Update & Delete

In this lesson, we will discover how to update and delete data from the database.

WE'LL COVER THE FOLLOWING

Update and delete are also basic operations that are performed on a database. So, let's find out how we perform these operations using the `SQLAlchemy` **ORM**.

Update

The method for updating a record is very straightforward, and you do not have to remember any new commands. It can be achieved in three easy steps.

1. Retrieve the object to update

First, to update any record that is already present in the database, we must first retrieve it. We have already learned how to do the retrieval in the last lesson. So let us retrieve one record with respect to its `primary_key`.

```
user = User.query.get("veronica.lodge@email.com")
```

2. Update the object

Afterward, we can update the value inside the retrieved object.

```
user.email = "veronica@email.com"
```

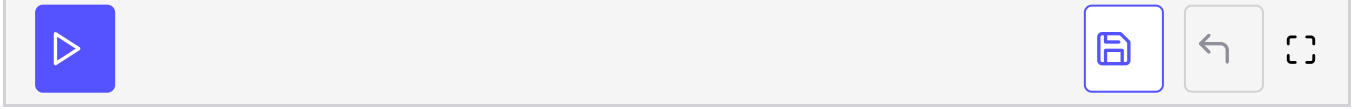
3. Commit the changes in `session`

Now, we `commit` these changes so that they become persistent in the database.

```
db.session.commit()
```

Complete example for update

```
from flask import Flask, render_template
from flask_sqlalchemy import SQLAlchemy

app = Flask(__name__)
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///example.db'
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
db = SQLAlchemy(app)

class User(db.Model):
    email = db.Column(db.String, primary_key=True, unique=True, nullable=False)
    password = db.Column(db.String, nullable=False)

db.create_all()
db.session.add(User(email = "archie.andrews@email.com", password = "football4life"))
db.session.add(User(email = "veronica.lodge@email.com", password = "fashiondiva"))

try:
    db.session.commit()
except Excetion as e:
    db.session.rollback()

user = User.query.get("veronica.lodge@email.com")
print(user)

user.email = "veronica@email.com"

try:
    db.session.commit()
except Excetion as e:
    db.session.rollback()

print("All Users : ", User.query.all())
```



Deletion

The deletion operation is also straightforward. The steps to delete an object from the database are given below.

1. Retrieve the object to delete

Similar to the insertion process, we first need to retrieve the object we want to delete.

```
user = User.query.get("veronica.lodge@email.com")
```

2. Delete using `session.delete()`

We can then delete this object from the session by passing it as an argument to the `session.delete()` function.

```
db.session.delete(user)
```

3. Commit the changes in the `session`

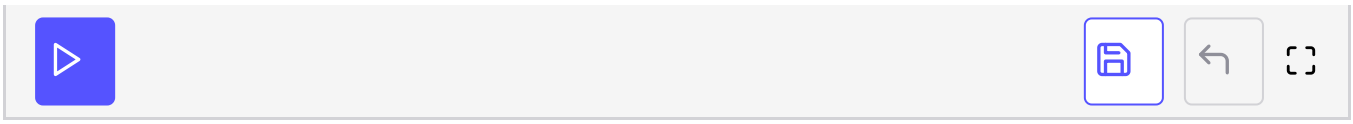
Last, to persist the changes in the database, we `commit()` these changes to the database.

```
db.session.commit()
```

Complete example for delete

```
from flask import Flask, render_template
from flask_sqlalchemy import SQLAlchemy

app = Flask(__name__)
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///example.db'
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
db = SQLAlchemy(app)

class User(db.Model):
    email = db.Column(db.String, primary_key=True, unique=True, nullable=False)
    password = db.Column(db.String, nullable=False)

db.create_all()
db.session.add(User(email = "archie.andrews@email.com", password = "football4life"))
db.session.add(User(email = "veronica.lodge@email.com", password = "fashiondiva"))

try:
    db.session.commit()
except Exception as e:
```

```
    db.session.rollback()

user = User.query.get("veronica.lodge@email.com")
print(user)

db.session.delete(user)

try:
    db.session.commit()
except Excetion as e:
    db.session.rollback()

print("All Users : ", User.query.all())
```



In the next lesson, we will be solving yet another challenge!