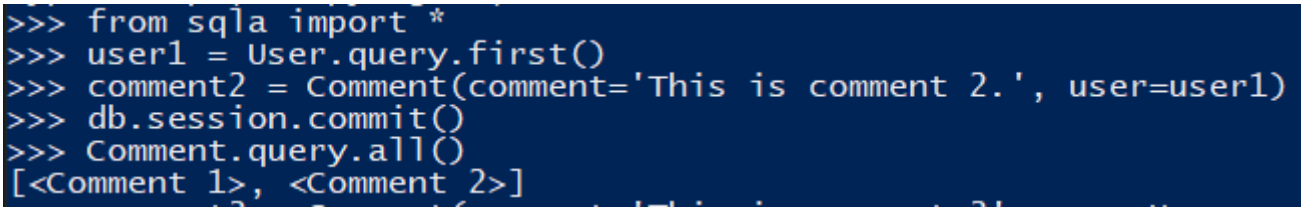
Flask-SQLAlchemy

1. SQLAlchemy number of rows in a query
2. Difference between filter and filter_by in SQLAlchemy
3. How to delete rows from a table using an SQLAlchemy query without ORM?
4. Cascade delete not working depending on how deletion is made #7974
5. Foreign Key Properties ON UPDATE and ON DELETE

Session:

Suppose you have two tables (User, Comment) linked by a one-to-many relationship. Suppose you query and get a user (say user1). You then create a comment (say, comment1) and set its user as user1. Before commiting, we don't need to add comment1 to the session because user1 is already in the session via the relationship.



Q: SQLAlchemy - Is there a way to see what is currently in the session?

Getting objects in a session:

- `session.new` - objects that will be added to the database
- `session.dirty` - objects which will be updated
- `session.deleted` - objects which will be deleted from the database

```
>>> user1 = User.query.first()
>>> db.session.new
IdentitySet([])
>>> db.session.dirty
IdentitySet([])
>>> db.session.deleted
IdentitySet([])
>>> comment4 = Comment(comment='This is the fourth comment', user=user1)
>>> db.session.new
IdentitySet([<Comment (transient 2536242230800)>])
>>> db.session.dirty
IdentitySet([<User ramesh>])
>>> db.session.deleted
IdentitySet([])
>>> db.session.commit()
>>> db.session.new
IdentitySet([])
>>> db.session.dirty
IdentitySet([])
>>> db.session.deleted
IdentitySet([])
>>> _
```

Lazy Parameter

```python
class User(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(80), unique=True, nullable=False)
    email = db.Column(db.String(120), unique=True, nullable=False)
    comments = db.relationship('Comment', backref='user')

    def __repr__(self):
        return f'<User {self.username}>'

class Comment(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    comment = db.Column(db.Text)
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'), nullable=False)
```

- lazy = select (or True) → default
- lazy = dynamic
- lazy = joined (or False)
- lazy = subquery

When ***querying through relationships***, the lazy parameter determines how related objects are loaded. By default it is `select` or `True`.

`select` returns the objects directly while `dynamic` returns a sqlalchemy object, upon which additional methods like `all()` or `first()` needs to be added to get the objects.

`joined` and `subquery` do the same thing, they return the joined tables, but the method of joining is different which may result in performance

differences during execution.

lazy = select

```
>>> User.query.get(1).comments
[<Comment 1>, <Comment 2>, <Comment 4>]
>>>
```

lazy = dynamic

```
>>> Comment.query.filter_by(user_id=1)
<flask_sqlalchemy.BaseQuery object at 0x0000024E83DAF160>
>>> Comment.query.filter_by(user_id=1).all()
[<Comment 1>, <Comment 2>, <Comment 4>]
```

article

video