性能调优整体思路

1. 空间换时间
   对热点数据缓存，减少数据查询时间。
2. 分而治之
   将大任务切片，分开执行。HDFS、MapReduce就是这个原理。
3. 异步处理
   若业务链中有某一环节耗时严重，则该环节将拉长业务链的整体耗时。可以将耗时业务采用消息队列异步化，从而缩短业务链耗时。
4. 并行处理
   采用多进程、多线程同时处理，提升处理速度。
5. 离用户更近一点
   如CDN技术，将静态资源放到离用户更近的地方，从而缩短请求静态资源的时间。
6. 提升可扩展性
   采用业务模块化、服务化的手段，提升系统的可扩展性，从而可根据业务需求实现弹性计算。

性能调优关注点

1. 硬件资源

1. CPU
   CPU使用率过高的原因：
   - 计算量大
   - 非空闲等待？
   - 过多的系统调用
   - 过多的中断
2. 内存
   内存使用率过高的原因：
   - 过多的页交换
   - 可能存在内存泄露
3. IO
   IO繁忙的原因：
   - 读写频繁
     磁盘的读写过程是物理动作，频繁的读写势必会使IO来不及处理。

4. 网络
   要关注服务器的出入口带宽。

2. 操作系统

1. 系统负载
   系统负载指的是CPU就绪队列中任务数。若该值超过6，则表示系统负载较高了。
2. 连接数
   操作系统处于安全考虑，会限制操作系统的最大TCP连接数，所以如果服务器需要提供大量服务的话，就需要修改TCP的最大连接数。

3. 数据库

4. 中间件

5. 应用程序

6. 前端优化

服务端架构演进

1. 单机结构

当系统访问量较小时，使用单机即可满足需求。所谓单机，即应用程序、数据库均放在一台服务器上完成。但单机的处理能力毕竟是有限的，当系统的访问量增加时，单机无法满足需求时，从而就出现了集群结构。

2. 集群结构

集群结构中，将数据库、应用程序分别放在多台服务器上，那么用户的请求究竟由哪台服务器处理呢？这就由负载均衡服务器来控制。

负载均衡服务器分为两种，分别是对TCP/IP的四层、七层进行负载分发。
四层负载分发常用的手段有：

- LVS：免费开源，性能不如F5。
- F5：它是个硬件交换机，很贵，但性能很高。

七层负载分发常用的手段有：

- Tengine
- Nginx
- ATS
- Vanish
- Squid
- Haproxy

集群结构基本能满足中小企业的业务需求，但它存在如下几个缺点：

1. 所有业务均在一个war包中，耦合度过高
2. 所有业务均在一个war包中，代码不易于维护

为了解决上述问题，因此出现了分布式结构。

3. 分布式结构

3.1 微服务

在分布式结构中，将业务进行服务化。
所谓服务化，就是将一个完整的应用，根据逻辑功能拆分成多个子应用，每个应用都有各自独立的war包，部署在不同的服务器上。
服务化有如下好处：

1. 系统逻辑清晰、耦合度低。
2. 可以根据服务的业务量合理分配计算资源。
3. 问题更容易排查。

3.2 分布式数据库

分布式数据库是对数据库进行分库分表，将数据分片存储在不同数据库的不同表中，并在数据库存储层之上增加了数据访问层，可通过hash找到数据所处的库与表。很多公司都基于Mariadb开发自己的分布式数据库，该数据库之上建立数据访问层，用来实现分库分表，并对上层透明。

3.3 注册中心 Zookeeper

注册中心用来管理所有的分布式服务。当A服务需要请求B服务时，A服务首先会从注册中心获取B服务的IP，然后向该IP发起请求。
此外，当服务不可用、或增加新的服务时，配置中心就会相应地在服务列表中删除、增加该项服务。