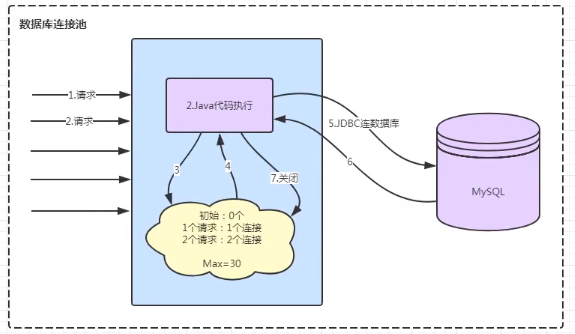
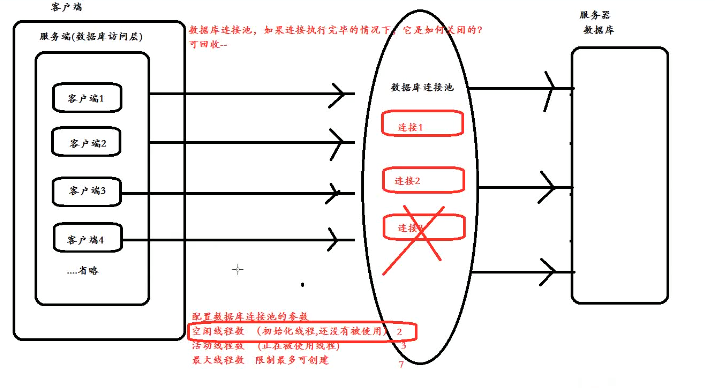
# 概述

数据库连接是一种昂贵的资源，创建数据库连接是一个耗时的操作，在多线程并发情况下尤为突出，对数据库连接的高效管理影响到程序的性能指标，数据库连接池正是针对该问题提出的。

数据库连接池负责分配、管理和释放数据库连接，它允许应用程序重复使用一个现有的数据库连接，而不是再重新建立一个新连接，利用数据库连接池将明显提高对数据库操作的性能。



# 原理



数据库连接池实现原理（实现代码角度）：

1. 定义核心参数：

空闲线程（容器）：没有被使用的连接

活动县城：正在使用的连接

1. 初始化线程池（初始化空闲线程）
2. 调用getConnection获取连接
3. 调用releaseConnection释放连接：

获取activeConnection集合连接，转移到freeConnection集合中

# 作用

1. 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。在减少系统性能消耗的基础上，增进了系统环境的平稳性（减少内存碎片以及数据库临时进程、线程的数量）。

1. 更快的访问速度

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内备用。此时，连接池的初始化操作均已完成。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的响应时间。

1. 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，实现数据库连接技术。

1. 统一的连接管理，避免数据库连接泄露

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，强制收回被占用的连接，从而避免了常规数据库连接操作中可能出现的资源泄露。

线程池好处：重复利用，提高响应效率，统一管理。

# 计算公式

<https://mp.weixin.qq.com/s/2JpPMauLq5aPkJTEMm6NuA>

下面的公式是由PostgreSQL提供的，不过我们认为可以广泛地应用于大多数数据库产品。你应该模拟预期的访问量，并从这一公式开始测试你的应用，寻找最合适的连接数值。

连接数 = ((核心数 \* 2) + 有效磁盘数)

核心数不应包含超线程(hyper thread)，即使打开了hyperthreading也是。如果活跃数据全部被缓存了，那么有效磁盘数是0，随着缓存命中率的下降，有效磁盘数逐渐趋近于实际的磁盘数。这一公式作用于SSD时的效果如何尚未有分析。按这个公式，你的4核i7数据库服务器的连接池大小应该为((4 \* 2) + 1) = 9。取个整就算是是10吧。是不是觉得太小了？跑个性能测试试一下，我们保证它能轻松搞定3000用户以6000TPS的速率并发执行简单查询的场景。如果连接池大小超过10，你会看到响应时长开始增加，TPS开始下降。扩展：[用了这么久的数据库连接池，你知道原理吗？](http://mp.weixin.qq.com/s?__biz=MzI4Njc5NjM1NQ==&mid=2247490668&idx=2&sn=229c7bf8df9a3750eeb68b4eeee38a8f&chksm=ebd62340dca1aa56678800efeeae3b54649bdbed8472cd1340f39b02bb20ddc262b8ae61cc69&scene=21" \l "wechat_redirect" \t "https://mp.weixin.qq.com/_blank)

注：  
 这一公式其实不仅适用于数据库连接池的计算，大部分涉及计算和I/O的程序，线程数的设置都可以参考这一公式。

# 实现

可以采用wait和notifyAll方式实现，也可以通过阻塞队列实现。

注：可以采用代理模式实现数据库连接池。

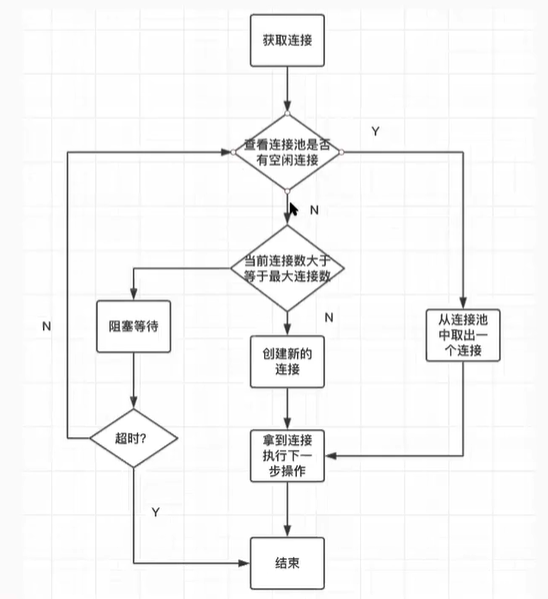
## 初始化连接池

配置数据库连接池的参数：

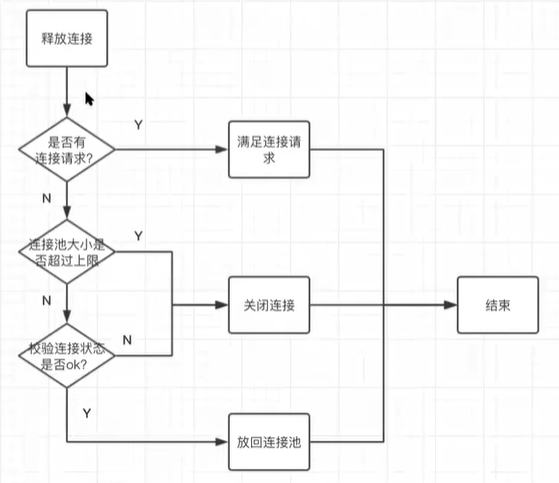
1. 空闲线程数（初始化线程，还没有被使用）
2. 活动线程数（正在被使用线程）
3. 最大线程数（限制最多可创建）



## 获取连接



## 释放连接



## 连接池关闭

# 技术方案

数据库连接池属于一种池化技术。

池化技术：**HTTP访问（httpclient）、redis访问（redispool）、线程（线程池）**。

## JDBC Template

## C3P0

步骤：

1. 导入jar包：c3p0-0.9.5.2.jar+mchange-commons-java-0.2.12.jar
2. 定义配置文件：

名称：c3p0.properties、c3p0-config.xml

路径：直接将文件放在src目录下即可

1. 创建核心对象：数据库连接池对象ComboPooledDataSource
2. 获取连接：getConnection

## DBCP

## Proxool

## Tomcat JDBC Pool

## BoneCP

## Druid

## HikariCP