课程介绍

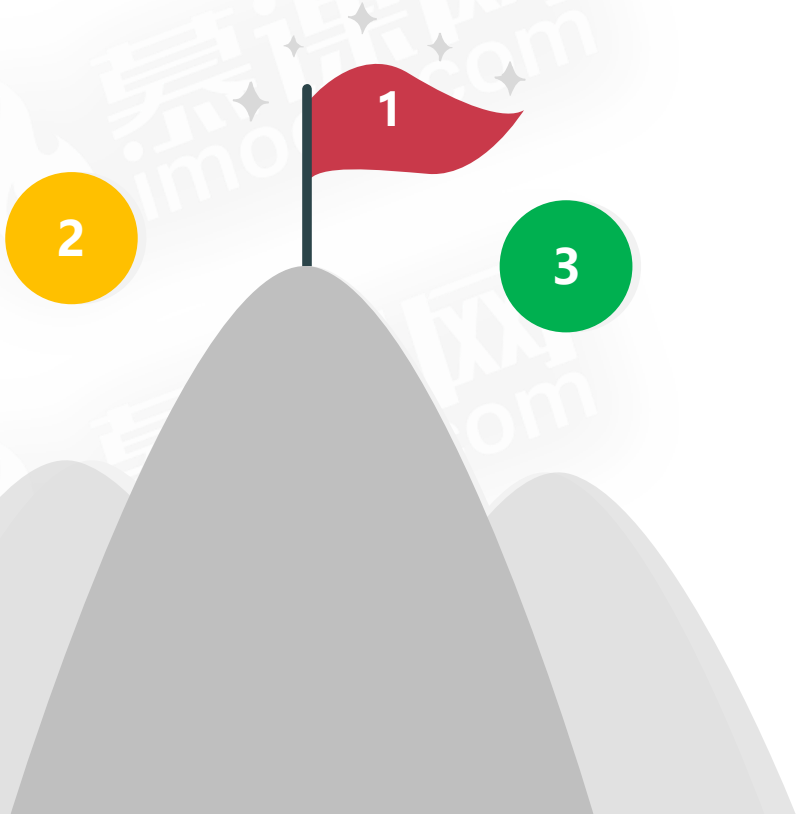
学习目标

MySQL-Connector模块

数据库连接池、预编译SQL、CRUD操作、事物管理、异常处理

新闻管理系统

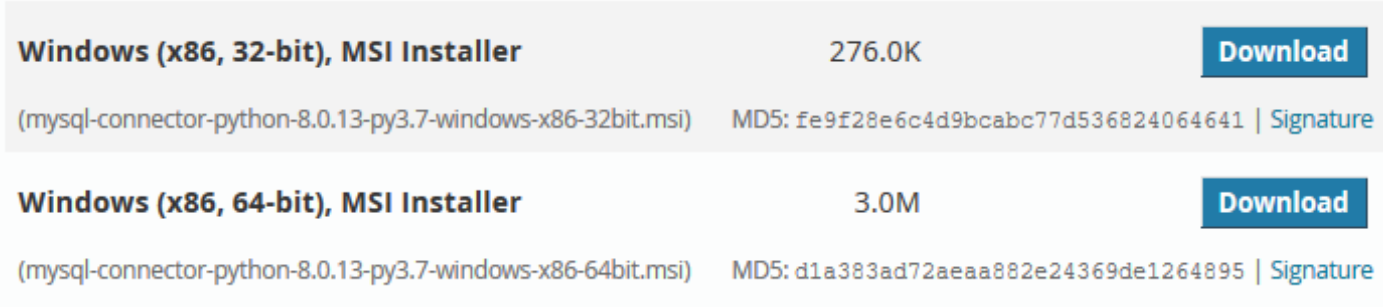
新闻管理、用户管理、系统登陆、数据分页……



MySQL Connector（一）

MySQL官方驱动模块

- MySQL Connector是MySQL官方的驱动模块，兼容性特别好
- https://dev.mysql.com/downloads/connector/python/



创建连接（一）

```
import mysql.connector
con = mysql.connector.connect(
    host="localhost", port="3306",
    user="root", password="abc123456",
    database="demo"
);
```

创建连接（二）

```
import mysql.connector
config = {
    "host": "localhost",
    "port": 3306,
    "user": "root",
    "password": "abc123456",
    "database": "demo"
}
con = mysql.connector.connect(**config)
```

游标(Cursor)

- MySQL Connector里面的游标用来执行SQL语句，而且查询的结果集也会保存在游标之中

```
cursor = con.cursor()
cursor.execute( sql语句 )
```

执行SQL语句

```
cursor=con.cursor()
sql = "SELECT empno,ename,hiredate FROM t_emp;"
cursor.execute(sql)
for one in cursor:
    print(one[0], one[1], one[2])
```

MySQL Connector（二）

SQL注入攻击案例

```
username="1 OR 1=1";
password="1 OR 1=1";
sql="SELECT COUNT(*) FROM t_user WHERE
username="+username+" AND
AES_DECRYPT(UNHEX(password),'HelloWorld') ="+password;
cursor.execute(sql);
print(cursor.fetchone()[0]);
```

SQL注入攻击的危害

- 由于SQL语句是解释型语言，所以在拼接SQL语句的时候，容易被注入恶意的SQL语句

```
id = "1 OR 1=1"
sql = "DELETE FROM t_news WHERE id=" + id;
```

SQL预编译机制

- 预编译SQL就是数据库提前把SQL语句编译成二进制，这样反复执行同一条SQL语句的效率就会提升


```
sql="INSERT INTO
t_emp(empno,ename,job,mgr,hiredate,sal,comm,deptno)
VALUES(%s,%s,%s,%s,%s,%s,%s,%s)";
```

SQL预编译机制抵御注入攻击

- SQL语句编译的过程中，关键字已经被解析过了，所以向编译后的SQL语句传入参数，都被当做字符串处理，数据库不会解析其中注入的SQL语句

```
id = "1 OR 1=1"
sql = "DELETE FROM t_news WHERE id=%s";
```

预防SQL注入攻击

```
username = "1 OR 1=1";
password = "1 OR 1=1";
sql = "SELECT COUNT(*) FROM t_user WHERE username=%s
AND AES_DECRYPT(UNHEX(password),'HelloWorld')=%s";
cursor.execute(sql, (username, password));
print(cursor.fetchone()[0]);
```

MySQL Connector（三）

事务控制

- Connector为我们提供了非常简单的事务控制函数

```
con.start_transaction( [ 事务隔离级别 ] )
con.commit()
con.rollback()
```

异常处理

```
try:
    con = mysql.connector.connect(……)
    [ con = start_transaction() ]
    ……
except Exception as e:
    [ con.rollback() ]
    print(e)
finally:
    if "con" in dir():
        con.close()
```

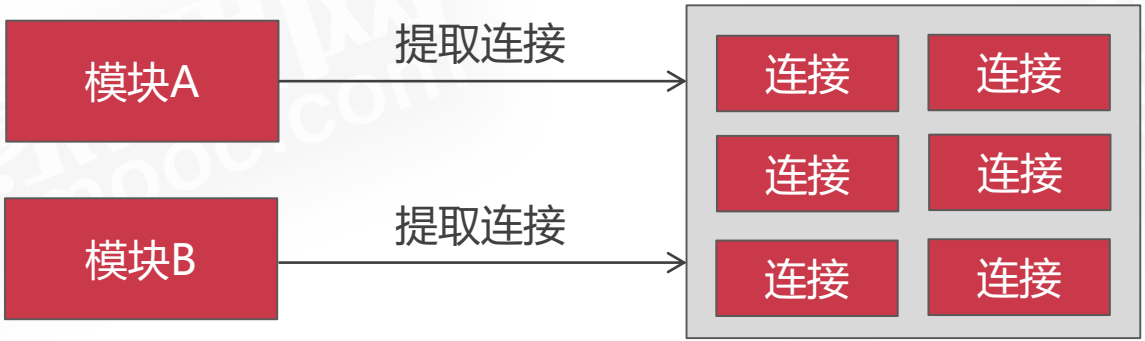
MySQL Connector（四）

数据库连接的昂贵之处

- 数据库连接是一种关键的、有限的、昂贵的资源，在并发执行的应用程序中体现得尤为突出。
- TCP连接需要三次握手，四次挥手，然后数据库还要验证用户信息

数据库连接池的意义

- 数据库连接池（Connection Pool）预先创建出一些数据库连接，然后缓存起来，避免了程序语言反复创建和销毁连接昂贵代价



数据库连接池的语法

```
import mysql.connector.pooling
config = {……}
pool = mysql.connector.pooling.MySQLConnectionPool(
    **config,
    pool_size=10
)
con=pool.get_connection()
```

MySQL Connector（五）

循环执行SQL语句

- 游标对象中的executeMany()函数可以反复执行一条SQL语句

```
sql = "INSERT INTO t_dept(deptno,dname,loc)
VALUES(%s,%s,%s)"
data = [[100, "A部门", "北京"], [110, "B部门", "上海"]]
cursor.executemany(sql, data)
```

MySQL Connector（六）

案例练习

- 使用INSERT语句，把部门平均底薪超过公司平均底薪的这样部门里的员工信息导入到t_emp_new表里面，并且让这些员工隶属于sales部门

- 编写一个INSERT语句向dept表插入两条记录，每条记录都在dept原有最大主键值的基础上+10

MySQL Connector（七）

案例练习

- 使用INSERT语句把所在部门平均工资超过公司平均工资的员工信息导入到t_emp_new表里面，并且让这些员工隶属于sales部门

- 编写一个INSERT语句向dept表插入两条记录，每条记录都在dept原有最大主键值的基础上+10

MySQL Connector（八）

案例练习

- 使用INSERT语句把所在部门平均工资超过公司平均工资的员工信息导入到t_emp_new表里面，并且让这些员工隶属于sales部门

- 编写一个INSERT语句向部门表插入两条记录，每条记录都在部门原有最大主键值的基础上+10

课程总结

技能清单

- 掌握了数据库连接池技术
- 掌握了Python程序的CRUD操作
- 掌握了用预编译SQL抵御SQL注入攻击
- 完成了新闻管理系统第一阶段的开发