

Introducción

A través del ORM de SQLAlchemy se puede crear las relaciones entre clases, de la misma forma que se hace cuando se diseña e implementa una base de datos con lenguaje SQL:

- uno a uno
- uno a muchos
- muchos a muchos

Relaciones de entidades con ORM SQLAlchemy Ejemplo

<https://github.com/taw-desarrollo-plataformas-web/EjemploSqlAlchemy>

Generación de Tablas o Entidades

```
17 class Club(Base):
18     __tablename__ = 'club'
19     id = Column(Integer, primary_key=True)
20     nombre = Column(String)
21     deporte = Column(String)
22     fundacion = Column(Integer, nullable=False)
23     ....
24     def __repr__(self):
25         return "Club: nombre=%s deporte=%s fundacion=%d" % (
26             self.nombre,
27             self.deporte,
28             self.fundacion)
29
```

```
30 class Jugador(Base):
31     __tablename__ = 'jugador'
32     id = Column(Integer, primary_key=True)
33     nombre = Column(String, nullable=False)
34     dorsal = Column(Integer)
35     posicion = Column(String)
36     club_id = Column(Integer, ForeignKey('club.id'))
37     club = relationship("Club", back_populates="jugadores")
38     ....
39     def __repr__(self):
40         return "Jugador: %s - dorsal:%d - posición: %s" % (
41             self.nombre, self.dorsal, self.posicion)
42
43 Club.jugadores = relationship("Jugador", \
44     back_populates="club")
```

Relaciones de entidades con ORM SQLAlchemy Ejemplo

Generación de Tablas o Entidades - Vista Sqlite

| | | |
|------------|---------|---|
| Tables (2) | | |
| club | | CREATE TABLE club (id INTEGER NOT NULL, nombre VARCHAR, deporte VARCHAR, f |
| id | INTEGER | `id` INTEGER NOT NULL |
| nombre | VARCHAR | `nombre` VARCHAR |
| deporte | VARCHAR | `deporte` VARCHAR |
| fundacion | INTEGER | `fundacion` INTEGER NOT NULL |
| jugador | | CREATE TABLE jugador (id INTEGER NOT NULL, nombre VARCHAR NOT NULL, dorsal |
| id | INTEGER | `id` INTEGER NOT NULL |
| nombre | VARCHAR | `nombre` VARCHAR NOT NULL |
| dorsal | INTEGER | `dorsal` INTEGER |
| posicion | VARCHAR | `posicion` VARCHAR |
| club_id | INTEGER | `club_id` INTEGER |

Relaciones de entidades con ORM SQLAlchemy Ejemplo

Generación de registros

```
19 # se crea un objetos de tipo Club.  
20 club1 = Club(nombre="Barcelona", deporte="Fútbol", \  
21             fundacion=1920)  
22  
23 club2 = Club(nombre="Emelec", deporte="Fútbol", \  
24             fundacion=1930)  
25  
26 club3 = Club(nombre="Liga de Quito", deporte="Fútbol", \  
27             fundacion=1940)
```


Relaciones de entidades con ORM SQLAlchemy Ejemplo

Generación de registros

```
29 # Se crean objeto de tipo Jugador
30 #
31 jugador1 = Jugador(nombre = "Damian Diaz", dorsal=10, posicion="mediocampo", \
32                     club=club1)
33 jugador2 = Jugador(nombre = "Matias Oyola", dorsal=18, posicion="mediocampo", \
34                     club=club1)
35 jugador3 = Jugador(nombre = "Dario Aymar", dorsal=2, posicion="defensa", \
36                     club=club1)
37
38
39 jugador4 = Jugador(nombre = "Oscar Bagui", dorsal=6, posicion="defensa", \
40                     club=club2)
41 jugador5 = Jugador(nombre = "Romario Caicedo", dorsal=11, posicion="mediocampo", \
42                     club=club2)
43
44
45 jugador6 = Jugador(nombre = "Adrián Gabbarini", dorsal=1, posicion="arquero", \
46                     club=club3)
47 jugador7 = Jugador(nombre = "Cristian Martinez", dorsal=9, posicion="delantero", \
48                     club=club3)
```

Relaciones de entidades con ORM SQLAlchemy Ejemplo

Generación de registros

Table:  club

| | id | nombre | deporte | fundacion |
|---|--------|---------------|---------|-----------|
| | Filter | Filter | Filter | Filter |
| 1 | 1 | Barcelona | Fútbol | 1920 |
| 2 | 2 | Emelec | Fútbol | 1930 |
| 3 | 3 | Liga de Quito | Fútbol | 1940 |

Table:  jugador

| | id | nombre | dorsal | posicion | club_id |
|---|--------|-----------------|--------|------------|---------|
| | Filter | Filter | Filter | Filter | Filter |
| 1 | 1 | Damian Diaz | 10 | mediocampo | 1 |
| 2 | 2 | Matias Oyola | 18 | mediocampo | 1 |
| 3 | 3 | Dario Aymar | 2 | defensa | 1 |
| 4 | 4 | Oscar Bagui | 6 | defensa | 2 |
| 5 | 5 | Romario Cai... | 11 | mediocampo | 2 |
| 6 | 6 | Adrián Gab... | 1 | arquero | 3 |
| 7 | 7 | Cristian Mar... | 9 | delantero | 3 |



Relaciones de entidades con ORM SQLAlchemy Ejemplo

Consulta de Datos

```
20 # Obtener todos los registros de
21 # la entidad Club
22 clubs = session.query(Club).all()
23
24 # Se recorre la lista a través de un ciclo
25 # repetitivo for en python
26 print("Presentación de Clubs")
27 for s in clubs:
28     print("%s" % (s))
29     print("-----")
```

```
Presentación de Clubs
Club: nombre=Barcelona deporte=Fútbol fundacion=1920
-----
Club: nombre=Emelec deporte=Fútbol fundacion=1930
-----
Club: nombre=Liga de Quito deporte=Fútbol fundacion=1940
-----
```

Relaciones de entidades con ORM SQLAlchemy Ejemplo

Consulta de Datos

```
31 # Obtener todos los registros de
32 # la entidad Jugador
33 jugadores = session.query(Jugador).all()
34
35 # Se recorre la lista a través de un ciclo
36 # repetitivo for en python
37
38 print("Jugadores")
39 for s in jugadores:
40     print("%s" % (s))
41     print("-----")
```

```
Jugadores
Jugador: Damian Diaz - dorsal:10 - posición: mediocampo
-----
Jugador: Matias Oyola - dorsal:18 - posición: mediocampo
-----
Jugador: Dario Aymar - dorsal:2 - posición: defensa
-----
Jugador: Oscar Bagui - dorsal:6 - posición: defensa
-----
Jugador: Romario Caicedo - dorsal:11 - posición: mediocampo
-----
Jugador: Adrián Gabbarini - dorsal:1 - posición: arquero
-----
Jugador: Cristian Martinez - dorsal:9 - posición: delantero
-----
```


Relaciones de entidades con ORM SQLAlchemy Ejemplo

Consulta de Datos

```
21 # Obtener todos los registros de·
22 # la entidad estudiantes (clase Estudiante)·
23 jugadores = session.query(Jugador).all()·
24
25 # Se recorre la lista a través de un ciclo
26 # repetitivo for en python
27 print("Presentación de Jugadores")
28 for s in jugadores:
29     print("%s" % (s))
30     # desde cada objeto de la lista
31     # jugador
32     # se puede acceder al club; como lo definimos
33     # al momento de crear la clase Jugador
34     print("El Jugador pertenece a: %s " % (s.club))
35     print("-----")
36
```

```
Presentación de Jugadores
Jugador: Damian Diaz - dorsal:10 - posición: mediocampo
El Jugador pertenece a: Club: nombre=Barcelona deporte=Fútbol fundacion=1920
-----
Jugador: Matias Oyola - dorsal:18 - posición: mediocampo
El Jugador pertenece a: Club: nombre=Barcelona deporte=Fútbol fundacion=1920
-----
Jugador: Dario Aymar - dorsal:2 - posición: defensa
El Jugador pertenece a: Club: nombre=Barcelona deporte=Fútbol fundacion=1920
-----
Jugador: Oscar Bagui - dorsal:6 - posición: defensa
El Jugador pertenece a: Club: nombre=Emelec deporte=Fútbol fundacion=1930
-----
Jugador: Romario Caicedo - dorsal:11 - posición: mediocampo
El Jugador pertenece a: Club: nombre=Emelec deporte=Fútbol fundacion=1930
-----
Jugador: Adrián Gabbarini - dorsal:1 - posición: arquero
El Jugador pertenece a: Club: nombre=Liga de Quito deporte=Fútbol fundacion=1940
```

Relaciones de entidades con ORM SQLAlchemy Ejemplo

Consulta de Datos

```
37 print("Presentación de Jugadores - op2")
38 for s in jugadores:
39     print("%s" % (s))
40     # desde cada objeto de la lista
41     # jugador
42     # se puede acceder al club; como lo definimos
43     # al momento de crear la clase Jugador
44     print("El Jugador pertenece a: %s " % (s.club.nombre))
45     print("-----")
```

```
Presentación de Jugadores - op2
Jugador: Damian Diaz - dorsal:10 - posición: mediocampo
El Jugador pertenece a: Barcelona
-----
Jugador: Matias Oyola - dorsal:18 - posición: mediocampo
El Jugador pertenece a: Barcelona
-----
Jugador: Dario Aymar - dorsal:2 - posición: defensa
El Jugador pertenece a: Barcelona
-----
Jugador: Oscar Bagui - dorsal:6 - posición: defensa
El Jugador pertenece a: Emelec
-----
Jugador: Romario Caicedo - dorsal:11 - posición: mediocampo
El Jugador pertenece a: Emelec
-----
Jugador: Adrián Gabbarini - dorsal:1 - posición: arquero
El Jugador pertenece a: Liga de Quito
-----
Jugador: Cristian Martinez - dorsal:9 - posición: delantero
El Jugador pertenece a: Liga de Quito
```

Relaciones de entidades con ORM SQLAlchemy Ejemplo

Consulta de Datos

```
21 # Obtener todos los registros de
22 # la entidad Club.
23 clubs = session.query(Club).all()
24
25 # Se recorre la lista a través de un ciclo
26 # repetitivo for en python
27 print("Presentación de Clubs y sus jugadores")
28 for s in clubs:
29     print("%s" % (s))
30     # desde cada objeto de la lista
31     # de tipo Club
32     # se puede acceder
33     # a los jugadores
34     jugadores = s.jugadores # es una secuencia
35     for i in jugadores:
36         print(i)
```

```
Presentación de Clubs y sus jugadores
Club: nombre=Barcelona deporte=Fútbol fundacion=1920
Jugador: Damian Diaz - dorsal:10 - posición: mediocampo
Jugador: Matias Oyola - dorsal:18 - posición: mediocampo
Jugador: Dario Aymar - dorsal:2 - posición: defensa
-----
Club: nombre=Emelec deporte=Fútbol fundacion=1930
Jugador: Oscar Bagui - dorsal:6 - posición: defensa
Jugador: Romario Caicedo - dorsal:11 - posición: mediocampo
-----
Club: nombre=Liga de Quito deporte=Fútbol fundacion=1940
Jugador: Adrián Gabbarini - dorsal:1 - posición: arquero
Jugador: Cristian Martinez - dorsal:9 - posición: delantero
-----
```

Relaciones de entidades con ORM SQLAlchemy Ejemplo

Consulta de Datos

```
21 # Obtener un listado de todos los registros.  
22 # de la tabla Club, que tengan al menos.  
23 # un jugador con el nombre que tenga incluida la cadena "Da"  
24  
25 # para la solución se hace uso del método.  
26 # join aplicado a query  
27  
28 clubs = session.query(Club).join(Jugador).\br/>29     filter(Jugador.nombre.like("%Da%")).all()  
30  
31 print("Consulta 1 ")  
32 for e in clubs:  
33     print(e)
```

```
Consulta 1  
Club: nombre=Barcelona deporte=Fútbol fundacion=1920
```



Relaciones de entidades con ORM SQLAlchemy Ejemplo

Consulta de Datos

```
35 # Obtener un listado de todos los registros.
36 # de la tabla Club y Jugador, que tengan al menos.
37 # un jugador con el nombre que tenga incluida la cadena "Da"
38
39 # para la solución se hace uso del método.
40 # join aplicado a query
41 # en el query se ubican las dos entidades involucradas
42 #.
43
44 registros = session.query(Club, Jugador).join(Jugador).\
45     filter(Jugador.nombre.like("%Da%")).all()
46 .
47 print("Consulta 2 ")
48 .
49 for registro in registros:
50     # el registro continen.
51     # dos valores en un tupla
52     # posición 0 el club
53     # posición 1 el jugador.
54     # que cumplen con la condición
55     print(registro[0])
56     print(registro[1])
```

Consulta 2

Club: nombre=Barcelona deporte=Fútbol fundacion=1920

Jugador: Damian Diaz - dorsal:10 - posición: mediocampo

Club: nombre=Barcelona deporte=Fútbol fundacion=1920

Jugador: Dario Aymar - dorsal:2 - posición: defensa



UTPL
UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA

Ingeniería en Computación

Gracias

r r e l i z a l d e @ u t p l . e d u . e c
@ r e r o e s