



무료 전자 책

배우기

sqlalchemy

Free unaffiliated eBook created from
Stack Overflow contributors.

#sqlalchemy

y

| | |
|----------------------------|-----------|
| | 1 |
| 1: sqlalchemy | 2 |
| | 2 |
| | 2 |
| Examples..... | 2 |
| | 2 |
| , ! (SQLAlchemy) | 2 |
| h11 | 3 |
| , ! (SQLAlchemy ORM) | 3 |
| 2: ORM | 5 |
| | 5 |
| Examples | 5 |
| dict | 5 |
| | 6 |
| | 6 |
| | 7 |
| 3: SQLAlchemy | 9 |
| Examples..... | 9 |
| dict | 9 |
| 4: | 10 |
| | 10 |
| Examples..... | 10 |
| | 10 |
| | 10 |
| 5: | 12 |
| Examples..... | 12 |
| | 12 |
| | 12 |
| | 12 |
| | 12 |

| | |
|----------------------|----|
| 6: - SQLAlchemy..... | 14 |
| | 14 |
| Examples..... | 14 |
| | 14 |
| | 15 |

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: [sqlalchemy](#)

It is an unofficial and free sqlalchemy ebook created for educational purposes. All the content is extracted from [Stack Overflow Documentation](#), which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official sqlalchemy.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

1: sqlalchemy

SQLALCHEMY

SQLAlchemy :

SQL . SQLAlchemy .

SQLAlchemy . select . SQLAlchemy .

SQLAlchemy ORM (Object-Relational Mapper) .

SQLAlchemy SQL / ORM . SQL . SQL .

SQLAlchemy SQL !

| 1.1 | 1.1 | 2016-07-26 |
|-----|---------------------|------------|
| 1.0 | 1.0 | 2015-04-16 |
| 0.9 | 0.9 | 2013-12-30 |
| 0.8 | 0.8 | 2013-03-09 |

Examples

```
pip install sqlalchemy
```

, flask-sqlalchemy .

```
pip install flask-sqlalchemy
```

, ! (SQLAlchemy)

SQLAlchemy Core , , . [SQLAlchemy ORM](#) .

```
from sqlalchemy import create_engine  
engine = create_engine('sqlite://')
```

SQLAlchemy . SQLAlchemy DBAPI " " / DBAPI . DBAPI dialect .

```

from sqlalchemy import Column, Integer, Text, MetaData, Table

metadata = MetaData()
messages = Table(
    'messages', metadata,
    Column('id', Integer, primary_key=True),
    Column('message', Text),
)

messages.create(bind=engine)

```

MetaData

SQL CREATE TABLE MetaData

!

```

insert_message = messages.insert().values(message='Hello, World!')
engine.execute(insert_message)

```

select . Column Table c . select fetchone () , fetchall () fetchmany () ResultProxy .
select .

```

from sqlalchemy import select
stmt = select([messages.c.message])
message, = engine.execute(stmt).fetchone()
print(message)

```

Hello, World!

! [SQLAlchemy SQL Expressions](#) .

, ! (SQLAlchemy ORM)

SQLAlchemy ORM . [SQLAlchemy Core](#) .

SQLAlchemy Core (Core) .

```

from sqlalchemy import create_engine

engine = create_engine('sqlite://')

```

. SQLAlchemy ORM Core . ORM . SQLAlchemy [Declarative](#) ORM .

```

from sqlalchemy.ext.declarative import declarative_base
Base = declarative_base()

```

models .

```

from sqlalchemy import Column, Integer, String

class Message(Base):
    __tablename__ = 'messages'

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

```

Table Mapper . :

Table MetaData . Declarative .metadata .

SQLAlchemy Core MetaData .

```
Base.metadata.create_all(engine)
```

!

```

message = Message(message="Hello World!")
message.message # 'Hello World!'

```

ORM . .

```

from sqlalchemy.orm import sessionmaker

Session = sessionmaker(bind=engine)
session = Session()

```

. SQLAlchemy :

```

session.add(message)
session.commit()

```

ORM .

```

query = session.query(Message)
instance = query.first()
print (instance.message) # Hello World!

```

! filter , order_by . [SQLAlchemy ORM](#) .

[sqlalchemy](https://riptutorial.com/ko/sqlalchemy/topic/1697/sqlalchemy-) : <https://riptutorial.com/ko/sqlalchemy/topic/1697/sqlalchemy->

2: ORM

SQLAlchemy ORM [SQLAlchemy Core](#) . , Column .

ORM , (SQLAlchemy).

Examples

dict

:

```
import datetime as dt
from sqlalchemy import Column, Date, Integer, Text, create_engine, inspect
from sqlalchemy.orm import sessionmaker
from sqlalchemy.ext.declarative import declarative_base

Base = declarative_base()
Session = sessionmaker()

class User(Base):
    __tablename__ = 'users'

    id = Column(Integer, primary_key=True)
    name = Column(Text, nullable=False)
    birthday = Column(Date)

engine = create_engine('sqlite://')
Base.metadata.create_all(bind=engine)
Session.configure(bind=engine)

session = Session()
session.add(User(name='Alice', birthday=dt.date(1990, 1, 1)))
session.commit()
```

KeyedTuple `_asdict` . [namedtuple API](#) (private!).

```
query = session.query(User.name, User.birthday)
for row in query:
    print(row._asdict())
```

ORM . SQLAlchemy .

```
def object_as_dict(obj):
    return {c.key: getattr(obj, c.key)
            for c in inspect(obj).mapper.column_attrs}

query = session.query(User)
for user in query:
    print(object_as_dict(user))
```


declarative_base .

```
from sqlalchemy.ext.declarative import as_declarative

@as_declarative()
class Base:
    def _asdict(self):
        return {c.key: getattr(self, c.key)
                for c in inspect(self).mapper.column_attrs}
```

```
class User(Base):
    __tablename__ = 'users'

    id = Column(Integer, primary_key=True)
    name = Column(Text, nullable=False)
    birthday = Column(Date)
```

```
import datetime as dt
session.query(User).filter(User.name == 'Bob')
session.query(User).filter(User.birthday < dt.date(2000, 1, 1))
```

```
session.query(User).filter_by(name='Bob')
```

AND filter :

```
(session.query(User).filter(User.name.like('B%'))
    .filter(User.birthday < dt.date(2000, 1, 1)))
```

& | | :

```
session.query(User).filter((User.name == 'Bob') | (User.name == 'George'))
```

```
class SpreadsheetCells(Base):
    __tablename__ = 'spreadsheet_cells'

    id = Column(Integer, primary_key=True)
    y_index = Column(Integer)
    x_index = Column(Integer)
```

order_by .

```
query = session.query(SpreadsheetCells).order_by(SpreadsheetCells.y_index)
```

filter .

```
query = session.query(...).filter(...).order_by(...)
```

.

```
query = session.query(...).filter(...)  
ordered_query = query.order_by(...)
```

.

1. asc desc :

```
query.order_by(SpreadsheetCells.y_index.desc()) # desc  
query.order_by(SpreadsheetCells.y_index.asc()) # asc
```

2. asc desc :

```
from sqlalchemy import asc, desc  
  
query.order_by(desc(SpreadsheetCells.y_index)) # desc  
query.order_by(asc(SpreadsheetCells.y_index)) # asc
```

for .

:

```
from datetime import date  
  
class User(Base):  
    __tablename__ = 'users'  
  
    id = Column(Integer, primary_key=True)  
    name = Column(Text, nullable=False)  
    birthday = Column(Date)  
  
    # Find users who are older than a cutoff.  
    query = session.query(User).filter(User.birthday < date(1995, 3, 3))
```

all() all() .

```
reslist = query.all() # all results loaded in memory  
nrows = len(reslist)
```

count() count() .

```
nrows = query.count()
```

first() . order_by() .

```
oldest_user = query.order_by(User.birthday).first()
```

one() .

```
bob = session.query(User).filter(User.name == 'Bob').one()
```

. one_or_none() one_or_none() .

```
bob = session.query(User).filter(User.name == 'Bob').one_or_none()
if bob is None:
    create_bob()
```

'Bob' .

ORM : <https://riptutorial.com/ko/sqlalchemy/topic/2020/orm>

3: SQLAlchemy

Examples

dict

SQLAlchemy RowProxy . dict(row) .

:

```
import datetime as dt
from sqlalchemy import (
    Column, Date, Integer, MetaData, Table, Text, create_engine, select)

metadata = MetaData()
users = Table(
    'users', metadata,
    Column('id', Integer, primary_key=True),
    Column('name', Text, nullable=False),
    Column('birthday', Date),
)

engine = create_engine('sqlite://')
metadata.create_all(bind=engine)

engine.execute(users.insert(), name='Alice', birthday=dt.date(1990, 1, 1))
```

```
with engine.connect() as conn:
    result = conn.execute(users.select())
    for row in result:
        print(dict(row))

    result = conn.execute(select([users.c.name, users.c.birthday]))
    for row in result:
        print(dict(row))
```

SQLAlchemy : <https://riptutorial.com/ko/sqlalchemy/topic/2022/sqlalchemy->

4:

ORM .

Examples

Session sessionmaker sessionmaker . Session .

```
from sqlalchemy.orm import sessionmaker

# Initial configuration arguments
Session = sessionmaker(bind=engine)
```

engine Session .

```
# This session is bound to provided engine
session = Session()
```

Session.configure() Session.configure() :).

```
Session = sessionmaker()

# later
Session.configure(bind=engine)
```

Session sessionmaker .

```
session_bound_to_engine2 = Session(bind=engine2)
```

add() .

```
session.add(obj)
```

add_all() .

```
session.add_all([obj1, obj2, obj3])
```

INSERT . .

delete delete() .

```
session.delete(obj)
```

.

: <https://riptutorial.com/ko/sqlalchemy/topic/2258/>

5:

Examples

URL .

```
from sqlalchemy import create_engine

engine = create_engine('postgresql://user:pass@localhost/test')
```

.
(, SQLAlchemy 5).

(,).

```
with engine.connect() as conn:
    result = conn.execute('SELECT price FROM products')
    for row in result:
        print('Price:', row['price'])
```

```
conn = engine.connect()
result = conn.execute('SELECT price FROM products')
for row in result:
    print('Price:', row['price'])
conn.close()
```

```
result = engine.execute('SELECT price FROM products')
for row in result:
    print('Price:', row['price'])
```

engine.begin . .

```
with engine.begin() as conn:
    conn.execute(products.insert(), price=15)
```

, .

```
with conn.begin() as trans:
    conn.execute(products.insert(), price=15)
```

execute . trans.rollback() trans.rollback() .

.

```
trans = conn.begin()
try:
    conn.execute(products.insert(), price=15)
    trans.commit()
except:
    trans.rollback()
    raise
```

: [https://riptutorial.com/ko/sqlalchemy/topic/2025/-](https://riptutorial.com/ko/sqlalchemy/topic/2025/)

6: - SQLAlchemy

Flask-SQLAlchemy .

Examples

Flask , Flask SQLAlchemy .

sqlalchemy sqlalchemy.orm . Model .

```
from flask import Flask
from flask_sqlalchemy import SQLAlchemy

app = Flask(__name__)
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///tmp/test.db'
db = SQLAlchemy(app)

class User(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(80), unique=True)
    email = db.Column(db.String(120), unique=True)

    def __init__(self, username, email):
        self.username = username
        self.email = email

    def __repr__(self):
        return '<User %r>' % self.username
```

- SQLAlchemy : <https://riptutorial.com/ko/sqlalchemy/topic/4601/---sqlalchemy>

| S. No | | Contributors |
|-------|--------------|--|
| 1 | sqlalchemy | adarsh , Community , Mattew Whitt , RazerM , Stephen Fuhry |
| 2 | ORM | Ilja Everilä , Mattew Whitt , RazerM , Tom Hunt |
| 3 | SQLAlchemy | RazerM |
| 4 | | Ilja Everilä , RazerM |
| 5 | | RazerM |
| 6 | - SQLAlchemy | Ilya Rusin |