Adding Examples for One-to-One, One-to-Many, and Many-to-Many Relationships in SQLAlchemy

In this section, we'll extend the previous exercise by demonstrating how to implement **one-to-one**, **one-to-many**, and **many-to-many** relationships using SQLAlchemy.

Step 10: One-to-One Relationship

Let's create an example where **one student** has **one address** (one-to-one relationship).

1. Define the Models

```
python
Copy code
from sqlalchemy import ForeignKey
from sqlalchemy.orm import relationship

class Address(Base):
    __tablename__ = 'addresses'

    id = Column(Integer, primary_key=True)
    student_id = Column(Integer, ForeignKey('students.id'))
    street = Column(String)
    city = Column(String)

    student = relationship("Student", back_populates="address")

class Student(Base):
    __tablename__ = 'students'

    id = Column(Integer, primary_key=True)
    name = Column(String)
    age = Column(Integer)
    grade = Column(String)

    # One-to-One relationship
    address = relationship("Address", uselist=False,
back_populates="student")
```

- Here, a `Student` has one `Address`, and each `Address` is linked to one `Student`.
- `uselist=False` ensures this is a **one-to-one** relationship.

2. Creating and Associating Data

```
python
Copy code
# Add a student and their address
student = Student(name='Charlie', age=25, grade='B')
address = Address(street='123 Main St', city='New York', student=student)

session.add(student)
session.add(address)
session.commit()
```

Step 11: One-to-Many Relationship

Now, let's create an example where **one student** can have **multiple courses** (one-to-many relationship).

1. Define the Models

```
python
Copy code
class Course(Base):
    __tablename__ = 'courses'

    id = Column(Integer, primary_key=True)
    student_id = Column(Integer, ForeignKey('students.id'))
    course_name = Column(String)

    student = relationship("Student", back_populates="courses")

class Student(Base):
    __tablename__ = 'students'

    id = Column(Integer, primary_key=True)
    name = Column(String)
    age = Column(Integer)
    grade = Column(String)

    # One-to-Many relationship
    courses = relationship("Course", back_populates="student")
```

- In this case, one `Student` can enroll in many `Course`s, while each `Course` belongs to one `Student`.

2. Creating and Associating Data

```
python
Copy code
# Add a student with multiple courses
student = Student(name='Diana', age=22, grade='A')
course1 = Course(course_name='Math', student=student)
course2 = Course(course_name='Physics', student=student)

session.add(student)
session.add(course1)
session.add(course2)
session.commit()
```

Step 12: Many-to-Many Relationship

Now, we'll create an example where **many students** can enroll in **many courses** (many-to-many relationship).

1. Define the Models

To implement a many-to-many relationship, you need an **association table**.

```
python
Copy code
from sqlalchemy import Table

# Association table for the many-to-many relationship
student_course_association = Table('student_course', Base.metadata,
    Column('student_id', Integer, ForeignKey('students.id')),
    Column('course_id', Integer, ForeignKey('courses.id'))
)

class Course(Base):
    __tablename__ = 'courses'

    id = Column(Integer, primary_key=True)
    course_name = Column(String)

    # Many-to-Many relationship
    students = relationship("Student",
secondary=student_course_association, back_populates="courses")

class Student(Base):
    __tablename__ = 'students'

    id = Column(Integer, primary_key=True)
    name = Column(String)
    age = Column(Integer)
    grade = Column(String)

    # Many-to-Many relationship
    courses = relationship("Course", secondary=student_course_association,
back_populates="students")
```

- The `student_course_association` table is an **association table** that links `Student` and `Course` in a many-to-many relationship.

2. Creating and Associating Data

```
python
Copy code
# Create some students and courses
student1 = Student(name='Eve', age=21, grade='A')
student2 = Student(name='Frank', age=23, grade='B')

course1 = Course(course_name='Math')
course2 = Course(course_name='Physics')

# Many-to-Many relationship: enrolling students in courses
student1.courses.append(course1)
student1.courses.append(course2)
student2.courses.append(course1)

session.add(student1)
session.add(student2)
session.commit()
```