



No SQLAlchemy, o método `query` é comumente usado para interagir com o banco de dados e recuperar dados usando modelos ORM (Object Relational Mapping). Aqui estão alguns métodos comuns e úteis que você pode usar com `query` no SQLAlchemy para realizar consultas e manipulação de dados:

1. `session.query(Model)`

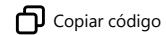
Inicia uma consulta em uma tabela/modelo específico.

2. `query.filter()`

Aplica uma condição de filtro na consulta. Aceita expressões como `Model.campo == valor`.

python

```
session.query(User).filter(User.age > 18)
```

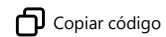


3. `query.filter_by()`

Filtro direto usando argumentos nomeados.

python

```
session.query(User).filter_by(name="Alice")
```

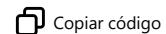


4. `query.all()`

Retorna todos os registros que correspondem à consulta.

python

```
session.query(User).all()
```

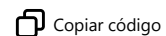


5. `query.first()`

Retorna o primeiro registro encontrado.

python

```
session.query(User).filter(User.age > 18).first()
```

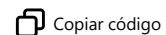


6. `query.one()`

Retorna exatamente um registro, lançando exceção se houver zero ou mais de um.

python

```
session.query(User).filter(User.id == 1).one()
```

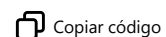


7. `query.one_or_none()`

Retorna um único registro ou `None`, evitando exceção se não houver resultados.

python

```
session.query(User).filter(User.name == "Alice").one_or_none()
```

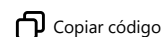


8. `query.get(primary_key)`

Busca um registro pelo valor de sua chave primária.

python


```
session.query(User).get(1)
```



9. `query.order_by()`

Ordena os resultados da consulta.

python


 Copiar código

```
session.query(User).order_by(User.age.desc())
```

#### 10. query.limit()

Limita o número de resultados.

python


 Copiar código

```
session.query(User).limit(10)
```

#### 11. query.offset()

Ignora um número específico de registros antes de começar a retornar resultados.

python


 Copiar código

```
session.query(User).offset(5).limit(10)
```

#### 12. query.count()

Retorna o número total de registros que correspondem à consulta.

python

 Copiar código

```
session.query(User).filter(User.age > 18).count()
```

#### 13. query.join()

Realiza uma junção entre tabelas com base em um relacionamento ou uma condição.

python


 Copiar código

```
session.query(User).join(Order, User.id == Order.user_id)
```

#### 14. query.group\_by()

Agrupa os resultados da consulta com base em uma ou mais colunas.

python


 Copiar código

```
session.query(User.city, func.count(User.id)).group_by(User.city)
```

#### 15. query.distinct()

Retorna resultados distintos, eliminando duplicatas.

python


 Copiar código

```
session.query(User.city).distinct()
```

#### 16. query.exists()

Verifica se existe algum registro que corresponda à condição.

python

 Copiar código

```
session.query(User).filter(User.age > 18).exists()
```

Esses métodos são poderosos para construir consultas SQL complexas e são frequentemente combinados para consultas mais detalhadas.