boolean，true /false

char，**\u0000（0）～** **\uffff（**65,535**）**

byte，**-128～127 默认值：0**

short，**-32768～32767 默认值：0**

int，**-2,147,483,648～2,147,483,647 默认值：0**

long，**-9,223,372,036,854,775,808～9,223,372,036,854,775,807 默认值：0L**

float，默认值：**0.0f**

double默认值： **0.0d**

**public** **class** Test03 {

**public** **static** **void** main(String[] args) {

        Integer f1 = 100, f2 = 100, f3 = 150, f4 = 150;

        System.out.println(f1 == f2);

        System.out.println(f3 == f4);

    }

}

如果不明就里很容易认为两个输出要么都是true要么都是false。首先需要注意的是f1、f2、f3、f4四个变量都是Integer对象引用，所以下面的==运算比较的不是值而是引用。装箱的本质是什么呢？当我们给一个Integer对象赋一个int值的时候，会调用Integer类的静态方法valueOf，如果看看valueOf的源代码就知道发生了什么。

**public** **static** Integer valueOf(**int** i) {

**if** (i >= IntegerCache.low && i <= IntegerCache.high)

**return** IntegerCache.cache[i + (-IntegerCache.low)];

**return** **new** Integer(i);

}

IntegerCache是Integer的内部类，其代码如下所示：

**private** **static** **class** IntegerCache {

**static** **final** **int** low = -128;

**static** **final** **int** high;

**static** **final** Integer cache[];

**static** {

            // high value may be configured by property

**int** h = 127;

            String integerCacheHighPropValue =

                sun.misc.VM.getSavedProperty("java.lang.Integer.IntegerCache.high");

**if** (integerCacheHighPropValue != **null**) {

**try** {

**int** i = parseInt(integerCacheHighPropValue);

                    i = Math.max(i, 127);

                    // Maximum array size is Integer.MAX\_VALUE

                    h = Math.min(i, Integer.MAX\_VALUE - (-low) -1);

                } **catch**( NumberFormatException nfe) {

                    // If the property cannot be parsed into an int, ignore it.

                }

            }

            high = h;

            cache = **new** Integer[(high - low) + 1];

**int** j = low;

**for**(**int** k = 0; k < cache.length; k++)

                cache[k] = **new** Integer(j++);

            // range [-128, 127] must be interned (JLS7 5.1.7)

**assert** IntegerCache.high >= 127;

        }

**private** IntegerCache() {}

    }

简单的说，如果整型字面量的值在-128到127之间，那么不会new新的Integer对象，而是直接引用常量池中的Integer对象，所以上面的面试题中f1==f2的结果是true，而f3==f4的结果是false。

## Cross-Site Scripting(跨脚本攻击)

跨站脚本（cross site script）为了避免与样式css混淆，所以简称为XSS。

XSS是一种经常出现在web应用中的计算机安全漏洞，也是web中最主流的攻击方式。那么什么是XSS呢？

XSS是指恶意攻击者**利用**网站没有对用户提交数据进行转义处理或者过滤不足的**缺点**，进而添加一些代码，嵌入到web页面中去。使别的用户访问都会执行相应的嵌入代码。

从而盗取用户资料、利用用户身份进行某种动作或者对访问者进行病毒侵害的一种攻击方式。

**XSS攻击的危害包括：**

1、盗取各类用户帐号，如机器登录帐号、用户网银帐号、各类管理员帐号

2、控制企业数据，包括读取、篡改、添加、删除企业敏感数据的能力

3、盗窃企业重要的具有商业价值的资料

4、非法转账

5、强制发送电子邮件

6、网站挂马

7、控制受害者机器向其它网站发起攻击

### 原因解析

**主要原因：**过于信任客户端提交的数据！

**解决办法：**不信任任何客户端提交的数据，只要是客户端提交的数据就应该先进行相应的过滤处理然后方可进行下一步的操作。

**进一步分析细节：**

　　客户端提交的数据本来就是应用所需要的，但是恶意攻击者利用网站对客户端提交数据的信任，在数据中插入一些符号以及javascript代码，那么这些数据将会成为应用代码中的一部分了。那么攻击者就可以肆无忌惮地展开攻击啦。

　　因此我们绝**不可以信任**任何客户端提交的数据！！！

#### XSS攻击类型

#### 反射型xss攻击

　又称为非持久性跨站点脚本攻击，它是最常见的类型的XSS。漏洞产生的原因是攻击者注入的数据反映在响应中。一个典型的非持久性XSS包含一个带XSS攻击向量的链接(即每次攻击需要用户的点击)

**正常发送消息：**

http://www.test.com/messag?send=Hello,World！

接收者将会接收信息并显示Hello,Word

非正常发送消息：

http://www.test.com/message?send=<script>alert(‘foolish!’)</script>！

接收者接收消息显示的时候将会弹出警告窗口

#### 储存型xss攻击

又称为持久型跨站点脚本，它一般发生在XSS攻击向量(一般指XSS攻击代码)存储在网站数据库，当一个页面被用户打开的时候执行。每当用户打开浏览器,脚本执行。持久的XSS相比非持久性XSS攻击危害性更大,因为每当用户打开页面，查看内容时脚本将自动执行。谷歌的orkut曾经就遭受到XSS。

**简单例子：**

从名字就可了解到存储型XSS攻击就是将攻击代码存入数据库中，然后客户端打开时就执行这些攻击代码。例如留言板

留言板表单中的表单域：<input type=“text” name=“content” value=“这里是用户填写的数据”>

**正常操作：**

用户是提交相应留言信息；将数据存储到数据库；其他用户访问留言板，应用去数据并显示。

**非正常操作：**

攻击者在value填写<script>alert(‘foolish!’)</script>【或者html其他标签（破坏样式。。。）、一段攻击型代码】；

将数据存储到数据库中；

其他用户取出数据显示的时候，将会执行这些攻击性代码

## 磁盘性能指标 –IOPS理论

**为什么机械硬盘没有固态硬盘效率高**？

机械硬盘的连续读写性很好， 但随机读写性能很差。这是因为磁头移动至正确的磁道上需要时间，随机读写时，磁头不停的移动，时间都花在了磁头寻道上，并且日常生活中读写90%都是随机读写，所以性能不高

在存储小文件(图片)、OLTP数据库应用时，随机读写性能（IOPS）是最重要指标。

固态硬盘iops以万为单位，而机械硬盘只有几百

磁盘性能指标--IOPS  
----------------------------------------------------------  
        IOPS (Input/Output Per Second)即每秒的输入输出量(或读写次数)，是衡量磁盘性能的主要指标之一。IOPS是指单位时间内系统能处理的I/O请求数量，一般以每秒处理的I/O请求数量为单位，I/O请求通常为读或写数据操作请求。  
  
    随机读写频繁的应用，如小文件存储(图片)、OLTP数据库、邮件服务器，关注随机读写性能，IOPS是关键衡量指标。  
  
    顺序读写频繁的应用，传输大量连续数据，如电视台的视频编辑，视频点播VOD(Video On Demand)，关注连续读写性能。数据吞吐量是关键衡量指标。  
  
IOPS和数据吞吐量适用于不同的场合：  
读取10000个1KB文件，用时10秒  Throught(吞吐量)=1MB/s ，IOPS=1000  追求IOPS  
读取1个10MB文件，用时0.2秒  Throught(吞吐量)=50MB/s, IOPS=5  追求吞吐量  
  
**磁盘服务时间**  
--------------------------------------  
传统磁盘本质上一种机械装置，如FC, SAS, SATA磁盘，转速通常为5400/7200/10K/15K rpm不等。影响磁盘的关键因素是磁盘服务时间，即磁盘完成一个I/O请求所花费的时间，它由寻道时间、旋转延迟和数据传输时间三部分构成。  
  
**寻道时间** Tseek是指将读写磁头移动至正确的磁道上所需要的时间。寻道时间越短，I/O操作越快，目前磁盘的平均寻道时间一般在3－15ms。  
**旋转延迟** Trotation是指盘片旋转将请求数据所在扇区移至读写磁头下方所需要的时间。旋转延迟取决于磁盘转速，通常使用磁盘旋转一周所需时间的1/2表示。比如，7200 rpm的磁盘平均旋转延迟大约为60\*1000/7200/2 = 4.17ms，而转速为15000 rpm的磁盘其平均旋转延迟为2ms。  
**数据传输时间** Ttransfer是指完成传输所请求的数据所需要的时间，它取决于数据传输率，其值等于数据大小除以数据传输率。目前IDE/ATA能达到133MB/s，SATA II可达到300MB/s的接口数据传输率，数据传输时间通常远小于前两部分消耗时间。简单计算时可忽略。

**常见磁盘平均物理寻道时间为：**  
7200转/分的STAT硬盘平均物理寻道时间是9ms  
10000转/分的STAT硬盘平均物理寻道时间是6ms  
15000转/分的SAS硬盘平均物理寻道时间是4ms

常见硬盘的旋转延迟时间为：

7200   rpm的磁盘平均旋转延迟大约为60\*1000/7200/2 = 4.17ms

10000 rpm的磁盘平均旋转延迟大约为60\*1000/10000/2 = 3ms，

15000 rpm的磁盘其平均旋转延迟约为60\*1000/15000/2 = 2ms。

最大IOPS的理论计算方法  
--------------------------------------  
IOPS = 1000 ms/ (寻道时间 + 旋转延迟)。可以忽略数据传输时间。  
  
7200   rpm的磁盘 IOPS = 1000 / (9 + 4.17)  = 76 IOPS（每秒读写76次）  
10000 rpm的磁盘IOPS = 1000 / (6+ 3) = 111 IOPS  
15000 rpm的磁盘IOPS = 1000 / (4 + 2) = 166 IOPS

## Mysql导入大批量数据问题

#### 错误信息：

MySQL server has gone away

解决方法：

官方解释是适当增大 max\_allowed\_packet 参数可以使client端到server端传递大数据时，系统能够分配更多的扩展内存来处理

show global variables like 'max\_allowed\_packet'//查询当允许大小

set global max\_allowed\_packet=268435456设置256M

#### 衍生问题

公司机器打无法进行服务的启动或者关闭出现提示无法操作，拒绝访问

解决方法一、

重新删除服务，然后所有数据表进行导入操作，前提是有备份，或者了解MySQL bin中的data文件可以进行数据转移。

1.配置环境变量

我的电脑->属性->高级->环境变量->path

如:C:\Program Files\MySQL\MySQL Server 5.6\bin

注意是追加，不要覆盖

2.修改my-default.ini

在其中修改或添加配置：

[mysqld]

basedir=C:\Program Files\MySQL\MySQL Server 5.6（mysql所在目录）

datadir=C:\Program Files\MySQL\MySQL Server 5.6\data （mysql所在目录\data）

3.以管理员身份运行cmd（win10右键左下角开始按钮选择以管理员身份运行cmd即可）

以管理员身份运行cmd（一定要用管理员身份运行，不然权限不够），

输入：cd C:\Program Files\MySQL\MySQL Server 5.6\bin 进入mysql的bin文件夹(不管有没有配置过环境变量，也要进入bin文件夹，否则之后启动服务仍然会报错误2)

输入mysqld -install(如果不用管理员身份运行，将会因为权限不够而出现错误：Install/Remove of the Service Denied!)

安装成功

4.运行mysqld  --initialize（标题问题所在，若没有init则不存在data目录，自然无法启动成功）

5.安装成功后就要启动服务了，继续在cmd中输入:net start mysql,服务启动成功！

服务启动成功之后，就可以登录了，输入mysql -u root -p（第一次登录没有密码，直接按回车过）,登录成功！

追加内容：

在安装mysql5.7版本时，经常会遇到mysql -u root -p直接回车登陆不上的情况，原因在于5.7版本在安装时自动给了一个随机密码，坑爹的是在init步骤的时候不像linux系统会给出命令行提示，需要手动在mysql目录下搜索\*.err，以文本形式打开才能看到如下内容：

016-02-25T15:09:43.033062Z 1 [Note] A temporary password is generated for root@localhost: >mso<k70mrWe

第二种解决方法：

当我回想第一种方法的时候恍然发现命令行需要使用管理员权限，于是我再次尝试，使用管理员权限进行启动，发现可以启动…..

以管理员身份运行cmd

Net start mysql

MySQL正常启动

## QPS(query per second)每秒查询率

30000qps 如何搭建：

1. 首先要解决掉数据库的压力，3万qps对应的磁盘 iops 很大，不过现在好的 SSD 能提供很好的 iops, 比如这款： [ARK | Intel® SSD DC P3700 Series (800GB, 2.5in PCIe 3.0, 20nm, MLC)](http://link.zhihu.com/?target=http%3A//ark.intel.com/products/79627/Intel-SSD-DC-P3700-Series-800GB-2_5in-PCIe-3_0-20nm-MLC) 单盘 90000 IOPS，应该能撑住你的数据库，考虑到主备，以及你的sharding需求，3-9 台数据库机器，高内存，高CPU，SSD磁盘应该能抗住  
     
   2. 业务逻辑这一层: Java 系，用线程来抗并发的，如果业务逻辑不太复杂，那么基本能做到 100ms 内响应，那么 30000qps, 对应的是 3000并发线程，这部分设计的时候记得保持无状态，单台支撑 300-1000 并发没问题，加上一倍的冗余，那么 6~20 台业务型机器可以抗住。  
     
   3. 缓存层: 支付订单一般对缓存需求不高，但缓存层一般都会有，避免把查询压力压到数据库，简单两台缓存，或者缓存平行部署在业务型机器上都可以解决，具体看你的情况了。  
     
   4. 接入层: nginx 做LVS就可以了，记得 backlog 配大点就可以了, 3万qps, 假设单个请求的数据在 10KB 左右，那么是 300MB/s，如果是千兆机，每台4网卡，两内两外，加上冗余，我会部署4台入口机，如果是万兆机，两台做主备（心跳或者LVS)即可。

## TPS（每秒处理事务数）

TPS：Transactions Per Second（每秒传输的事物处理个数），即服务器每秒处理的事务数。TPS包括一条消息入和一条消息出，加上一次用户数据库访问。（业务TPS = CAPS × 每个呼叫平均TPS）

**技术要点**

TPS是软件测试结果的测量单位。**一个事务是指一个客户机向服务器发送请求然后服务器做出反应的过程。客户机在发送请求时开始计时，收到服务器响应后结束计时，以此来计算使用的时间和完成的事务个数。**

一般的，评价系统性能均以每秒钟完成的技术交易的数量来衡量。系统整体处理能力取决于处理能力最低模块的TPS值。