SQLAlchemy et SQLSoup

Afpyro Lyon 24 mai 2012

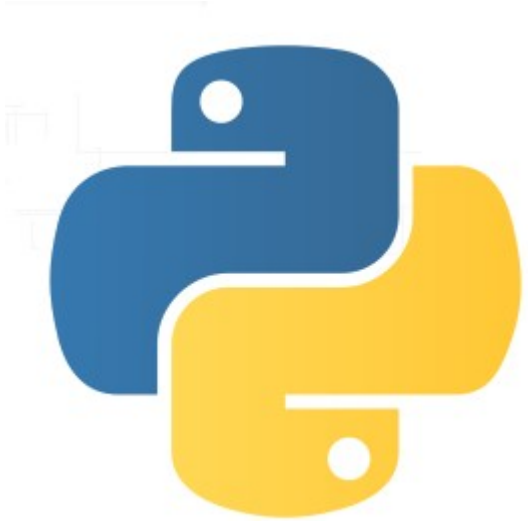
À propos de moi

À propos de moi

À propos de moi

À propos de moi

À propos de moi



À propos de moi

SQLAlchemy

SQLAlchemy

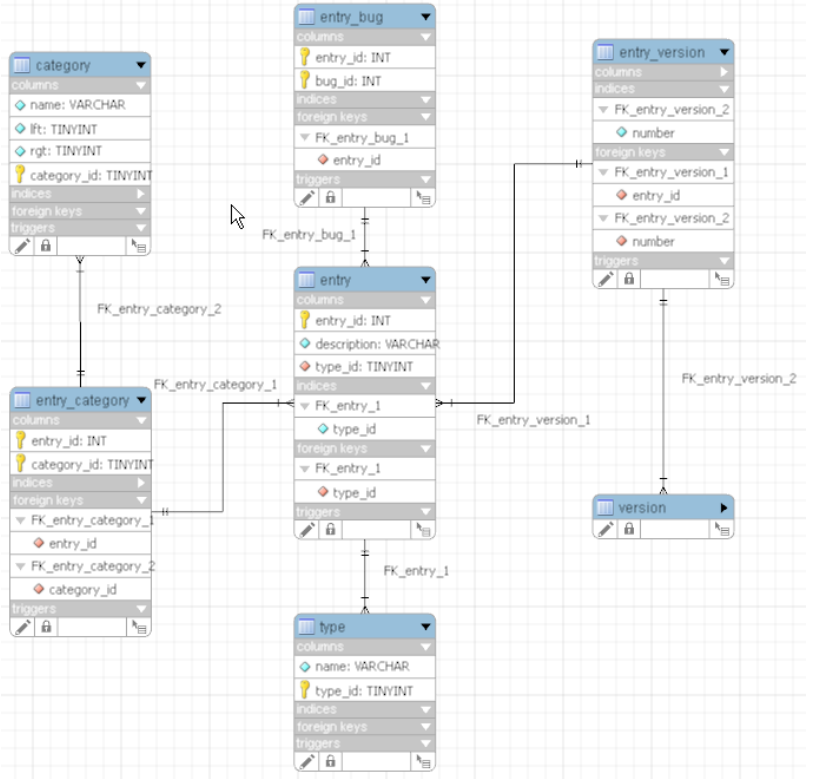
SQLAlchemy

SQLAlchemy

SQL

SQLAlchemy **SQL**

Structured Query Language



SQLAlchemy

SQLAlchemy

SQLAlchemy

SQLAlchemy

SQLAlchemy


SQLAlchemy
MySQL
PostgreSQL
SQLite
ORACLE
Microsoft®
SQL Server®
Firebird


SQLAlchemy
MySQL
PostgreSQL
SQLite
ORACLE
Microsoft®
SQL Server®
Firebird
SYBASE®
DB2

SQLAlchemy

CORE

SQLAlchemy **CORE**

Define engine configuration

```
from sqlalchemy import create_engine

engine = create_engine('sqlite:///dbname.sqlite')

engine = create_engine('mysql+mysqldb://login:password@host/dbname')
```

SQLAlchemy **CORE**

Execute SQL

```
connection = engine.connect()
result = connection.execute("select username from users")
for row in result:
    print "username:", row['username']
connection.close()
```

SQLAlchemy **CORE**

Managing transaction

```
trans = connection.begin()
try:
    ## do some work here
    trans.commit()
except:
    trans.rollback()
    raise
```

SQLAlchemy ORM

SQLAlchemy **ORM**

Object Relationnal Mapping

SQLAlchemy **ORM**

Object Relationnal Mapping

←T→	id	name	context
☐ ✎ ✕	1	Webmail	wm
☐ ✎ ✕	2	Dav	dav
☐ ✎ ✕	3	SyncML	funambol
☐ ✎ ✕	4	Bond Diffusor	bond-diffusor
☐ ✎ ✕	5	Factory	factory
☐ ✎ ✕	6	Lord	lord
☐ ✎ ✕	7	Bond AlintoV1	bond-alintov1
☐ ✎ ✕	8	Webmail Mobile	mobile
☐ ✎ ✕	9	Protect	protect
☐ ✎ ✕	10	Bond Protect	bond-protect
☐ ✎ ✕	11	CalDAV	caldav
☐ ✎ ✕	12	Report	report
☐ ✎ ✕	13	Combine	combine

SQLAlchemy **ORM**

Object Relationnal Mapping

←T→			id	name	context
☐	✎	✗	1	Webmail	wm
☐	✎	✗	2	Dav	dav
☐	✎	✗	3	SyncML	funambol
☐	✎	✗	4	Bond Diffusor	bond-diffusor
☐	✎	✗	5	Factory	factory
☐	✎	✗	6	Lord	lord
☐	✎	✗	7	Bond AlintoV1	bond-alintov1
☐	✎	✗	8	Webmail Mobile	mobile
☐	✎	✗	9	Protect	protect
☐	✎	✗	10	Bond Protect	bond-protect
☐	✎	✗	11	CalDAV	caldav
☐	✎	✗	12	Report	report
☐	✎	✗	13	Combine	combine

```
class Webapp(Base):
    __tablename__ = "verman_webapp"

    id = Column(Integer, primary_key=True)
    name = Column(String(50), unique=True)
    context = Column(String(50), unique=False)

    history = relationship("History")
```

SQLAlchemy **ORM**

Object Relationnal Mapping

	id	name	context
	1	Webmail	wm
	2	Dav	dav
	3	SyncML	funambol
	4	Bond Diffusor	bond-diffusor
	5	Factory	factory
	6	Lord	lord
	7	Bond AlintoV1	bond-alintov1
	8	Webmail Mobile	mobile
	9	Protect	protect
	10	Bond Protect	bond-protect
	11	CalDAV	caldav
	12	Report	report
	13	Combine	combine

```
class Webapp(Base):
    __tablename__ = "verman_webapp"

    id = Column(Integer, primary_key=True)
    name = Column(String(50), unique=True)
    context = Column(String(50), unique=False)

    history = relationship("History")
```

SQLAlchemy **ORM**

Define mapping

```
session = scoped_session(sessionmaker(autocommit=False,
                                      autoflush=False,
                                      bind=engine))
Base = declarative_base()
Base.query = session.query_property()

class History(Base):
    __tablename__ = "verman_history"

    id = Column(Integer, primary_key=True)
    first_seen_date = Column(DateTime)
    last_seen_date = Column(DateTime)

    webapp = Column(Integer, ForeignKey('verman_webapp.id'))

class Webapp(Base):
    __tablename__ = "verman_webapp"

    id = Column(Integer, primary_key=True)
    name = Column(String(50), unique=True)
    context = Column(String(50), unique=False)

    history = relationship("History")
```

SQLAlchemy **ORM**

Querying

```
server = session.query(Server).filter(Server.id == server_id).one()
```

SQLAlchemy **ORM**

Inserting

```
webapp = Webapp()
webapp.name = "AFPYRO"
webapp.context = "afpy.ro"

session.add( webapp )
session.commit()
```

SQLAlchemy **ORM**

Updating

```
webapp.name = "Afpyro updated"

session.commit()
```

SQLAlchemy **ORM**

Deleting

```
session.delete( webapp )

session.commit()
```

SQLAlchemy ORM

EASY !

SQLAlchemy ORM

EFFICIENT !

SQLAlchemy ORM

SIMPLE !

SQLSoup

SQLSoup

=

SQLSoup

=

SQLSoup

=

SQLAlchemy - mapping definition

SQLSoup

```
from sqlalchemy.ext.sqlsoup import SqlSoup

db = SqlSoup('mysql://user:password@host/timesheet')

start_time = datetime.datetime(2012, 05, 01, 00, 00, 00)
end_time = start_time + datetime.timedelta(hours=7)

db.timesheet_times.insert(
    uid = "avuillard",
    start_time = start_time,
    end_time = end_time,
    proj_id = 1,
    task_id = 4
)

db.commit()
```

Arthur Vuillard

arthur.vuillard@gmail.com

http://www.sqlalchemy.org/
https://bitbucket.org/zzzeek/sqlsoup