**性能测试**

# **性能测试目的**

验证软件系统是否能够达到用户指出的性能指标，同时发现软件系统中存在的性能瓶颈，优化软件，最后达到优化系统的目的。包括：

* 评估系统的能力
* 识别体系中的弱点
* 系统调优
* 验证稳定性和可靠性

# **压测环境**

保证压测环境和生产环境高度一致是执行一次有效性能压测的首要原则，压测环境和生产环境有很细微的差别，都有可能导致整个压测活动评测出来的结果不准确。

1. 低配测试环境-研发阶段发现性能瓶颈
2. 生产环境-全链路压测

# **服务器性能指标**

## 1、在线用户数

此指标指的是某个时间段内，在服务器上保持登录状态的用户数，在线用户数不等同于并发用户数，因为，只要用户不向服务器发送请求，就不会对服务器产生影响。

## 2、并发用户数

在性能测试过程中，一般用的比较多的指标是绝对并发用户数，我们也通常直接将绝对并发用户数直接称为并发用户数。

相对并发用户数：在某段时间内，与服务器保持交互的用户数。

绝对并发用户数：表示同一时间点（准确的说是在足够短的时间内）与服务器进行交互的用户数。

## 3、思考时间

每个用户操作后的暂停时间，也就是用户两次请求发起的间隔时间，这段时间是对服务器不产生压力的，在压测过程中，如果想要预估实际系统承受的最大压力，要尽可能的模拟用户真正的思考时间。

## 4、响应时间

响应时间，包含了网络传输请求时间、服务器处理时间、网络传输响应时间，其中，服务器处理时间通常受到代码处理请求的业务逻辑的影响，而且这个时间也是我们重点关心的一个时间，它能反映出我们代码中的缺陷和业务逻辑处理的优化点。而网络传输请求和响应时间，通常情况下取决于网络质量。

## 5、TP90、TP99、TP999

对于这个指标的定义为：满足90%、99%等的网络请求所需要的最低耗时。举例说明如下：

一个接口6次请求的耗时分别为：100ms、20ms、300ms、40ms、50ms、150ms。将这些时间从小到大排列为：20ms、30ms、40ms、100ms、150ms、300ms。那么TP99的计算过程为：在这6次请求中，取第6的99%(6\*0.99 = 5.94，近似于6)次的耗时做为tp99值，从小到大排序后，第6次的耗时为250ms，因此此接口6次请求的TP99的值就是250ms。一般，tp99不能增长太快，且要根据实际情况去确定期望的TP99值。

## 6、TPS

TPS通常又称为吞吐量，它指的是服务器每秒处理的事务数，是直接反应系统性能的指标，TPS越大，性能越好。

那么TPS和响应时间以及并发用户数之间是什么关系呢？

假如1个用户在1s时间内完成1笔事务，那么TPS=1；

如果某笔业务的响应时间是1ms，那么每秒处理的事务所为1000，也就是说TPS=1000；

如果1个用户在1s只能完成1笔事务，要想达到TPS=1000，那么并发用户量就需1要是1000；

不难得出这样的公式：

设并发数为c，响应时间为t（ms），那么TPS=(1000/t)\*c

从公式可以看出，要想提高系统的TPS，有两种方式：增加并发量或者降低响应时间。

降低响应时间只能通过优化代码的方式实现。

提高并发用户数c，而并发用户数通常又和服务器程序的请求处理模型关系密切，如果一个线程处理一个请求，最大并发用户数则取决于服务器能处理的最大线程数；同理，如果是一个进程对应一个请求，那么最大的并发用户数取决于最大进程数。需要说明的是，随着并发用户数的增加，响应时间往往也会随之增大。因此，最大并发用户数要从实际得出才有意义。

## 7、代理服务器

代理服务器是介于浏览器和Web服务器之间的一台服务器，有了它之后，浏览器不是直接到Web服务器去取回网页而是向代理服务器发出请求，Request信号会先送到代理服务器，由代理服务器来取回浏览器所需要的信息并传送给你。

# **前端性能测指标**

1、前端性能的意义

对于访问一个网站，最花费时间的并不是后端应用程序处理以及数据库等消耗的时间，而是前端花费的时间（包括请求、网络传输、页面加载、渲染等）。根据web优化的黄金法则：

80%的最终用户响应时间花在前端程序上，而其大部分时间则花在各种页面元素，如图像、样式表、脚本和Flash等，的下载上。减少页面元素将会减少HTTP请求次数。这是快速显示页面的关键所在。

2、定义

****web前端性能：****

即是web用户在访问一个页面时所要花费的时间总和。即一个完全意义上的用户响应时间，相对于服务器的响应时间而言还会包括更多的内容和影响因素

3、指标

首屏绘制（First Paint，FP）

首屏内容绘制（First Contentful Paint，FCP）

可交互时间（Time to Interactive，TTI）

最大内容绘制（Largest Contentful Paint，LCP)

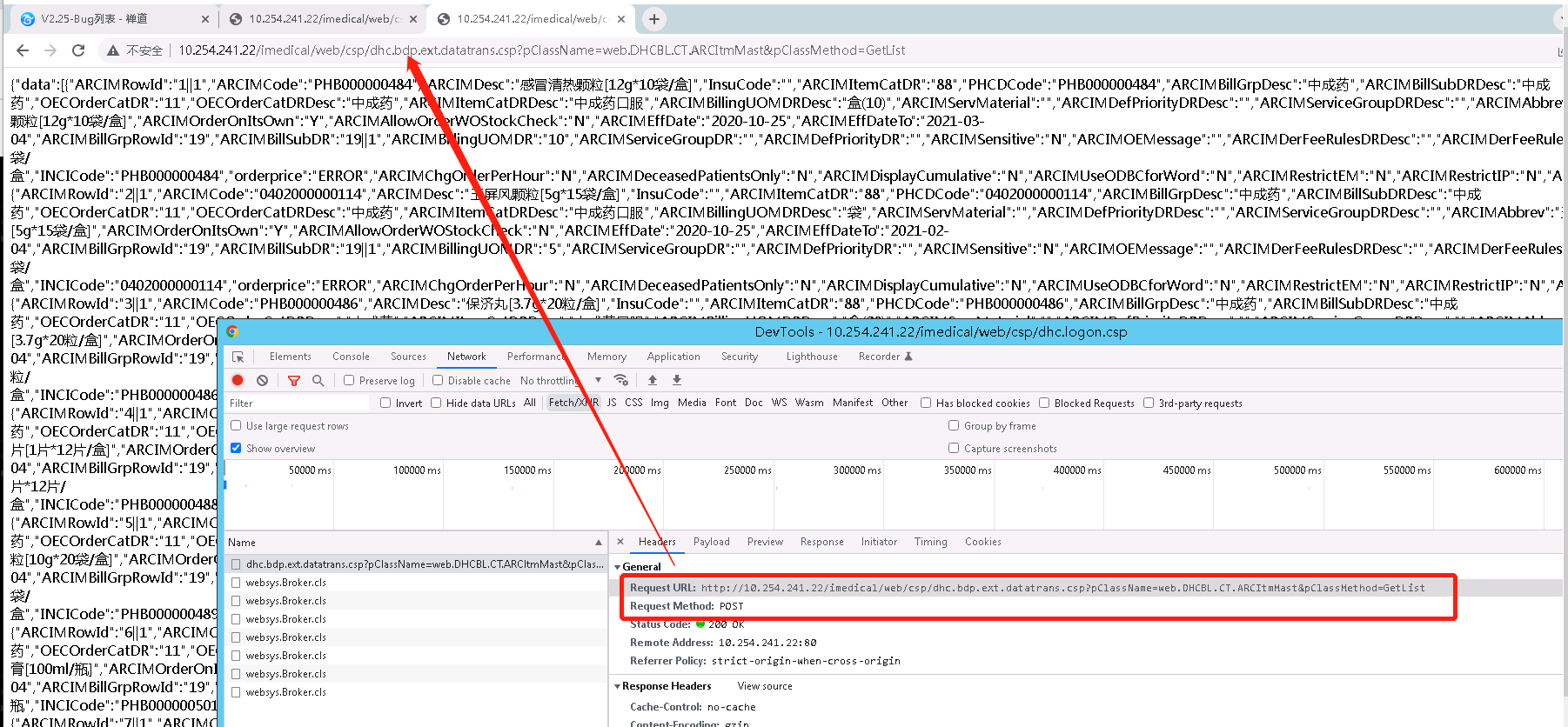
首次有效绘制（First Meaning Paint, FMP）

不同工具获取的指标会些许不同

# **测试响应时间方法**

1. web前端页面响应时间：F12开发工具（单机响应时间）
2. 小程序页面页面响应时间：perfdog
3. 压测（平均响应时间），响应时间跟并发量有关

如果将url输入浏览器，返回的是全部数据，因为这是个post请求，没有带参数，因此不带参数在浏览器访问，应该默认是返回全部数据。



# **降低响应时间**

**降低响应时间方法——优化代码**

1. 数据库角度

缓存数据库redis比关系型数据库mysql查询会更快，因为数据保存在内存，存取速度快，并发能力强；（内存属于内部存储设备）

redis应该是一种存储方法的集合，Redis 是以key-value形式存储：

Redis支持的数据类型：

String（字符串）

List（列表）

Set 求交集、并集

zset(sorted set：有序集合)

hash（哈希）

1. 数据表角度
2. 报表查询qury方法（查询数据并且过滤组成我想要的数据）：改写qury方法，跨表查询，尽量减少数据表个数去查，选择字段少的表格去查，查询时间会更快；
3. 设计简单表去查：重新设计一张数据表，将业务数据放入，查询时候，直接调用简单表的数据，而不必去调用复杂的多个业务表
4. 查表走历史数会比实时数据更快：历史数据：提前将数据准备好在相应表内，查询时候可以直接在表内调用；实时数据就是实时查询业务表，查询时间更久些；

# **测试方案**

测试方案：是描述被测对象需要测试的特性、测试的方法、测试环境的规划、[测试工具](https://so.csdn.net/so/search?q=%E6%B5%8B%E8%AF%95%E5%B7%A5%E5%85%B7&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/cuixin1108/article/details/_blank)的设计和选择、测试用例的设计方法、测试代码的设计方案。简言之，测试方案是从技术角度对整个测试活动进行规划和控制。

测试方案和测试计划的区别：

1. 测试计划是一个偏管理性质的文档，而测试方案是一个偏技术类型的文档；
2. 测试计划通俗来讲就是解决【谁来做？】【做什么？】的问题，而测试方案是解决【怎么做？】的问题；
3. 测试计划主要包含测试的目的与范围、角色与职责、资源及安排、风险与应对、测试标准等相关信息，而测试方案主要包含测试方法、测试环境与测试工具等内容。
4. 测试方案是从测试的角度去分析或者说分解需求，在方向上明确要怎么测，分析结果就是测试点和测试方法

# **性能测试方案**

1. 性能需求点分析

关键业务——产品定义的关键业务，需与客户沟通确认

日PV值较高的业务——通过运维数据得出调频功能点和时间分布（PV：页面浏览的人次计数））

逻辑复杂度高的业务——开发梳理逻辑复杂度高对应的功能点

运营主推的活动业务——当前公司主推的业务，特殊节日活动

用户反馈的业务——线上用户反馈的性能问题

利益相关者关注的功能点——老板特别关注的功能点，即便不是重要业务

合约明确规定的条款——验收时必须提供相关报告，甲方明确提出的内容

1. 确定测试目标

业务目标：响应时间、TPS、吞吐量、正确率、稳定性

1. 测试服务对象确认
2. 关键场景定义
3. 关键路径获取及全面覆盖

# **数据库性能测试- jemeter**

jmeter压测的核心原理就是：基于各种协议，通过多线程的方式来模拟并发用户，设计各种场景来模拟真实的用户负载

数据库性能测试

1. 配置链接数据库
2. 设置线程组
3. 添加数据库请求
4. 监听器观察