# Jdbc（02）

## 事务

### 事物的四大特性（ACID）

* 原子性
* 一致性
* 隔离性
* 持久性

### Mysql中的事务

* 开启事务

Start transaction

* 结束事物

Commit || rollback

* 例如；

start transaction;

update account set balance=balance-100 where name='zs';

update account set balance=balance+100 where name='ls';

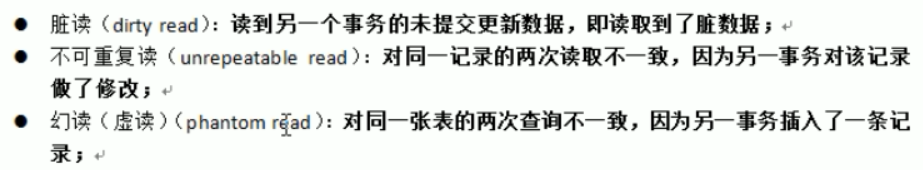
rollback; || commit;

### Jdbc事务

* Jdbc中处理事务，都通过Connection完成！
* 同一事务中的所有操作，都在使用同一个Connection对象。

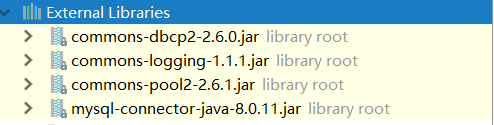
### 事务隔离级别

* 类型



## 数据库连接池(DBCP)

### Jar包

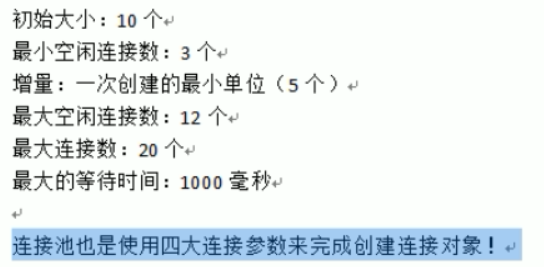


### 概念

* 管理连接的生命周期，创建连接，销毁连接；
* 连接池也是使用四大连接参数来完成来连接池的创建。

### 池参数

所有池参数都有默认值。



### 实现的接口

* 连接池必须实现javax.sql.DataSource接口！
* 连接池返回的connection对象，它的close()不是关闭连接，而是将连接归还给连接池。

### 对象增强的手段

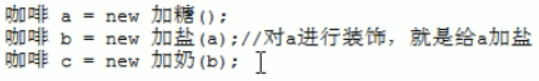
* 继承

缺点：

* 使用继承会使类的数量增多！
* 增强的内容是死的；
* 被增强的对象也是死的。
* 装饰者模式

特点：

* 增强的内容是固定的！
* 但是被增强的对象可以是任意的，即使不知道被增强对象的具体类型；
* 例如：



* 再例如：



* 动态代理（AOP）

特点；

* 被增强的对象可以切换：Service
* 增强的内容也可以切换：事务处理

## 数据库连接池(C3P0)

### 简介

开源免费的连接池，优于DBCP.

### 使用

*//创建连接池对象*ComboPooledDataSource comboPooledDataSource = **new** ComboPooledDataSource();  
*//对池进行四大参数配置*comboPooledDataSource.setDriverClass(**"com.mysql.jdbc.Driver"**);  
comboPooledDataSource.setJdbcUrl(**"jdbc:mysql://localhost:3306/mydb3"**);  
comboPooledDataSource.setUser(**"root"**);  
comboPooledDataSource.setPassword(**"1234"**);  
  
Connection connection = comboPooledDataSource.getConnection();  
System.***out***.print(connection);  
*//归还连接*connection.close();

### 配置文件

* 文件名称必须为：C3p0-config.xml
* 文件位置必须在SRC下
* 例如：



### 使用配置文件时的写法（重要）

* 默认配置

*/\*\*  
 \* 使用配置文件的默认配置  
 \* \*/*@Test  
**public void** fun2() **throws** Exception{  
 */\*  
 \* 在创建连接池对象时，这个对象就会自动加载配置文件！不用自己指定  
 \* \*/* ComboPooledDataSource comboPooledDataSource = **new** ComboPooledDataSource();  
 Connection connection = comboPooledDataSource.getConnection();  
 System.***out***.print(connection);  
 connection.close();  
}

* 命名配置

*/\*  
\* 使用配置文件的命名配置  
\* \*/*@Test  
**public void** fun3() **throws** Exception{  
 */\*  
 \* 构造器的参数指定命名配置元素的名称！  
 \* <named-config name="oracle-config">  
 \* \*/* ComboPooledDataSource comboPooledDataSource = **new** ComboPooledDataSource(**"oracle-config"**);  
 Connection connection = comboPooledDataSource.getConnection();  
 System.***out***.print(connection);  
 connection.close();  
}

## Tomcat配置连接池

### JNDI

* 简介

全称（Java Naming and Directory Interface），Java命名和目录接口。

* 作用

在服务器上配置资源，然后通过统一的方式来获取配置的资源。

### 配置资源

* 在Tomcat的config文件夹下的context.xml文件中添加：（服务器级别）

<Context>

<!--

name:指定资源的名称、

factory:资源由谁来负责创建

type:资源的类型

其他的东西都是资源的参数

-->

<Resource name="jdbc/dataSource"

factory="org.apache.naming.factory.BeanFactory"

type="com.mchange.v2.c3p0.ComboPooledDataSource"

jdbcUrl="jdbc:mysql://localhost:3306/mydb3"

driverClass="com.mysql.jdbc.Driver"

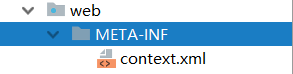
user="root"

password="1234"

/>

</Context>

* 或者（项目级别 优先配置此项）



内容同上。

### 获取资源



* 连接代码

**try** {  
 Context context = **new** InitialContext();  
 */\*  
 \*2.查询出入口  
 \* \*/* Context envContext = (Context) context.lookup(**"java:comp/env"**);  
 */\*  
 \* 3.再进行二次查找，找到我们的资源  
 \* `使用的名称与<Resource>元素的name对应  
 \* \*/* DataSource dataSource = (DataSource)envContext.lookup(**"jdbc/dataSource"**);  
 *//得到连接* Connection connection = dataSource.getConnection();  
 System.***out***.print(connection);  
 connection.close();  
} **catch** (Exception e) {  
 **throw new** RuntimeException(e);  
}

## ThreadLocal类

本地线程