1.系统性能瓶颈分析：

影响系统性能的因素：

* CPU：计算
* MEM：缓存
* 磁盘：持久化存储
* 网络：数据传输

2.不同的系统性能压力点不同

门户网站、文件服务、政企办公：网络带宽瓶颈、缓存多（偏静）

流媒体应用、科学计算：进程多、消耗内存多、磁盘IO频繁（偏动）

企业生产应用：DB压力大、存储压力大、内存压力大、CPU压力大（动&静）

3.系统架构层次分析：

**各层次性能的关注点：**

* 操作系统资源分配

　　　　　系统资源：服务器、客户机cpu、内存、硬盘等配置

　　　　　操作系统：操作系统资源分配

* 应用服务中间件配置

　　　　连接数等等

* 程序和数据库

　　　　程序：内存、线程的锁、或其他功能性的问题

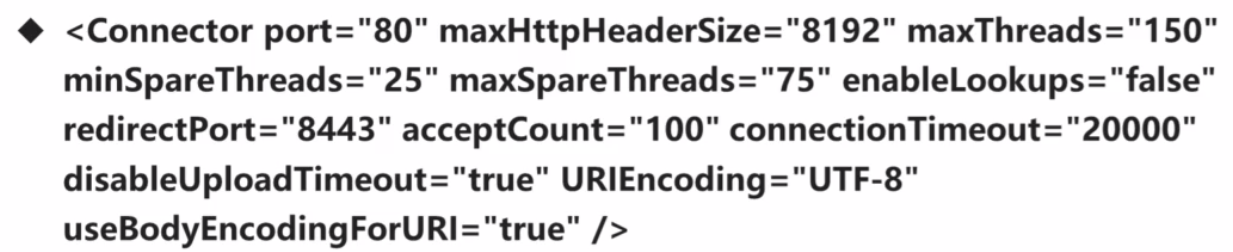
　　　　数据库：参数的配置

4.中间件Tomcat配置调优

Tomcat连接数配置实例：

A项目的server.xml的配置文件

sudo find / -name \*tomcat\*



相关配置参数解释：

* maxThreads:Tomcat使用线程来处理接收的每一个请求。这个值表示Tomcat可以创建的最大的线程数。默认值为200.可以根据机器的实际性能和内存大小调整，一般可以在400-500，这个数值决定最多同时处理的连接
* minSpareThreads="25"     表示即使没有用户也要开25个空线程等待。
* maxSpareThreads="75"    表示如果最多可以空75个线程，例如某时刻有80用户访问，之后没有用户访问了，则tomcat不会保留80个空线程，而是关闭5个空线程
* acceptCount  指定所有可以使用处理请求的线程数都被使用时，可以放到处理队列中的请求数，超过这个数，请求将不予处理。默认值10
* connectionTimeout    网络连接超时，默认值20000，单位：毫秒。设置为0表示永不超时，这样设置存在隐患。通常可设置为30000毫秒

遇到的问题：连接数配置不够，并发数超出最大连接数

* Loadrunner报错：Error -27791 Server "173.31.1.20" has shut down the connection prematurely

查看tomcat的日志（tomcat的安装目录下logs文件夹）文件catalina.-11-25.log,其中有以下错误信息：

　　1.-11-25 18:26:03 org.apache.tomcat.until.threads.ThreadPool logFull

　　2.严重：All threads(150) are currently busy, watting. Increase maxThreads(150) or check the servlet status.

5、jvm参数配置优化和代码优化

jvm参数配置

　　a. 设置环境变量： 变量名：CATALINA\_OPTS 变量值：-Xmx128m -Xms64m -Xmn32m -Xss16m

　　b.打开Tomcat根目录下的bin文件夹，编辑catalina.sh,将其中的%CATALINA\_OPTS%(共有四处)替换为：-Xmx128m -Xms64m -Xmn32m -Xss16m



最容易调优的地方

* 代码
* 最为成熟的部分主要开销是CPU
* 通过配置和应用平台调整可以实现

Java代码方面优化

　　1.减少new对象；用移位符号替代乘除号。

　　2.多使用局部变量，减少使用静态变量。

　　3.避免使用finalize,该方法会给GC增添很大负担；

　　4.如果是单线程，尽量使用非多线程安全的，因为线程安全来自于同步机制，同步机制会降低性能。

　　5.尽量使用基本类型而不是包装类型，尽量使用一维数组而不是二维数组。

　　6.尽量使用final修饰符,final表示不可修改，访问效率高。

　　7.单线程情况下，字符串尽量使用StringBuilder,比StringBuffer要快。

* 其实就是开发人员在写代码时处处都在考虑行问题；
* 这是一个成为高级程序员的必修之路，不在只考虑功能实现，还要考虑实现的效果如何

6、Mysql的监控和调优

系统SQL分析调优

优化方面：

　　主从复制

　　静动分开（比如百度就是这样的）

　　加数据库缓存

　　设计优化（数据库、表结构、标的设计上需要优化）

　　索引或视图减少查询内容

　　优化SQL结构（通过explan优化sql的结构）

Mysql配置优化

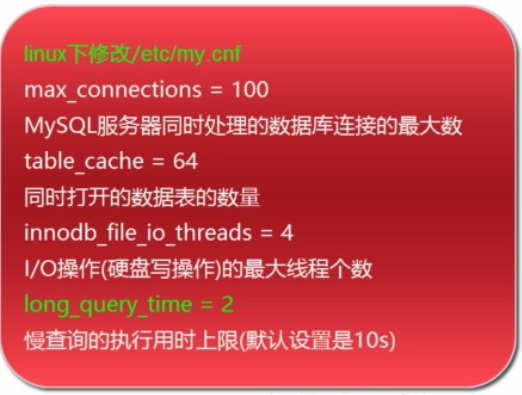
* 数据库连接数占满，事物失败
* 容器线程数占满（如tomcat）
* 连接池配置
* 设置慢查询

慢查询

　　分析出慢的原因

　　SQL本身

　　索引



前端调优-有效and难

通常前端的一些措施：

* 减少HTTP请求数据量及个数
* 使用CDN数据加速
* 合理使用Expires增加缓存效果
* 使用动态或静态压缩技术
* 在顶部加载CSS，JS放在页面底部，使用外部
* 使用AJAX请求可缓存
* 减少DNS查询

