数据库连接池

数据库连接池（Connection pooling），主要用来分配、管理、释放数据库的连接。数据库连接池首先会创建若干个 Connection 对象并将这些对象放入到池中，当系统需要使用 Connection 对象时，数据库连接池会从池中分配一个事先创建好的 Connection 对象给系统，当系统使用完毕或超时后，数据库连接池会将该 Connection 对象重新放入到池中。这样就减少了创建 Connection 对象所耗费的资源和时间，可以提高对数据库操作的性能

- 数据库的连接的建立和关闭是非常消耗资源的
- 频繁地打开、关闭连接造成系统性能低下

数据库连接池的作用

- 资源重用
    - 由于数据库连接得到重用，避免了频繁创建、释放连接引起的大量性能开销。在减少系统消耗的基础上，另一方面也增进了系统运行环境的平稳性，减少了内存碎片以及数据库临时进程和线程的数量
- 更快的系统响应速度
    - 数据库连接池在初始化过程中，往往已经创建了若干数据库连接置于池中备用。此时连接的初始化工作均已完成。对于业务请求处理而言，直接利用现有可用连接，避免了数据库连接初始化和释放过程的时间开销，从而缩减了系统整体响应时间
- 新的资源分配手段
    - 对于多应用共享同一数据库的系统而言，可在应用层通过数据库连接的配置，实现数据库连接池技术。某一应用最大可用数据库连接数的限制，避免某一应用独占所有数据库资源
- 统一的连接管理，避免数据库连接泄漏
    - 在较为完备的数据库连接池实现中，可根据预先的连接占用超时设定，强制收回被占用连接。从而避免了常规数据库连接操作中可能出现的资源泄漏

连接池类

连接池类是对某一数据库所有连接的缓冲池，主要实现以下功能：

- 从连接池获取或创建可用连接
- 使用完毕之后，把连接返还给连接池
- 在系统关闭前，断开所有连接并释放连接占用的系统资源
- 能够处理无效连接，并能够限制连接池中的连接总数不低于某个预定值和不超过某个预定值

最小连接数与最大连接数

- 最小连接数：连接池一直保持的数据库连接
    - 如果应用程序对数据库连接的使用量不大，将会有大量的数据库连接资源被浪费
- 最大连接数：连接池能申请的最大连接数
    - 如果数据库连接请求超过次数，后面的数据库连接请求将被加入到等待队列中，会影响后续的数据库操作
- 如果最小连接数与最大连接数相差很大：那么最先连接请求将会获利，之后超过最小连接数量的连接请求等价于建立一个新的数据库连接。不过，这些大于最小连接数的数据库连接在使用完不会马上被释放，他将被放到连接池中等待重复使用或是空间超时后被释放

数据库连接池模拟

```java
// 需要实现DataSource接口
// 需要注意数据库连接池要保证线程安全
// LinkedList的添加和删除操作效率高
private static LinkedList<Connection> pool = (LinkedList<Connection>)
Collections.synchronizedList(new LinkedList<Connection>());

static {
    try {
        // 向连接池中放10个连接
        for (int i = 0; i < 10; i++) {
            Connection con = DBUtil.getConnection();
            pool.add(con);
        }
    } catch (SQLException e) {
        throw new ExceptionInInitializerError("初始化数据库连接失败！");
    }
}

public static Connection getConnectionFromPool() {
    Connection con = null;
    if (pool.size() > 0) {
        // 将连接池中的一个连接取出
        con = pool.removeFirst();
        return con;
    } else {
        throw new RuntimeException("无空闲连接");
    }
}

// 当程序用完连接后，需要将该连接重新放入到连接池中
public static void release(Connection con) {
    pool.addLast(con);
}
```

第三方连接池

在实际应用中，通常不需要我们自己编写数据库连接池，有很多组织都提供了数据库连接池

- c3p0：开源的，成熟的，高并发第三方数据库连接池，相关的文档资料比较完善
- DBCP：DataBase Connection Pool，由 Apache 开发的一个数据库连接池，性能不太好，Apache 又开发了 Tomcat JDBC Pool来替代 DBCP
- Druid：阿里巴巴出品，号称是 Java 语言中最好的数据库连接池，能够提供强大的监控和扩展功能
- HiKariCP：日语中光的意思，性能很好，非常轻巧

Druid

可以使用 properties 或者 XML 配置

```properties
driver=com.mysql.jdbc.Driver
url=jdbc:mysql://localhost:3306/test
username=root
password=1234
#初始化的连接个数
initialSize=10
#最大连接数
maxActive=20
#最小连接数
minIdle=10
```

DruidUtil

```java
// 得到一个Druid的数据源
private static DruidDataSource dataSource = null;

static {
    Properties properties = new Properties();

    try {
        //加载配置文件
 properties.load(DruidUtil.class.getClassLoader().getResourceAsStream("db.properties")
);
        //得到一个数据源
        dataSource =
(DruidDataSource)DruidDataSourceFactory.createDataSource(properties);
    } catch (IOException e) {
        e.printStackTrace();
    } catch (Exception e) {
        e.printStackTrace();
    }
}

// 从数据源中得到一个连接对象
// 这个返回的Connection实际上是Druid经过装饰之后的Cconnection
public static Connection getConnection() {
    try {
        return dataSource.getConnection();
    } catch (SQLException e) {
        throw new RuntimeException("服务器错误");
    }
}
```