# **Database Connection Pooling**

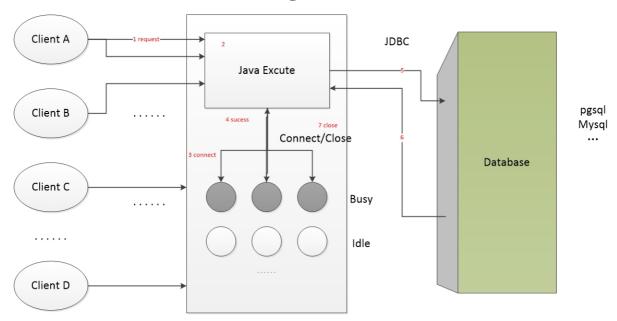
## **Pooling**

An import thought in program development, such as, Integer pool, string pool, object pool, connection pool, thread pool. Base on the thought of **reusing objects** to reduce resource consumption caused by creating and releasing objects, to achieve the goal that improving system performance.

#### Tips:

- reuse objects
- improve performance(save opening/closing time)
- a container(a pool)

## **Database Connection Pooling**



C3PO/Proxool/Dbcp/Druid/psycopg2.pool

#### **Describe about Druid**

- performance: LRU, load balancing technique
- stability
- expandability
- monitoring: a useful tool StatFilter has lower resource consumption
- security: against SQL injection attacks

**Tips**: The comparison with the other databases , <u>Chinese doc</u>.

# **Experiment 1: create a database connection pool and test connection statement**

- use Druid pooling structure
   https://druid.apache.org/docs/latest/design/
- Client: Java script(druidtest.java)-->DB: Postgresql
- set some parameters of pool, setInitialSize, setMinIdele, setMaxActive, setMaxWait Step1: Create a new Module and new java class to test Druid pooling, class name as you like

Step2: Here we use Maven to manager dependencies. If there has version conflict with local configuration, please change the detail in file pom.xml.

Step3: Import package and set connection information

Step4: get connection and check the connection statement

```
Connection connection = dataSource.getConnection();
    System.out.println(connection.getClass().getName());
```

### **Experiment 2: set some parameters of pool**

• set some parameters of pool, initialPoolSize, MaxPoolSize

```
dataSource.setInitialSize(3);
dataSource.setMaxActive(3);
```

**Tips**: The most necessary parameters: url username password, maxActive

## **Experiment 3: Get Statement of pool**

• Four Number of Connection in Pool, Busy, Close, Recycle, NumConnection

```
System.out.println("Busy Num " + dataSource.getActiveCount());
System.out.println("Close Num " + dataSource.getCloseCount());
System.out.println("Recycle Num "+ dataSource.getRecycleCount());
System.out.println("All Num " + dataSource.getConnectCount());
```

## **Experiment 4: Simple Query using pooling technique**

```
// insert
String sql_insert = ".....";
try {
    PreparedStatement ps_insert = connection.prepareStatement(sql_insert);
    ps_insert.executeLargeUpdate();
    poolStatus(dataSource);
}catch (Exception e){
    e.printStackTrace();
}
```