

面

试

宝

典

序言

传智播客和黑马程序员

江苏传智播客教育科技股份有限公司（简称传智播客）是一家专门致力于高素质软件开发人才培养的高科技公司，黑马程序员是传智播客旗下高端 IT 教育品牌。

黑马程序员的学员多为大学毕业后，有理想、想从事 IT 行业，而没有机遇改变自己命运的年轻人。黑马程序员的学员筛选制度，远比现在 90%以上的企业招聘流程更为严格，这些流程中不仅包括严格的技术测试、自学能力测试，还包括性格测试、压力测试、品德测试等测试。毫不夸张的说：黑马程序员所有学员都是精挑细选出来的。百里挑一的残酷筛选制度确保学生质量，并降低企业的用人风险。

自黑马程序员成立以来，教学研发团队也一直致力于打造精品课程资源，不断在产学研层面创新自己的执教理念与教学方针，并依托黑马程序员的优势力量，针对性地出版了计算机系列教材 30 多册、配套教学视频数十套、发表各类技术文章数百篇。

为何出版本书

随着移动互联网的发展，Java 开发求职者越来越多，作为一名传智播客就业服务部 Java 方向就业指导人员，面对如此激烈的市场竞争，该如何指导学员给面试官留下深刻印象，从众多的求职者中脱颖而出呢？针对这个问题，2017 年 7 月，传智播客顺义校区就业服务部提出想要出版一本可以帮助在校学生和 Java 程序员面试的面试宝典。

传智播客·黑马程序员从成立以来，开设的第一门学科，全国 16 所校区为社会培养了数以万计的 Java 技术人才。在这个过程中，传智播客顺义校区就业服务部就业老师根据市场需求和学员面试的反馈搜集了近 6 年来的大部分 Java 经典面试题。如果把这些面试题按照面试流程与难易程度进行系统性的分类、整理，则可以整理出一本经典的面试宝典。

如何在短时间内获得面试官的青睐是所有求职者的一个疑问，而本书则详细回答了这个问题。本书搜集了 Java 面试中遇到的经典面试题，并对每道题都编写了较为详细的答案说明。除此之外，本书对绝大多数 Java 开发求职者有极大地阅读价值，它能帮助求职者梳理核心技能点，让求职者在面试过程中胸有成竹。

本书的指导作用

- (1) 利用最短的时间使 Java 开发求职者对看似熟悉、实则陌生的企业环境有比较全面、清晰的认知。本书将在求职者与企业需求之间发挥重要的桥梁纽带作用。
- (2) 帮助求职者快速补充自身欠缺的谋职、求职、任职等各项实务知识，学以致用，用之有效。
- (3) 指导求职者在准备面试的过程中需要准备什么，准备到什么时候，有备则无患。

(4) 最大限度地避免求职者在职业发展道路上的失误、碰壁、走弯路，让求职者把宝贵的时间用在刀刃上。

(5) 可以有效的帮助学生节省用于摸索面试经验上的时间。

(6) 促使求职者成为企业乐于接纳、培养的实用性人才。

本书的指导宗旨

本书将以“知道面试，智取 Offer”为宗旨，为广大 Java 开发求职者扫清面试道路上的障碍，成为面试官眼中的精英，朋友圈里的大神。在面试场上“胸有成竹”，坦然面对每个面试官的“拷问”，做到进可攻“项目经理、项目总监”等高级职务，视之为翘首可及；退可守“Java 工程师、Java 测试工程师”等职务，视之为探囊取物。无论进退、皆可立于不败之地。

附言

所有准备面试或正在面试的求职者一定要清楚：不是工作难找，而是大家没有为找工作做好充分准备，一到关键时刻，就叹息“读书用时方恨少”。企业用人向来是宁缺毋滥，它绝不会聘用不合格的人。所谓知己知彼，百战不殆；大家一定要懂得不打无准备之战的道理。因此，准备面试或者正在面试的求职者自身不能做到未雨绸缪、运筹帷幄是非常不可取的，也是极不明智的。不能先发制人，结果只能是后发制于人，为何还怨天尤人？

黑马程序员 顺义校区就业服务部

2017 年 7 月

第一部分：基础知识.....	13
1、常用的集合有哪些，它们之间的关系是怎样的?.....	13
2、java 中形参和实参是什么意思?.....	13
3、list 集合底层是什么?长度是多少?内存扩展机制?.....	13
4、“==”和 equals 方法究竟有什么区别?.....	14
5、Java 中的方法覆盖 (Overwrite) 和方法重载 (Overloading) 是什么意思?.....	14
6、sleep() 和 wait() 有什么区别?.....	14
7、使用 final 关键字修饰一个变量时，是引用不能变，还是引用的对象不能变?.....	14
8、throws 与 throw 的区别?.....	14
9、Collection 和 Collections 的区别?.....	15
10、final, finally, finalize 的区别?.....	15
11、什么是哈希表?.....	15
12、数组有没有 length()方法? String 有没有 length()方法?.....	15
13、List、Map、Set 三个接口，存取元素时，各有什么特点?.....	15
14、遇到过的异常?.....	16
第二部分：多线程部分.....	17
1、什么是线程?说一下多线程的好处?.....	17
2、进程和线程的区别是什么?.....	17
3、如何在 Java 中实现线程?.....	17
4、那么你平时是用 Runnable 还是 Thread?.....	17
5、创建线程有几种不同的方式?你喜欢哪一种?为什么?.....	17
6、如何确保 N 个线程可以访问 N 个资源同时又不导致死锁?.....	18
第三部分：集合相关面试题.....	18
1、你用过 HashMap 吗?“什么是 HashMap?你为什么用到它?.....	18
2、那么你知道 HashMap 的工作原理吗?“你知道 HashMap 的 get()方法的工作原理吗?.....	18

3、当两个对象的 hashCode 相同会发生什么？.....	18
4、如果两个键的 hashCode 相同，你如何获取值对象？.....	19
5、hashmap 的底层架构是什么？.....	19
6、hashmap 和 hashtable 的区别？.....	19
7、HashSet 和 TreeSet 有什么区别？.....	20
8、如何遍历一个 List？.....	20
第四部分：Http 部分.....	20
1、说一下什么是 Http 协议？它有什么组成？（大型互联网公司公司会问）.....	20
2、Http 协议中有常用的请求方式你知道有哪些么？.....	20
3、对网络通信协议熟悉吗？对 socket 和 http 了解吗？能解释下吗？.....	21
4、说一下 Http 协议中 302 状态(阿里经常问).....	21
5、Http 协议有那些特征？.....	21
第五部分：IO 部分.....	21
1、Java 中有几种类型的流？.....	21
2、io 流怎样读取文件的？说下流程.....	22
3、String,StringBuffer 和 StringBuilder 之间的区别？.....	22
4、解释一下 java.io.Serializable 接口？.....	22
第六部分：JavaWeb.....	22
1、JSP 中动态 include 与静态 include 的区别？.....	22
2、两种跳转方式是什么？有什么区别？.....	22
4、jsp 和 servlet 的区别、共同点、各自应用的范围？？.....	23
5、cookie 和 session 的区别？.....	23
6、Tomcat 的优化经验？.....	23
7、request.getParameter()和 request.getAttribute()的区别？.....	23
8、如果 JSP 表单元素的值为空，如何避免 null 出现在页面上？.....	24
9、你在项目中用到了 XML 技术的哪些方面？如何实现的？.....	24
第七部份：数据库.....	24

一：关系型数据库.....	24
1、Mysql 部分.....	24
1.1 架构图介绍.....	24
1.1.1 连接管理与安全验证是什么？.....	25
1.1.2 解析器是什么？.....	25
1.1.3 优化器怎么用？.....	25
1.1.4 执行器是什么？.....	25
1.2、存储引擎都有哪些？.....	25
1) InnoDB 存储引擎.....	25
2) MyISAM 存储引擎.....	25
3) MEMORY 存储引擎.....	25
1.3、事务介绍.....	26
1.4、创建存储过程怎么写？.....	27
1.5、触发器怎么写？.....	27
1.6、内连接,左外连接,右外连接,全外连接,交叉连接?.....	28
2、Oracle.....	29
2.1、Oracle 内存结构介绍.....	29
2.2、数据结构是什么样子？.....	30
2.3、oracle 数据库之事务有哪些？.....	31
2.4、Oracle 创建存储过程怎么写？.....	32
2.5、如何使用 Oracle 的游标？.....	32
2.6、Oracle 中字符串用什么符号链接？.....	32
2.7、Oracle 是怎样分页的？.....	33
2.8、什么是存储过程？它有什么优点？.....	33
2.9、存储过程和函数有什么区别？.....	33
2.10 、Oracle 中的函数与存储过程的特点:.....	33
2.11、mysql, oracle, sql server 三者的区别？.....	34

二：非关系型数据库.....	35
1、Redis.....	35
1.1、什么是 redis?.....	35
1.2 、Reids 的特点？.....	35
1.3、 为什么 redis 需要把所有数据放到内存中？.....	36
1.4、 Redis 常见的性能问题都有哪些？如何解决？.....	36
1.5、redis 最适合的场景有哪些？.....	36
2、Memcache 与 Redis 的区别都有哪些？.....	37
三：数据库语句优化.....	37
1、Mysql sql 语句优化：.....	37
2、oracle sql 语句优化有哪些？.....	38
四：数据库优化：.....	39
1. Mysql 数据库优化有哪些？.....	39
2、Oracle 数据库优化有哪些？.....	40
第八部份：框架.....	40
一：Hibernate.....	40
1、Hibernate 和 JDBC 优缺点对比.....	40
2、关于 Hibernate 的 orm 思想你了解多少？.....	41
3、Hibernate 的开发流程是怎么样的？.....	41
4、谈谈 Hibernate 里边持久态对象的三种状态。.....	41
5、Hibernate 缓存机制是怎样的？谈谈 Hibernate 的一级缓存和二级缓存.....	41
6、Hibernate 有哪几种查询数据的方式.....	42
7、get 和 load 区别？.....	43
8、如何优化 Hibernate.....	43
二：Spring.....	44
1、谈谈你对 spring 的了解谈谈及你对 spring 里边 DI， AOP， IOC 认识。.....	44
1. 谈谈你理解的 spring 的工作流程的理解？.....	45

2. 说说 spring 对象创建的三种方式。.....	45
3. 不使用注解的情况下，如果给对象注入值的话，你知道的有几种方式？常用的是哪些？.....	45
4. Spring 有注解方式和 xml 配置两种，两者有什么区别？你常用 spring 注解有哪些？.....	46
5. Spring 里面 applicationContext.xml 文件能不能改成其他文件名？.....	46
6. 谈谈 spring 的事务管理.....	47
三：Struts2.....	47
1、谈谈你对 struts2 的理解。.....	47
2、Struts2 的配置流程。.....	48
3、struts 里边的 Action 配置的注意事项都有哪些？.....	48
4、拦截器和过滤器有什么区别？.....	48
5、Struts2 封装获取表单数据的方式有几种？谈谈你熟悉的那种。.....	49
6、struts2 里边的值栈你是如何使用的？如何向值栈中放值，如何从值栈中取值？... ..	49
7、Struts2 里边的 OGNL 表达式里边的#、%分别是做什么的？有什么区别？.....	50
8、你在开发中，值栈主要有哪些应用？.....	51
9、你会使用 struts2 自定义拦截器么？谈谈其中的原理。.....	51
四：SpringMVC.....	52
1、SpringMVC 的执行流程是什么？请简单阐述一下。.....	52
2、你常用的 SpringMVC 注解有哪些？如何开启 SpringMVC 的注解扫描？.....	52
3、如何开启注解处理器和适配器的配置？.....	53
4、使用 springMVC 框架的时候，如何解决 post 和 get 的乱码问题？.....	53
5、谈谈你对 restful 的理解及项目中的应用。.....	53
6、SpringMVC 与 struts2 的区别？.....	53
五：Mybatis.....	54
1、MyBatis 编程步骤是什么样的？.....	54
2、Mybatis 和 Hibernate 的区别。.....	54

3、JDBC 编程有哪些不足之处，MyBatis 是如何解决这些问题的？	54
4、简单的说一下 MyBatis 的一级缓存和二级缓存？	55
5、使用 MyBatis 的 mapper 接口调用时有哪些要求？	55
第九部份：传统项目	55
1、PowerDesigner (p2p,ERP,BOS,医药采购,杰信)	55
2、plsql (p2p,ERP,BOS,医药采购,杰信)	56
3、AngularJS (p2p)	56
应用场景：	56
4、Spring data jpa (p2p, 医药, Bos)	56
应用场景：	57
5、Bootstrap (p2p)	57
6、Redis.....	57
应用场景：	58
7、短信验证 (p2p、杰信)	58
8、ActiveMQ (p2p)	58
应用场景：	58
9、Webservice.....	58
应用场景：	59
10、Shiro.....	59
应用场景：	59
11、Easyui (Bos、ERP)	59
应用场景：	60
12、Poi (p2p、BOS、ERP)	60
应用场景：	60
13、MD5.....	60
应用场景：	60
14、JavaMail (p2p、杰信)	61

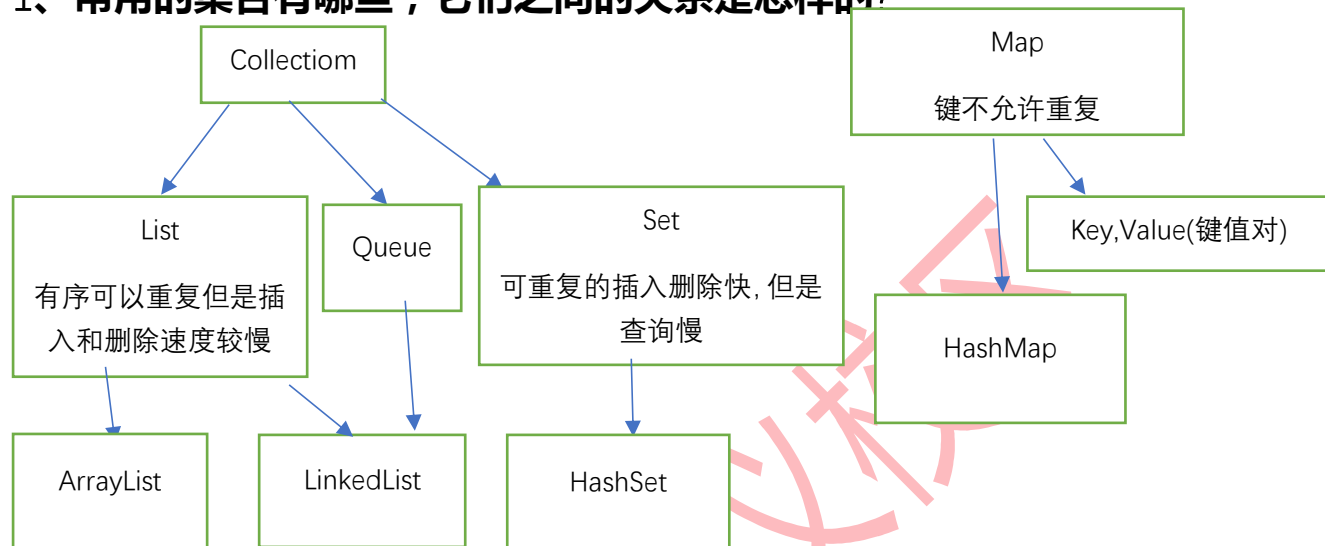
应用场景：.....	61
15、Quartz（杰信）.....	61
应用场景：.....	61
16、Amchart（杰信）.....	61
应用场景：.....	62
17、FreeMarker.....	62
18、Maven（p2p,ERP,BOS,医药采购,杰信）.....	62
19、分布式开发：.....	62
20、SVN（p2p,BOS,医药采购）.....	63
21、Git（BOS）.....	63
第十部份：电商项目.....	63
1、Dubbo.....	63
1. Dubbo 是什么？.....	63
2. Dubbo 能做什么？.....	64
3. Dubbo 服务开发流程，运行流程？Zookeeper 注册中心的作用？.....	64
2、FastDFS.....	64
1. 什么是 FastDFS？.....	64
2. FastDFS 架构.....	64
3. 文件上传的流程.....	66
4. 文件下载.....	67
5. 最简单的 FastDFS 架构.....	67
3、富文本编辑器.....	68
1. KindEditor：.....	68
2. UEditor：百度编辑器.....	68
4、Redis.....	69
1. Redis 的优点？.....	69
2. Redis 的缺点？.....	70

3. Redis 持久化.....	70
3.1、RDB 持久化 :	70
3.2、AOF 持久化 :	70
3.3、无持久化 :	70
3.4、同时应用 AOF 和 RDB.....	70
4、Redis 集群.....	70
5、Memcached 是什么 ?	71
5、MQ.....	71
1. RabbitMQ :	71
RabbitMQ 的优点 (适用范围)	71
2. ActiveMQ.....	72
1. 什么是 ActiveMQ ?	72
2. ActiveMQ 的消息形式.....	72
3. ActiveMQ 的作用、原理 ? (生产者。消费者。 p2p、订阅实现流程)	73
4. ActiveMQ 在项目中应用场景 ?	73
5. ActiveMQ 如果数据提交不成功怎么办 ?	73
6、Solr.....	74
1. Solr 是什么.....	74
特点 :	74
工作方式 :	74
2. Solr 里面的 IK 分词器的原理.....	75
3. Solr 怎么设置搜索结果排名靠前 (得分) ?	75
7、Zookeeper (淘淘商城)	75
1. Zookeeper 是什么 ?	75
2. 原理 :	76
8、Freemarker.....	76
1. 什么是 freemarker ?	76

2. Freemarker 的使用方法.....	76
3. 网页的静态化方案.....	77
9、Nginx.....	78
1. Nginx 是什么？.....	78
2. 应用场景.....	78
3. Nginx 与 Apache 的对比.....	79
1.1、nginx 相对于 apache 的优点：	79
1.4 nginx 处理动态请求.....	79
4. Nginx 反向代理为什么可以提高网站性能？.....	80
10、HttpClient.....	80
1. HttpClient 是什么？.....	80
2. HttpClient 的使用.....	80
11、Jsonp.....	80
jsonp 到底是什么？.....	80
12、Quartz.....	81
1. Quartz 是什么？.....	81
2. 用 Quartz 做定时任务调度.....	81
3. 如何监控 Quartz 的 job 执行状态：运行中，暂停中，等待中？.....	81
13、Mycat 中间件（新巴巴项目）	81
1. 为什么需要 MyCat？.....	81
2. MyCat 是什么？.....	82

第一部分：基础知识

1、常用的集合有哪些，它们之间的关系是怎样的？



区别		是否有序	是否允许元素重复
Collection		否	是
List		是	是
Set	AbstractSet	否	否
	HashSet		
	TreeSet	是(用二叉排序树)	
Map	AbstractMap	否	使用 key value 来映射储蓄数据,key 必须唯一,value 可以重复
	HashMap		
	TreeMap	是(用二叉排序树)	

2、java 中形参和实参是什么意思？

形参是方法小括号里面的参数，实参是调用此方法传入的数据。

3、list 集合底层是什么？长度是多少？内存扩展机制？

list 集合底层是数组它的默认长度是 10

4、"=="和 equals 方法究竟有什么区别？

==	比较两个变量的值是否相等	地址值比较
equals	两个独立对象的内容是否相同	字符串的比较

5、Java 中的方法覆盖（Overwrite）和方法重载（Overloading）是什么意思？

重载 重写 区别	重载（Overloading）	<ol style="list-style-type: none"> 1. 一般类当中 2. 方法名相同,参数列表不同(参数的个数,类型顺序)
	覆盖（Overwrite）	<ol style="list-style-type: none"> 1. 必须有继承 2. 方法名参数相同

6、sleep() 和 wait() 有什么区别？

wait：可以指定时间也可以不指定时间。

sleep：必须指定时间，时间到自动从冻结状态转成运行状态(临时阻塞状态)。

7、使用 final 关键字修饰一个变量时，是引用不能变，还是引用的对象不能变？

使用 final 关键字修饰一个变量时，是指引用变量不能变，引用变量所指向的对象中的内容还是可以改变的。

8、throws 与 throw 的区别？

Throw 是语句抛出一个异常,一般会程序员主动抛出某种特定类型异常

Throws 是方法可能抛出异常的声明。(用在声明方法时，表示该方法可能要抛出异常).交由上级调用它的方法程序进行处理。

9、Collection 和 Collections 的区别？

Collection	集合类的上级接口，继承与他的接口主要有 Set 和 List.
Collections	针对集合类的一个帮助类，他提供一系列静态方法实现对各种集合的搜索、排序、线程安全化等操作。

10、final, finally, finalize 的区别？

final	用于声明属性，方法和类，分别表示属性不可变，方法不可覆盖，类不可继承。
finally	异常处理语句结构的一部分，表示总是执行。
finalize	Object 类的一个方法，在垃圾收集器执行的时候会调用被回收对象的此方法，可以覆盖此方法提供垃圾收集时的其他资源回收，例如关闭文件等。

11、什么是哈希表？

哈希表又称散列表，是一种能将关键字映射成存储地址的记录存储技术。

12、数组有没有 length()方法？String 有没有 length()方法？

数组	没有 length()方法，只有 length 的属性
String	有 length()方法

13、List、Map、Set 三个接口，存取元素时，各有什么特点？

List	Map	Set
------	-----	-----

特定次序来持有元素，可有重复元素	保存 key-value 值， value 可多值	无法拥有重复元素， 内部排序
------------------	------------------------------	-------------------

14、遇到过的异常？

RuntimeException	使用 try{}catch()/用 throws 关键字声明抛出异常类
Error：错误类	程序无法处理的类
ClassCastException：数据类型转换异常	在前台控制程序中添加相应的转换方法
NumberFormatException：字符串转换为数字类型时抛出的异常	字符型转换为数字出现的问题，但是如果字符串中出现非数字的字符是，程序便会报错
java.lang.NullPointerException：空指针异常	调用了未初始化的对象或者不存在的对象
java.lang.ClassNotFoundException	如果程序中出现此类异常，可能是缺少某些第三方开源 jar 包
java.lang.ArithmeticException：数字运算的异常	如除数为零等

java.lang.ArithmeticException：数组下标越界	对数组进行操作时，如果显式的,用常数当做下标。
InterruptedException	线程调用 sleep()方法时会抛出此异常，

第二部分：多线程部分

1、什么是线程？说一下多线程的好处？

线在一个应用程序中,同时，有多个不同的执行路径，是进程中的实际运作单位。

好处是提供程序效率。

2、进程和线程的区别是什么？

进程是执行着的应用程序，而线程是进程内部的一个执行序列。一个进程可以有多个线程。线程又叫做轻量级进程。

3、如何在 Java 中实现线程？

在语言层面有两种方式。java.lang.Thread 类的实例就是一个线程但是它需要调用 java.lang.Runnable 接口来执行，由于线程类本身就是调用的 Runnable 接口所以你可以继承 java.lang.Thread 类或者直接调用 Runnable 接口来重写 run()方法实现线程。

4、那么你平时是用 Runnable 还是 Thread？

大家都知道我们可以通过继承 Thread 类或者调用 Runnable 接口来实现线程，问题是，哪个方法更好呢？什么情况下使用它？这个问题很容易回答，如果你知道 Java 不支持类的多重继承，但允许你调用多个接口。所以如果你要继承其他类，当然是调用 Runnable 接口好了。

5、创建线程有几种不同的方式？你喜欢哪一种？为什么？

有三种方式可以用来创建线程：

继承 Thread 类

实现 Runnable 接口

应用程序可以使用 Executor 框架来创建线程池

实现 Runnable 接口这种方式更受欢迎，因为这不需要继承 Thread 类。在应用设计中已经继承了别的对象的情况下，这需要多继承（而 Java 不支持多继承），只能实现接口。同时，线程池也是非常高效的，很容易实现和使用。

6、如何确保 N 个线程可以访问 N 个资源同时又不导致死锁？

使用多线程的时候，一种非常简单的避免死锁的方式就是：指定获取锁的顺序，并强制线程按照指定的顺序获取锁。因此，如果所有的线程都是以同样的顺序加锁和释放锁，就不会出现死锁了。

第三部分：集合相关面试题

1、你用过 HashMap 吗？” “什么是 HashMap？你为什么用到它？

几乎每个人都会回答“是的”，然后回答 HashMap 的一些特性，譬如 HashMap 可以接受 null 键值和值，而 Hashtable 则不能；HashMap 是非 synchronized；HashMap 很快；以及 HashMap 储存的是键值对等等。这显示出你已经用过 HashMap，而且对它相当的熟悉。但是面试官来个急转直下，从此刻开始问出一些刁钻的问题，关于 HashMap 的更多基础的细节。面试官可能会问出下面的问题：

2、那么你知道 HashMap 的工作原理吗？” “你知道 HashMap 的 get()方法的工作原理吗？

但一些面试者可能可以给出答案，“HashMap 是基于 hashing 的原理，我们使用 put(key, value) 存储对象到 HashMap 中，使用 get(key) 从 HashMap 中获取对象。当我们给 put() 方法传递键和值时，我们先对键调用 hashCode() 方法，返回的 hashCode 用于找到 bucket 位置来储存 Entry 对象。”

3、当两个对象的 hashCode 相同会发生什么？

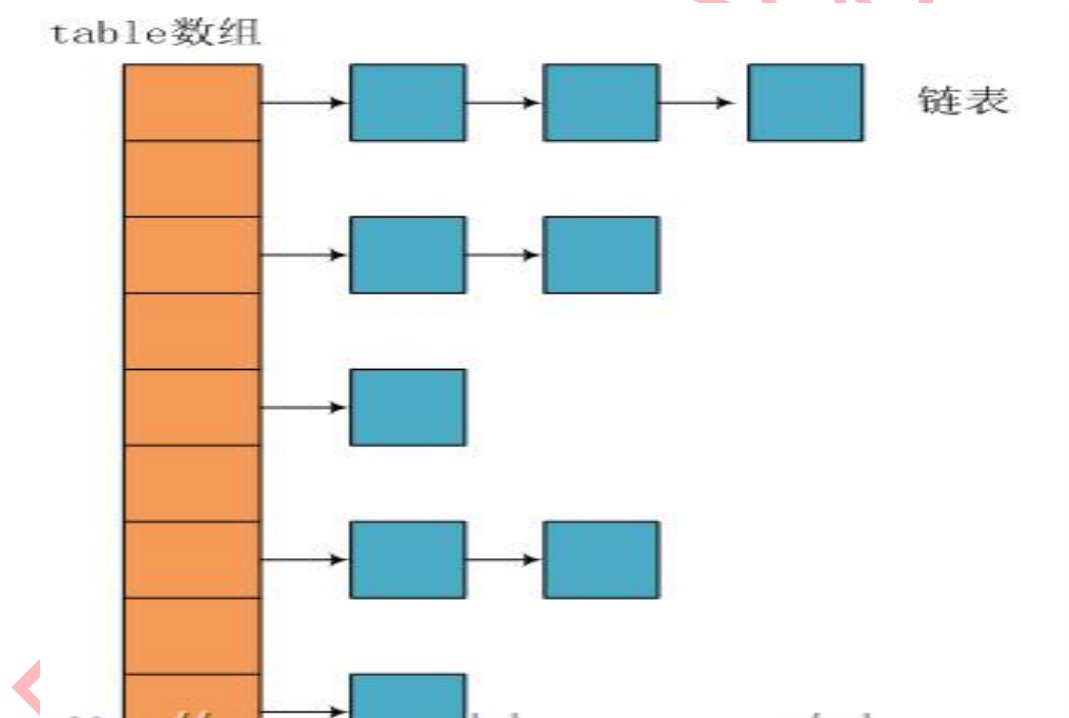
从这里开始，真正的困惑开始了，一些面试者会回答因为 hashCode 相同，所以两个对象是相等的，HashMap 将会抛出异常，或者不会存储它们。然后面试官可能会提醒他们有 equals() 和 hashCode() 两个方法，并告诉他们两个对象就算 hashCode 相同，但是它们可能并不相等。一些面试者可能就此放弃，而另外一些还能继续挺进，他们回答“因为 hashCode 相同，所以它们的 bucket 位置相同，‘碰撞’会发生。因为 HashMap 使用链表存储对象，这个 Entry(包含有键值对的 Map.Entry 对象)会存储在链表中。”这个答案非常的合理，虽然有很多种处理碰撞的方法，这种方法是最简单的，也正是 HashMap 的处理方法。但故事还没有完结，面试官会继续问：

4、如果两个键的 hashCode 相同，你如何获取值对象？

面试者会回答：当我们调用 get()方法，HashMap 会使用键对象的 hashCode 找到 bucket 位置，然后获取值对象。面试官提醒他如果有两个值对象储存在同一个 bucket，他给出答案：将会遍历链表直到找到值对象。面试官会问因为你并没有值对象去比较，你是如何确定确定找到值对象的？除非面试者直到 HashMap 在链表中存储的是键值对，否则他们不可能回答出这一题。

其中一些记得这个重要知识点的面试者会说，找到 bucket 位置之后，会调用 keys.equals()方法去找到链表中正确的节点，最终找到要找的值对象。完美的答案！

5、hashmap 的底层架构是什么？



6、hashmap 和 hashtable 的区别？

Hashtable	HashMap
线程安全	线程不安全
不允许 null 作为 key	可以使用 null 作为 key
HashMap 是对 Map 接口的实现	HashTable 实现了 Map 接口和 Dictionary 抽象类

7、HashSet 和 TreeSet 有什么区别？

HashSet 是由一个 hash 表来实现的，因此，它的元素是无序的。

TreeSet 是由一个树形的结构来实现的，它里面的元素是有序的。

8、如何遍历一个 List？

使用迭代器更加线程安全，因为它可以确保，在当前遍历的集合元素被更改的时候，它会抛出异常

第四部分：Http 部分

1、说一下什么是 Http 协议？它有什么组成？（大型互联网公司会问）

对客户端和 服务器端之间数据传输的格式规范，格式简称为“超文本传输协议”。

请求报文包含三部分：

- a、请求行：包含请求方法、URI、HTTP 版本信息
- b、请求首部字段
- c、请求内容实体

响应报文包含三部分：

- a、状态行：包含 HTTP 版本、状态码、状态码的原因短语
- b、响应首部字段
- c、响应内容实体

2、Http 协议中有常用的请求方式你知道有哪些么？

此时面试官不是想让你回答 get post 这两种大家都知道，其实他是想让你回答 get 和 post 有什么区别。

最主要有两种：

GET：用于请求访问已经被 URI（统一资源标识符）识别的资源，可以通过 URL 传参给服务器

POST：用于传输信息给服务器，主要功能与 GET 方法类似，但一般推荐使用 POST 方式。

3、对网络通信协议熟悉吗？对 socket 和 http 了解吗？能解释下吗？

面试官是想让你说一下 socket、http 的区别。大概介绍一下分别是什么，他们的区别在哪里就好了

socket 是嵌套字，负责网络通讯，http 是建立在通讯已经建立的情况下（socket 已经连通）之后的一种网络访问协议
平时上网就是用的 http 协议

tcp/ip 是网络通信协议 socket 是网络编程接口

打个比喻：将 socket 比喻为高速公路，HTTP 就是公路上跑的车。路上除了 HTTP 外，还有很多种其他类型的车，Ftp, Telnet, SMTP……

4、说一下 Http 协议中 302 状态(阿里经常问)

http 协议中，返回状态码 302 表示重定向。

这种情况下，服务器返回的头部信息中会包含一个 Location 字段，内容是重定向到的 url

5、Http 协议有那些特征？

1、支持客户/服务器模式；2、简单快速；3、灵活；4、无连接；5、无状态

第五部分：IO 部分

1、Java 中有几种类型的流？

分类	字节输入流	字节输出流	字符输入流	字符输出流
抽象基类	InputStream	OutputStream	Reader	Writer
访问文件	FileInputStream	FileOutputStream	FileReader	FileWriter
访问数组	ByteArrayInputStream InputStream	ByteArrayOutputStream OutputStream	CharArrayReader Reader	CharArrayWriter Writer
访问字符串			StringReader	StringWriter
缓冲流	BufferedReader	BufferedWriter	BufferedReader	BufferedWriter

提高效率	InputStream	OutputStream		Writer
------	-------------	--------------	--	--------

2、io 流怎样读取文件的？说下流程。

使用 File 对象获取文件路径，通过字符流 Reader 加入文件，使用字符缓存流 BufferedReader 处理 Reader，再定义一个字符串，循环遍历出文件。

3、String,StringBuffer 和 StringBuilder 之间的区别？

	类型	可变	线程安全/使用	操作
String	字符串常量	不可变	安全/少量数据	产生一个新对象
StringBuffer	字符串变量	可变	安全/多线程	内容发生改变
Stringbulider	字符串变量	可变	不安全/单线程	内容发生改变

1. 三者在执行速度: StringBuilder>StringBuffer>String

4、解释一下 java.io.Serializable 接口？

类通过实现，java.io.Serializable 接口以启用其序列化功能。未实现此接口的类将无法使其任何状态序列化或反序列化。

第六部分：JavaWeb

1、JSP 中动态 include 与静态 include 的区别？

静态 include	<%@include%>,直接将内容先包含后处理
动态 include	<jsp:include>,如果包含的是动态页,则先编译之后再进行处理

2、两种跳转方式是什么?有什么区别?

服务器端跳转	<jsp:forward>,跳转之后地址栏不改变,可以传递 request
--------	---------------------------------------

	属性,实际是对 <code>RequestDispatcher</code> 接口的封装.
客户端跳转	<code>response.sendRedirect()</code> ,跳转之后地址栏改变.

4、jsp 和 servlet 的区别、共同点、各自应用的范围??

	区别	各自应用范围
Jsp	JSP 是 Servlet 技术的扩展, 本质上就是 Servlet 的简易方式。JSP 编译后是“类 servlet” JSP 侧重于视图	是 Java 和 HTML 可以组合成一个扩展名为 .jsp 的文件
Servlet	JSP 是 Servlet 技术的扩展,Servlet 主要用于控制逻辑;在 struts 框架中,JSP 位于 MVC 设计模式的视图层,而 Servlet 位于控制层.	应用逻辑是在 Java 文件中, 并且完全从表示层中的 HTML 里分离开来

5、cookie 和 session 的区别?

cookie	主要用在保存客户端, 其值在客户端与服务端之间传送, 不安全, 存储的数据量有限
session	保存在服务端, 每一个 session 在服务端有一个 sessionId 作一个标识。存储的数据量大, 安全性高。占用服务端的内存资源

6、Tomcat 的优化经验?

去掉对 web.xml 的监视, 把 JSP 提前编辑成 Servlet ;有富余物理内存的情况下, 加大 Tomcat 使用的 JVM 内存。

7、request.getParameter()和 request.getAttribute()的区别?

<code>request.getParameter()</code>	获取的类型是 String	获取的是 POST/GET 传递的参数值和 URL 中的参数
<code>request.getAttribute()</code>	获取的类型是 Object	获取的是对象容器中的数据值/对象

8、如果 JSP 表单元素的值为空，如何避免 null 出现在页面上？

可以写一个简单的函数对空值进行处理，判断值是否为空，如果为空就返回空字符串。

9、你在项目中用到了 XML 技术的哪些方面？如何实现的？

在企业中一般用到了数据存储、信息配置两方面。

在做数据交换平台时，将不能数据源的数据组装成 XML 文件，然后将 XML 文件压缩打包加密后通过网络传送给接受者，接收解密与解压缩后再同 XML 文件中还原相关信息进行处理。

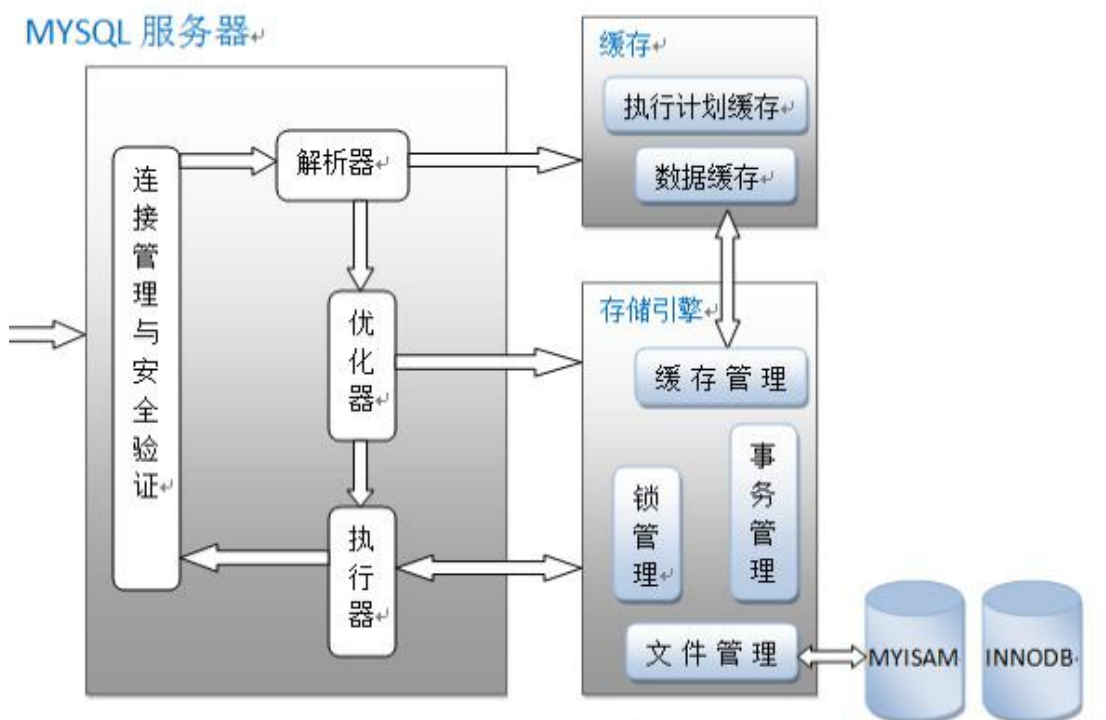
在做软件配置时，利用 XML 可以和方便的进行，软件的各种配置参数都存储在 XML 文件中。

第七部份：数据库

一：关系型数据库

1、Mysql 部分

1.1 架构图介绍



1.1.1 连接管理与安全验证是什么？

每个客户端都会建立一个与服务器连接的线程，服务器会有一个线程池来管理这些连接；如果客户端需要连接到 MySQL 数据库还需要进行验证，包括用户名、密码、主机信息等。

1.1.2 解析器是什么？

解析器的作用主要是分析查询语句，最终生成解析树；首先解析器会对查询语句的语法进行分析，分析语法是否有问题。还有解析器会查询缓存，如果在缓存中有对应的语句，就返回查询结果不进行接下来的优化执行操作。前提是缓存中的数据没有被修改，当然如果被修改了也会被清出缓存。

1.1.3 优化器怎么用？

优化器的作用主要是对查询语句进行优化操作，包括选择合适的索引，数据的读取方式，包括获取查询的开销信息，统计信息等，这也是为什么图中会有优化器指向存储引擎的箭头。之前在别的文章没有看到优化器跟存储引擎之间的关系，在这里我个人的理解是因为优化器需要通过存储引擎获取查询的大致数据和统计信息。

1.1.4 执行器是什么？

执行器包括执行查询语句，返回查询结果，生成执行计划包括与存储引擎的一些处理操作。

1.2、存储引擎都有哪些？

1) InnoDB 存储引擎

InnoDB 是事务型数据库的首选引擎，支持事务安全表 (ACID)，支持行锁定和外键，InnoDB 是默认的 MySQL 引擎。

2) MyISAM 存储引擎

MyISAM 基于 ISAM 存储引擎，并对其进行扩展。它是在 Web、数据仓储和其他应用环境下最常使用的存储引擎之一。MyISAM 拥有较高的插入、查询速度，但不支持事物。

3) MEMORY 存储引擎

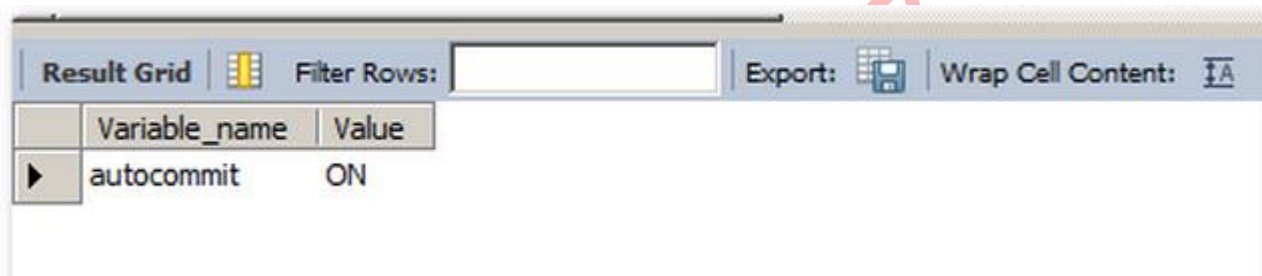
MEMORY 存储引擎将表中的数据存储在内存中，未查询和引用其他表数据提供快速访问。

1.3、事务介绍

mysql 和其它的数据库产品有一个很大的不同就是事务由存储引擎所决定，例如 MYISAM, MEMORY, ARCHIVE 都不支持事务，事务就是为了解决一组查询要么全部执行成功，要么全部执行失败。

mysql 事务默认是采取自动提交的模式，除非显示开始一个事务

SHOW VARIABLES LIKE 'AUTOCOMMIT';



Variable_name	Value
autocommit	ON

修改自动提交模式，0=OFF,1=ON

注意：修改自动提交对非事务类型的表是无效的，因为它们本身就没有提交和回滚的概念，还有一些命令是会强制自动提交的，比如 DDL 命令、lock tables 等。

SET AUTOCOMMIT=OFF 或 SET AUTOCOMMIT=0

事务的 ACID 特性是什么？

答：数据库事务 transaction 正确执行的四个基本要素。**ACID**、原子性(Atomicity)、一致性(Correspondence)、隔离性(Isolation)、持久性(Durability)。

- 1) **原子性**:整个事务中的所有操作，要么全部完成，要么全部不成功，不可能停滞在中间某个环节。事务在执行过程中发生错误，会被回滚 (Rollback) 到事务开始前的状态，就像这个事务从来没有执行过一样。
- 2) **一致性**:在事务开始之前和事务结束以后，数据库的完整性约束没有被破坏。
- 3) **隔离性**:隔离状态执行事务，使它们好像是系统在给定时间内执行的唯一操作。如果有两个事务，运行在相同的时间内，执行 相同的功能，事务的隔离性将确保每一事务在系统中认为只有该事务在使用系统。这种属性有时称为串行化，为了防止事务操作间的混淆，必须串行化或序列化请求，使得在同一时间仅有一个请求用于同一数据。
- 4) **持久性**:在事务完成以后，该事务所对数据库所作的更改便持久的保存在数据库之中，并不会被回滚。

mysql 有四种隔离级别分别是什么：

未提交读 (READ UNCOMMITTED)：未提交读隔离级别也叫读脏，就是事务可以读取其它事务未提交的数据。

提交读 (READ COMMITTED) :在其它数据库系统比如 SQL Server 默认的隔离级别就是提交读,已提交读隔离级别就是在事务未提交之前所做的修改其它事务是不可见的。

可重复读 (REPEATABLE READ) :保证同一个事务中的多次相同的查询的结果是一致的,比如一个事务一开始查询了一条记录然后过了几秒钟又执行了相同的查询,保证两次查询的结果是相同的,可重复读也是 mysql 的默认隔离级别。

可串行化 (SERIALIZABLE) :可串行化就是保证读取的范围内没有新的数据插入,比如事务第一次查询得到某个范围的数据,第二次查询也同样得到了相同范围的数据,中间没有新的数据插入到该范围中。

1.4、创建存储过程怎么写？

“pr_add”是个简单的 MySQL 存储过程,这个 MySQL 存储过程有两个 int 类型的输入参数 “a”、“b”,返回这两个参数的和。

1) drop procedure if exists pr_add;

2) 计算两个数之和

```
create procedure pr_add ( a int, b int ) begin declare c int;

if a is null then set a = 0;

end if;

if b is null then set b = 0;

end if;

set c = a + b;

select c as sum;
```

1.5、触发器怎么写？

触发器语句：

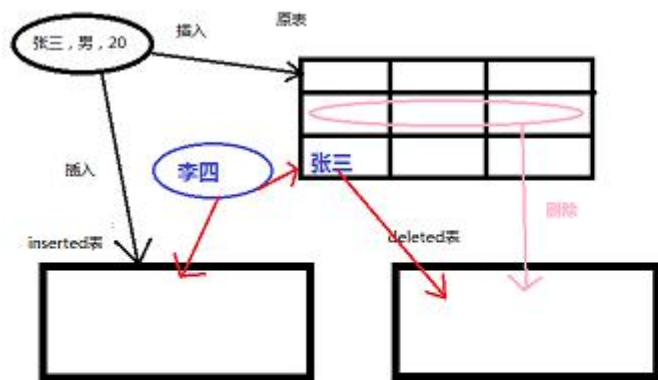
create trigger 触发器的名字 on 操作表

for|after instead of

update|insert|delete

as

SQL 语句



注：如果一次性的向表中插入或者删除多条数据，则触发器也进行相应次数的触发，但(在inserted或者deleted表中)只会保留最后一次结果！

Alpha触发器原理示意图

Situation One 插入操作：

在插入原表的同时，也将数据插入到了inserted表中

Situation Two 删除操作：

将原表中要删除的数据移动到deleted表中

Situation Three 更新操作：

更新操作如红线所示，先将原表中要更新的数据移动到deleted表中，再将更新后的内容，分别插入到原表与inserted表中

1.6、内连接,左外连接,右外连接,全外连接,交叉连接?

两张简单的信息表来说明问题

test1：

	name	sex
▶	张三	男
	王晓	女
*	NULL	NULL

test2：

	name2	age
▶	王晓	25
	王三	30
*	NULL	NULL

答：内连接：只连接匹配的行；

```
SELECT    dbo.test1.name, dbo.test1.sex, dbo.test2.name2, dbo.test2.age
1. FROM    dbo.test1 inner JOIN dbo.test2 on test1.name =test2.name2
```

	name	sex	name2	age
1	王晓	女	王晓	25

左外连接：包含左边表的全部行（不管右边的表中是否存在与它们匹配的行）以及右边表中全部匹配的行；

```
1. SELECT      dbo.test1.name, dbo.test1.sex, dbo.test2.name2, dbo.test2.age
2. FROM        dbo.test1 left JOIN dbo.test2 on test1.name =test2.name2
```

	name	sex	name2	age
1	张三	男	NULL	NULL
2	王晓	女	王晓	25

右外连接：包含右边表的全部行（不管左边的表中是否存在与它们匹配的行）以及左边表中全部匹配的行；

```
1. SELECT      dbo.test1.name, dbo.test1.sex, dbo.test2.name2, dbo.test2.age
2. FROM        dbo.test1 right JOIN dbo.test2 on test1.name =test2.name2
```

	name	sex	name2	age
1	王晓	女	王晓	25
2	NULL	NULL	王三	30

全外连接：包含左、右两个表的全部行，不管在另一边的表中是否存在与它们匹配行；

```
1. SELECT      dbo.test1.name, dbo.test1.sex, dbo.test2.name2, dbo.test2.age
2. FROM        dbo.test1 full outer JOIN dbo.test2 on test1.name =test2.name2
```

	name	sex	name2	age
1	张三	男	NULL	NULL
2	王晓	女	王晓	25
3	NULL	NULL	王三	30

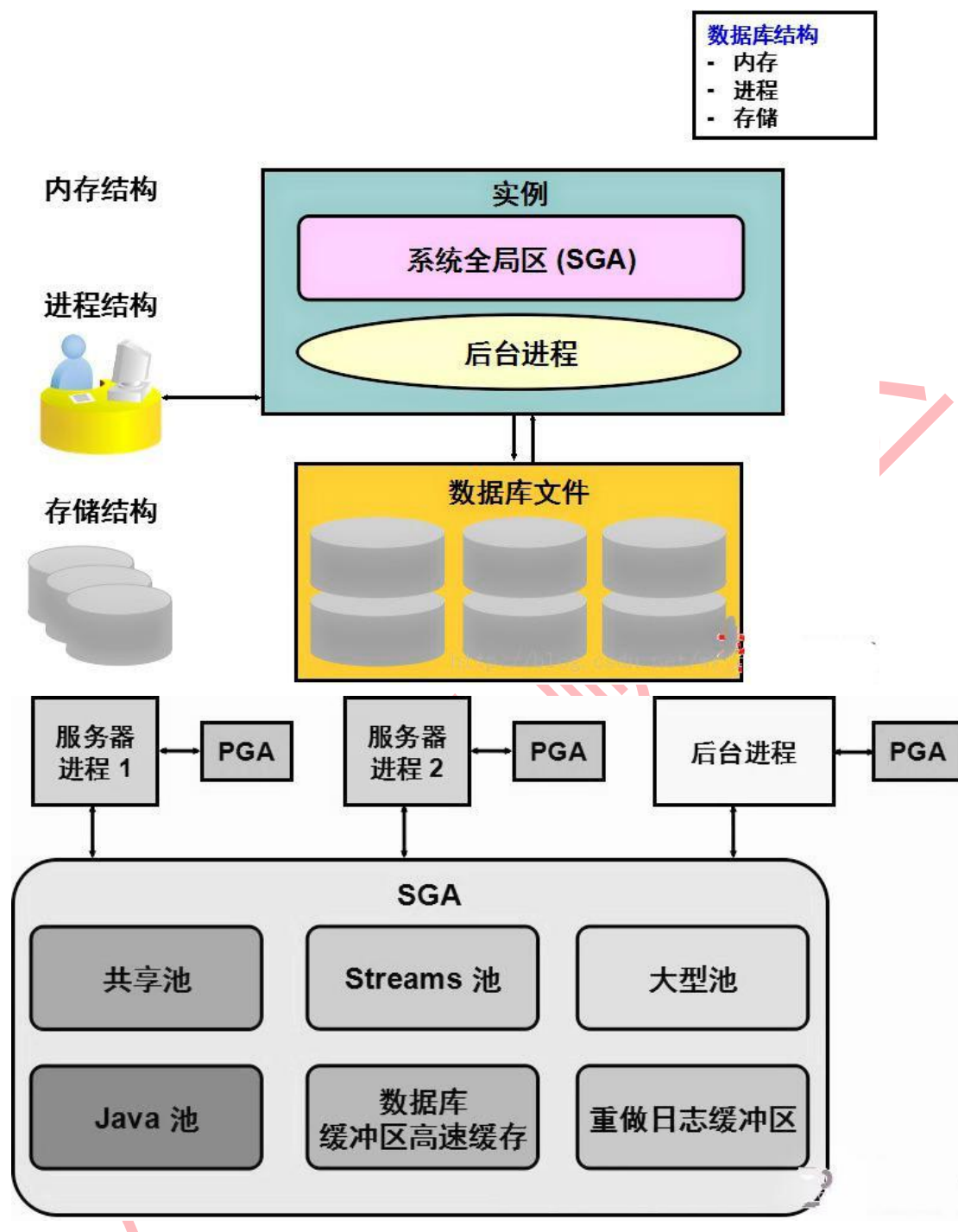
交叉连接：生成笛卡尔积——它不使用任何匹配或者选取条件，而是直接将一个数据源中的每个行与另一个数据源的每个行一一匹配；

```
1. SELECT      dbo.test1.name, dbo.test1.sex, dbo.test2.name2, dbo.test2.age
2. FROM        dbo.test1 CROSS JOIN
3.            dbo.test2
```

	name	sex	name2	age
▶	张三	男	王晓	25
	王晓	女	王晓	25
	张三	男	王三	30
	王晓	女	王三	30

2、Oracle

2.1、Oracle 内存结构介绍



2.2、数据结构是什么样子的？

<http://www.cnblogs.com/chengxiao/p/5904783.html>

2.3、oracle 数据库之事务有哪些？

1.什么是事务

在数据库中事务是工作的逻辑单元，一个事务是由一个或多个完成一组的相关行为的 SQL 语句组成，通过事务机制确保这一组 SQL 语句所作的操作要么都成功执行，完成整个工作单元操作，要么一个也不执行。

如：网上转帐就是典型的要用事务来处理，用以保证数据的一致性。

2.事务特性

原子性(Atomicity)：一个事务里面所有包含的 SQL 语句是一个执行整体，不可分割，要么都做，要么都不做。

一致性(Consistency)：事务开始时，数据库中的数据是一致的，事务结束时，数据库的数据也应该是一致的。

隔离性(Isolation)：是指数据库允许多个并发事务同时对其中的数据读写和修改的能力，隔离性可以防止事务的并发执行时，由于他们的操作命令交叉执行而导致的数据不一致状态。

持久性 (Durability)：是指当事务结束后，它对数据库中的影响是永久的，即便系统遇到故障的情况下，数据也不会丢失。

3.事务隔离级别

隔离级别	脏读	不可重复读	幻读
Read uncommitted(读未提交)	是	是	是
Read committed (读已提交)	否	是	是
Repeatable read (可重复读)	否	否	是
Serializable (串行读)	否	否	否

Oracle 默认的隔离级别是 read committed。

Oracle 支持上述四种隔离级别中的两种:read committed 和 serializable。除此之外，Oracle 中还定义 Read only 和 Read write 隔离级别。

Read only：事务中不能有任何修改数据库中数据的操作语句，是 Serializable 的一个子集。

Read write：它是默认设置，该选项表示在事务中可以有访问语句、修改语句，但不经常使用。

2.4、Oracle 创建存储过程怎么写？

注意：oracle 的存储过程是有一定难度的，看不懂的就说自己没用过。

```
create or replace procedure test_count
as
v_total number(1);
begin

select count(*) into v_total from TESTTABLE;
DBMS_OUTPUT.put_line('总人数：'||v_total);
end;
--准备
--线对 scott 解锁：alter user scott account unlock;
--应为存储过程是在 scott 用户下。还要给 scott 赋予密码
---alter user scott identified by tiger;
---去命令下执行
EXECUTE test_count;
----在 ql/spl 中的 sql 中执行
begin
-- Call the procedure
test_count;
end;
```

参考资料：<http://blog.csdn.net/o9109003234/article/details/24910039>

2.5、如何使用 Oracle 的游标？

1. oracle 中的游标分为显示游标和隐式游标
2. 显示游标是用 cursor... is 命令定义的游标，它可以对查询语句(select)返回的多条记录进行处理；隐式游标是在执行插入(insert)、删除(delete)、修改(update)和返回单条记录的查询(select)语句时由 PL/SQL 自动定义的。
3. 显式游标的操作：打开游标、操作游标、关闭游标；PL/SQL 隐式地打开 SQL 游标，并在它内部处理 SQL 语句，然后关闭它。

2.6、Oracle 中字符串用什么符号链接？

答：Oracle 中使用 || 这个符号连接字符串 如 'abc' || 'd'

2.7、Oracle 是怎样分页的？

Oracle 中使用 rownum 来进行分页，这个是效率最好的分页方法，hibernate 也是使用 rownum 来进行 oracle 分页的

```
select * from
```

```
( select rownum r,a from tableName where rownum <= 20 )
```

```
where r > 10
```

2.8、什么是存储过程？它有什么优点？

答：存储过程是一组预编译的 SQL 语句，

它的优点有：

允许模块化程序设计，就是说只需要创建一次过程，以后在程序中就可以调用该过程任意次。

允许更快执行，如果某操作需要执行大量 SQL 语句或重复执行，存储过程比 SQL 语句执行的要快。

减少网络流量，例如一个需要数百行的 SQL 代码的操作有一条执行语句完成，不需要在网络中发送数百行代码。

更好的安全机制，对于没有权限执行存储过程的用户，也可授权他们执行存储过程。

2.9、存储过程和函数有什么区别？

Oracle 中的函数与存储过程的区别：

1. 函数必须有返回值,而过程没有.
2. 函数可以单独执行.而过程必须通过 execute 执行.
3. 函数可以嵌入到 SQL 语句中执行.而过程不行.

其实我们可以将比较复杂的查询写成函数.然后到存储过程中去调用这些函数.

2.10 、Oracle 中的函数与存储过程的特点:

1. 一般来说，存储过程实现的功能要复杂一点，而函数的实现的功能针对性比较强。
2. 对于存储过程来说可以返回参数，而函数只能返回值或者表对象。
3. 存储过程一般是作为一个独立的部分来执行，而函数可以作为查询语句的一个部分来调用，由于函数可以返回一个表对象，因此它可以在查询语句中位于 FROM 关键字的后面。

2.11、 mysql , oracle , sql server 三者的区别？

1.mysql

优点：

体积小、速度快、总体拥有成本低，开源；支持多种操作系统；是开源数据库，提供的接口支持多种语言连接操作。

缺点：

不支持热备份；

MySQL 最大的缺点是其安全系统，主要是复杂而非标准，另外只有到调用 mysqladmin 来重读用户权限时才发生改变；

没有一种存储过程(Stored Procedure)语言，这是对习惯于企业级数据库的程序员的最大限制；

MySQL 的价格随平台和安装方式变化。Linux 的 MySQL 如果由用户自己或系统管理员而不是第三方安装则是免费的，第三方案则必须付许可费。Unix 或 Linux 自行安装 免费、Unix 或 Linux 第三方安装 收费；

2.oracle

优点：

开放性：Oracle 能在所有主流平台上运行（包括 windows）完全支持所有工业标准，采用完全开放策略，使客户选择适合解决方案；

可伸缩性,并行性：Oracle 并行服务器通过使组结点共享同簇工作来扩展 windownt 能力，提供高用性和高伸缩性簇解决方案。

安全性：获得最高认证级别的 ISO 标准认证。

性能：Oracle 性能高 保持开放平台下 TPC-D 和 TPC-C 世界记录；

客户端支持及应用模式：Oracle 多层次网络计算支持多种工业标准用 ODBC、JDBC、OCI 等网络客户连接

使用风险：Oracle 长时间开发经验完全向下兼容得广泛应用地风险低。

缺点：

对硬件的要求很高；

价格比较昂贵；

管理维护麻烦一些；
操作比较复杂，需要技术含量较高；

3.sqlserver：

优点：

易用性、适合分布式组织的可伸缩性、用于决策支持的数据仓库功能、与许多其他服务器软件紧密关联的集成性、良好的性价比等；

SQLServer 是一个具备完全 Web 支持的数据库产品，提供了对可扩展标记语言 (XML) 的核心支持以及在 Internet 上和防火墙外进行查询的能力；

缺点：SQL Server 只能 windows 上运行，没有丝毫开放性操作系统。

伸缩性并行性：数据卷伸缩性有限；

安全性：没有获得任何安全证书。

性能：SQL Server 多用户时性能佳；

客户端支持及应用模式：客户端支持及应用模式。只支持 C/S 模式，SQL Server C/S 结构只支持 windows 客户用 ADO、DAO、OLEDB、ODBC 连接；

使用风险：SQL server 完全重写代码经历了长期测试，需要时间来证明并十分兼容；

二：非关系型数据库

1、Redis

1.1、什么是 redis?

Redis 是一个基于内存的高性能 key-value 数据库。

1.2、Redis 的特点？

Redis 本质上是一个 **Key-Value** 类型的内存数据库，很像 **memcached**，整个数据库统统加载在内存当中进行操作，定期通过异步操作把数据库数据 **flush** 到硬盘上进行保存。因为是纯内存操作，Redis 的性能非常出色，每秒可以处理超过 10 万次读写操作，是已知性能最快的 Key-Value DB。

Redis 的出色之处不仅仅是性能，Redis 最大的魅力是支持保存多种数据结构，此外单

个 value 的最大限制是 1GB，不像 memcached 只能保存 1MB 的数据，另外 Redis 也可以对存入的 Key-Value 设置 expire 时间。

Redis 的主要缺点是数据库容量受到物理内存的限制，不能用作海量数据的高性能读写，因此 Redis 适合的场景主要局限在较小数据量的高性能操作和运算上。

1.3、为什么 redis 需要把所有数据放到内存中？

Redis 为了达到最快的读写速度将数据都读到内存中，并通过异步的方式将数据写入磁盘。所以 redis 具有快速和数据持久化的特征。如果不将数据放在内存中，磁盘 I/O 速度为严重影响 redis 的性能。在内存越来越便宜的今天，redis 将会越来越受欢迎。如果设置了最大使用的内存，则数据已有记录数达到内存限值后不能继续插入新值。

1.4、Redis 常见的性能问题都有哪些？如何解决？

1. Master 写内存快照，save 命令调度 rdbSave 函数，会阻塞主线程的工作，当快照比较大时对性能影响是非常大的，会间断性暂停服务，所以 Master 最好不要写内存快照。

2. Master AOF 持久化，如果不重写 AOF 文件，这个持久化方式对性能的影响是最小的，但是 AOF 文件会不断增大，AOF 文件过大会影响 Master 重启的恢复速度。Master 最好不要做任何持久化工作，包括内存快照和 AOF 日志文件，特别是不要启用内存快照做持久化，如果数据比较关键，某个 Slave 开启 AOF 备份数据，策略为每秒同步一次。

3. Master 调用 BGREWRITEAOF 重写 AOF 文件，AOF 在重写的时候会占大量的 CPU 和内存资源，导致服务 load 过高，出现短暂服务暂停现象。

4. Redis 主从复制的性能问题，为了主从复制的速度和连接的稳定性，Slave 和 Master 最好在同一个局域网内

1.5、redis 最适合的场景有哪些？

- (1)、会话缓存 (Session Cache)
- (2)、全页缓存 (FPC)
- (3)、队列
- (4)、排行榜/计数器
- (5)、发布/订阅

2、Memcache 与 Redis 的区别都有哪些？

- 1 存储方式不同，Memcache 是把数据全部存在内存中，数据不能超过内存的大小，断电后数据库会挂掉。Redis 有部分存在硬盘上，这样能保证数据的持久性。
- 2 数据支持的类型不同 memcache 对数据类型支持相对简单，redis 有复杂的数据类型。
- 3 使用底层模型不同 它们之间底层实现方式 以及与客户端之间通信的应用协议不一样。Redis 直接自己构建了 VM 机制，因为一般的系统调用系统函数的话，会浪费一定的时间去移动和请求。
- 3 支持的 value 大小不一样 redis 最大可以达到 1GB，而 memcache 只有 1MB。

三：数据库语句优化

1、Mysql sql 语句优化：

Mysql 语句优化：

1、问：where 子句中可以对字段进行 null 值判断吗？

```
select id from t where num is null
```

答：最好不要给数据库留 NULL，尽可能的使用 NOT NULL 填充数据库。

不要以为 NULL 不需要空间，比如：char(100) 型，在字段建立时，空间就固定了，不管是否插入值（NULL 也包含在内），都是占用 100 个字符的空间的，如果是 varchar 这样的变长字段，null 不占用空间。

可以在 num 上设置默认值 0，确保表中 num 列没有 null 值，然后这样查询：

```
select id from t where num = 0
```

2、问：select * from admin left join log on admin.admin_id = log.admin_id where log.admin_id>10 优化

优化为：select * from (select * from admin where admin_id>10) T1 left join log on T1.admin_id = log.admin_id

答：使用 JOIN 时候，应该用小的结果驱动大的结果（left join 左边表结果尽量小如果有条件应该放到左边先处理，right join 同理反向），同事尽量把牵涉到多表联合的查询拆分多个 query（多个连表查询效率低，容易到之后锁表和阻塞）。

3、注意 LIKE 模糊查询的使用，避免使用 %%,可以使用 后面带% ，双%是不走索引的。

例：select * from admin where admin name like '%de' 优化

优化为 select * from admin where admin_name >='de' and admin_name <'df'

4、limit 的基数比较大时使用 between。

例：select * from admin order by admin_id limit 100000,10 优化

优化为：select * from admin where admin_id between 100000 admin_id 100010 order by admin_id

5、 尽量避免在列上做运算，这样导致索引失效。

例：select * from admin where year(admin_time)>2014 优化

优化为： select * from admin where admin_time> '2014-01-01'

2、oracle sql 语句优化有哪些？

(1) 选择最有效率的表名顺序

必

ORACLE 采用自下而上的顺序解析 WHERE 子句,根据这个原理,表之间的连接

须写在其他 WHERE 条件之前, 那些可以过滤掉最大数量记录的条件必须写在 WHERE 子句的末尾.

(2) WHERE 子句中的连接顺序 .

必

ORACLE 采用自下而上的顺序解析 WHERE 子句,根据这个原理,表之间的连接

须写在其他 WHERE 条件之前, 那些可以过滤掉最大数量记录的条件必须写在 WHERE 子句的末尾.

(3) SELECT 子句中避免使用 ' * '

(4) 在 SQL*Plus , SQL*Forms 和 Pro*C 中重新设置 ARRAYSIZE 参数, 可以增加每次数据库访问的检索数据量 ,建议值为 200

(5) 删除重复记录：

最高效的删除重复记录方法（因为使用了 ROWID）例子：

```
1 DELETE FROM EMP E WHERE E.ROWID > (SELECT MIN(X.ROWID)
2 FROM EMP X WHERE X.EMP_NO = E.EMP_NO);
```

四：数据库优化：

1. Mysql 数据库优化有哪些？

数据库的设计

尽量把数据库设计的更小的占磁盘空间.

- 1).尽可能使用更小的整数类型.(mediumint 就比 int 更合适).
- 2).尽可能的定义字段为 not null,除非这个字段需要 null.
- 3).如果没有用到变长字段的话比如 varchar,那就采用固定大小的纪录格式比如 char.
- 4).表的主索引应该尽可能的短.这样的话每条纪录都有名字标志且更高效.
- 5).只创建确实需要的索引.索引有利于检索记录,但是不利于快速保存记录.如果总是要在表的组合字段上做搜索,那么就在这些字段上创建索引.索引的第一部分必须是最常使用的字段.如果总是需要用到很多字段,首先就应该多复制这些字段,使索引更好的压缩.
- 6).所有数据都得在保存到数据库前进行处理。
- 7).所有字段都得有默认值。
- 8).在某些情况下,把一个频繁扫描的表分成两个速度会快好多。在对动态格式表扫描以取得相关记录时,它可能使用更小的静态格式表的情况下更是如此。

系统的瓶颈

1).磁盘搜索.

并行搜索,把数据分开存放到多个磁盘中,这样能加快搜索时间.

2).磁盘读写(IO)

可以从多个媒介中并行的读取数据。

3).CPU 周期

数据存放在主内存中.这样就得增加 CPU 的个数来处理这些数据。

4).内存带宽

当 CPU 要将更多的数据存放到 CPU 的缓存中来的话,内存的带宽就成了瓶颈。

2、Oracle 数据库优化有哪些？

Oracle 分区是怎样优化数据库的？

Oracle 的分区可以分为：列表分区、范围分区、散列分区、复合分区。

- 1). 增强可用性：如果表的一个分区由于系统故障而不能使用，表的其余好的分区仍可以使用；
- 2). 减少关闭时间：如果系统故障只影响表的一部份分区，那么只有这部份分区需要修复，可能比整个大表修复花的时间更少；
- 3). 维护轻松：如果需要得建表，独产管理每个公区比管理单个大表要轻松得多；
- 4). 均衡 I/O：可以把表的不同分区分配到不同的磁盘来平衡 I/O 改善性能；
- 5). 改善性能：对大表的查询、增加、修改等操作可以分解到表的不同分区来并行执行，可使运行速度更快
- 6). 分区对用户透明，最终用户感觉不到分区的存在。

第八部份：框架

一：Hibernate

1、Hibernate 和 JDBC 优缺点对比

相同点

- 两者都是 java 数据库操作的中间件
- 两者对数据库进行直接操作的对象都是线程不安全的，都需要及时关闭
- 两者都可对数据库的更新操作进行显式的事务处理

不同点

- JDBC 是 SUN 公司提供一套操作数据库的规范,使用 java 代码操作数据库。Hibernate 是一个基于 jdbc 的主流持久化框架，对 JDBC 访问数据库的代码做了封装。
- 使用的 SQL 语言不同 JDBC 使用的是基于关系型数据库的标准 SQL 语言, Hibernate 使用的是 HQL(Hibernate query language)语言
- 操作的对象不同：JDBC 操作的是数据，将数据通过 SQL 语句直接传送到数据库中执行，Hibernate 操作的是持久化对象，由底层持久化对象的数据更新到数据库中。
- 数据状态不同：JDBC 操作的数据是“瞬时”的，变量的值无法与数据库中的值保持一致，而 Hibernate 操作的数据是可持久的，即持久化对象的数据属性的值是可以跟数据库中的值保持一致的。

2、关于 Hibernate 的 orm 思想你了解多少？

- ORM 指的是对象关系型映射(Object RelationShip Mapping)，指的就是我们通过创建实体类对象和数据库中的表关系进行一一对应，来实现通过操作实体类对象来更改数据库里边的数据信息。这里边起到关键作用的是通过 Hibernate 的映射文件+Hibernate 的核心配置文件

3、Hibernate 的开发流程是怎么样的？

- 第一步：加载 hibernate 的配置文件，读取配置文件的参数（jdbc 连接参数，数据库方言，hbm 表与对象关系映射文件）
- 第二步：创建 SessionFactory 会话工厂（内部有连接池）
- 第三步：打开 session 获取连接，构造 session 对象（一次会话维持一个数据连接，也是一级缓存）
- 第四步：开启事务
- 第五步：进行操作
- 第六步：提交事务
- 第七步：关闭 session(会话)将连接释放
- 第八步：关闭连接池

4、谈谈 Hibernate 里边持久态对象的三种状态。

持久化对象的三种状态

- ①瞬时态(临时态、自由态)：不存在持久化标识 OID，尚未与 Hibernate Session 关联对象，被认为处于瞬时态，失去引用将被 JVM 回收
 - ②持久态：存在持久化标识 OID，与当前 session 有关联，并且相关联的 session 没有关闭，并且事务未提交
 - ③脱管态(离线态、游离态)：存在持久化标识 OID，但没有与当前 session 关联，脱管状态改变 hibernate 不能检测到
- 区分三种状态：判断对象是否有 OID，判断对象是否与 session 关联(被一级缓存引用)

5、Hibernate 缓存机制是怎样的？谈谈 Hibernate 的一级缓存和二级缓存

- hibernate 向一级缓存放入数据时，同时保存快照数据(数据库备份)，当修改一级缓存数据，在 flush 操作时，对比缓存和快照，如果不一致，自动更新(将缓存的内容同步到数据库，更新快照区)。
- 当持久化一个对象时，该对象被载入缓存，以后即使程序中不再引用该对象，只要缓存不清空，该对象仍然处于生命周期中。
- Hibernate 中持久态对象具有自动更新数据库能力(持久态对象才保存在 Session 中，才有快照)

hibernate 缓存特点：

第一类 hibernate 的一级缓存

- (1) hibernate 的一级缓存默认打开的
- (2) hibernate 的一级缓存使用范围，是 session 范围，从 session 创建到 session 关闭范围
- (3) hibernate 的一级缓存中，存储数据必须是持久态数据

第二类 hibernate 的二级缓存

- (1) 目前已经不使用了，替代技术 redis
- (2) 二级缓存默认不是打开的，需要配置
- (3) 使用二级缓存引入其他的组件
- (4) 二级缓存使用范围，是 sessionFactory 范围

如果理解好的同学可以根据上课老师讲的知识自己扩充。

6、Hibernate 有哪几种查询数据的方式

这个题的回答重点是要能答出你在项目中使用 Hibernate 的常用查询数据方式。

我知道的有导航对象图查询，OID 查询，HQL 查询，QBC 查询，本地 SQL 查询。我在项目中用的最多的是 HQL 查询和 QBC 查询。

HQL(重要，项目中常用，希望能够流利说出)

1 HQL :hibernate 提供一种查询语言，一般通过创建 **query 对象**进行 hql 语句的操作。HQL 语言和普通 sql 很相似，**区别：普通 sql 操作数据库表和字段，HQL 操作实体类和属性**

2 常用的 HQL 语句<这里希望同学们用自己的话进行组织，说下使用 HQL 在项目中一般如何查询的>

- (1) 查询所有：from 实体类名称
- (2) 条件查询：from 实体类名称 where 属性名称=?
- (3) 排序查询：from 实体类名称 order by 实体类属性名称 asc/desc
- (4) 分页查询：**关键的两个设置是一个是 setfirstResult()<设置起始页从第几条开始>另一个是 setMaxResult()<设置每页最大显示多少条>。**

3 使用 HQL 查询操作时候，使用 Query 对象

- (1) 创建 Query 对象，写 HQL 语句
- (2) 调用 query 对象里面的方法得到结果

QBC<重要，项目中常用，希望能够流利说出>

使用 HQL 查询需要写 HQL 语句实现，但是使用 QBC 时候，不需要写语句了，使用方法实现

我在项目中常用的 QBC 方法有：<这里答的时候不要答的过于详细，只要你能够说出你使用以下查询方式的时候关键字就好>

- **查询所有** 创建一个 criteria 对象，调用它的 list()方法即可。
- **条件查询** 创建一个 criteria 对象，调用它的 add 方法来实现条件的设置。在这个方法里边我们使用 **Restrictions** 这个类里边的静态方法来实现。比如：
restrictions.eq(),restrictions.like()
- **模糊查询** restrictions.like()
- **排序查询** 创建一个 criteria 对象，调用 addOrder()方法来实现排序。在这个方法里边通过 Order.desc()来实现排序。
- **分页查询** 创建一个 criteria 对象，分页方法和 HQL 的分页方法一样。关键的两个设置是一个是 **setfirstResult()**<设置起始页从第几条开始>另一个是 **setMaxResult()**<设置每页最大显示多少条>。
- **统计查询** 创建一个 criteria 对象，**criteria.setProjection(Projections.rowCount());**
criteria.uniqueResult();

本地 SQL 就像我们之前 jdbc 写 sql 语句一样。Session.createSQLQuery(select * from t_user);

7、get 和 load 区别？

- get 如果没有找到会返回 null，load 如果没有找到会抛出异常。
- get 会先查一级缓存，再查二级缓存，然后查数据库;load 会先查一级缓存，如果没有找到，就创建代理对象，等需要的时候去查询二级缓存和数据库。(这里就体现 load 的延迟加载的特性。)

8、如何优化 Hibernate

- 1.使用双向一对多关联，不使用单向一对多
- 2.不用一对一，用多对一取代
- 3.配置对象缓存，不使用集合缓存
- 4.一对多集合使用 Bag,多对多集合使用 Set
- 5.继承类使用显式多态
- 6.表字段要少，表关联不要怕多，有二级缓存撑腰

我选择这样的问问题的方式，主要是针对企业考察一个人的逻辑性，表达性，技术掌握如何进行考虑。

二：Spring

1、谈谈你对 spring 的了解谈谈及你对 spring 里边 DI ,AOP ,IOC 认识。

- ◆ Spring 是一个封层的一站式开发的轻量级开源框架，通过配置文件中的<bean>元素配置用于创建实例对象的类名和实例对象的属性。
- ◆ Spring 提供支持 **IOC** 和 **AOP** 技术,我们可以将对象之间的依赖关系交由 Spring 进行控制，避免硬编码所造成的程序过度耦合。，通过 AOP 也可以对某个方法进行操作，例如：权限验证
- ◆ Spring 对 Junit4 支持，方便程序测试。
- ◆ 声明式事务支持，只要在配置文件中配置后，不用程序员在代码编写
- ◆ 降低了 JavaEE 的 API 使用难度, Spring 框架对对 JavaEE 开发中非常难用的一些 API (JDBC、JavaMail、远程调用等)，都提供了封装。
- ◆ 同时 Spring 方便集成各种优秀框架，如 Hibernate，struts2。

1.spring 的 **IOC** 控制反转，inverse of control,早期对象通过 new 的方式。通过程序代码创建对象。将创建对象的主动交给给 spring。

通过配置对象信息<bean id="user" class="cn.itcast.domain.User"/>

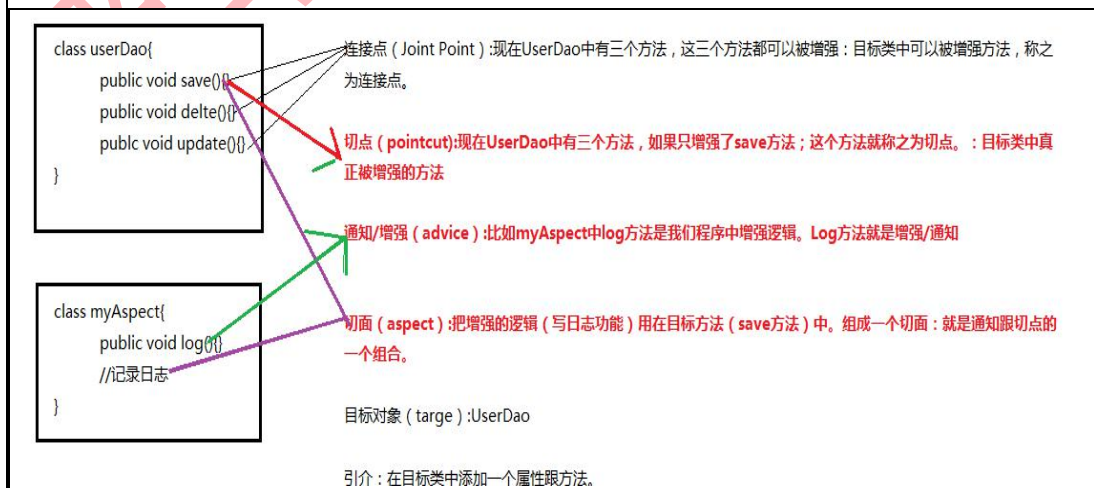
2.spring 的 **DI** 注入属性，Spring 中的配置<bean> </property name="userId" value="85"/> </bean>

注入对象（前提：spring 工厂有一个对象）

<bean><property name="对象中属性" ref="已经存在对象的 bean 的 id"/>

3.spring 的 **AOP** 面向切面编程

- 1、不修改源码就进行方法的扩展
- 2、Aop 的本质横向代码抽取机制
- 3、如果是接口采用 JDK 动态代理，如果是类采用 CGLIB 方式实现动态代理



织入：将增强逻辑用到切点过程

切点：真正被增强的方法就是切点

通知/增强：增强的逻辑就称为通知

切面：切点跟通知组合

1. 谈谈你理解的 spring 的工作流程的理解？

- 创建配置文件 applicationContext.xml
- 编写配置文件（加入一些对象的配置信息）
- Spring 内部采用工厂模式，配合 xml 解析+反射技术，可以根据用户的配置，生成相应的对象
- 工厂提供一个 getBean 方法，从工厂中获取对象
- 操作对象的方法，属性

2. 说说 spring 对象创建的三种方式。

1.无参构造 2.实例工厂 3.静态工厂<关于创建方式的描述，同学们自己组织一下自己的理解进行添加>

3. 不使用注解的情况下，如果给对象注入值的话，你知道的有几种方式？常用的是哪些？

1.set 注值(重点) 2.构造器注值 3.P 名称空间注值 4.Spel 注值 5.注入 java 复杂类型 6.注入对象类型(重点)

我常用的是 set 注值和对象类型注值。

Set 我们通过配置文件给对象(User)赋值,就相当于我们使用对象里边属性的 set 方法给对象设置值。

```
<bean id="lw" class="cn.itcast.domain.User">
  <property name="userId" value="300"></property>
  <property name="username" value="老王"></property>
</bean>
```

```
}
public void setUserId(Long userId) {
    System.out.println("调用了User的setUserId");
    this.userId = userId;
}
```

```
@Test
public void testSpringFactory(){
    //读取配置文件，加载spring的工厂
    ApplicationContext ac = new ClassPathXmlApplicationContext("applicationContext.xml");
    //获取对象--参数：配置文件中对象唯一标识Id
    User user = (User) ac.getBean("lw");
    //操作对象中方法，属性
    System.out.println(user.getUsername()+" "+user.getId());
}
```

对象类型注入 使用 ref 表示注入一个已经存在的对象（默认还是调用 set 方法注入）

4. Spring 有注解方式和 xml 配置两种，两者有什么区别？你常用 spring 注解有哪些？

我在公司主要使用的是注解开发，能够提高开发效率，在配置文件中把相关的注解扫描配置好，我们只要在自己写的类或者代码上边添加对应的注解就把事情解决了。

注解开发

1) 创建对象

导入 jar 包（IOC 基本）spring-aop-4.2.4.jar

导入约束 context

开启注解扫描<context component-scan base-package="包名">

在要创建的对象上使用注解@Component @Repository @Service

@Controller

如果要产生对象是多例@Scope(value="prototype")

2)注入对象

在 service 有 dao 的属性

在 dao 的属性上加注解 @Autowired 按类型注入

@Reosource (name="") 按 bean 的 id 注入

5. Spring 里面 applicationContext.xml 文件能不能改成其他文件名？

能。ContextLoaderListener 是一个 ServletContextListener，它在你的 web 应用启动的时候初始化。缺省情况下，它会在 WEB-INF/applicationContext.xml 文件找 Spring 的配置。你可以通过定义一个<context-param>元素名字为"contextConfigLocation"来改变 Spring 配置文件的位置。


```
1. <listener>
2.   <listener-class>org.springframework.web.context.ContextLoaderListener
3.   <context-param>
4.     <param-name>contextConfigLocation</param-name>
5.     <param-value>/WEB-INF/xyz.xml</param-value>
6.   </context-param>
7. </listener-class>
8. </listener>
```

6. 谈谈 spring 的事务管理

spring 提供的事务管理可以分为两类：编程式的和声明式的。编程式的，比较灵活，但是代码量大，存在重复的代码比较多；声明式的比编程式的更灵活。

编程式 主要使用 transactionTemplate。省略了部分的提交，回滚，一系列的事务对象定义，需注入事务管理对象。

声明式：使用 TransactionProxyFactoryBean:PROPAGATION_REQUIRED

PROPAGATION_REQUIRED PROPAGATION_REQUIRED,readOnly

围绕 Proxy 的动态代理 能够自动的提交和回滚事务

三：Struts2

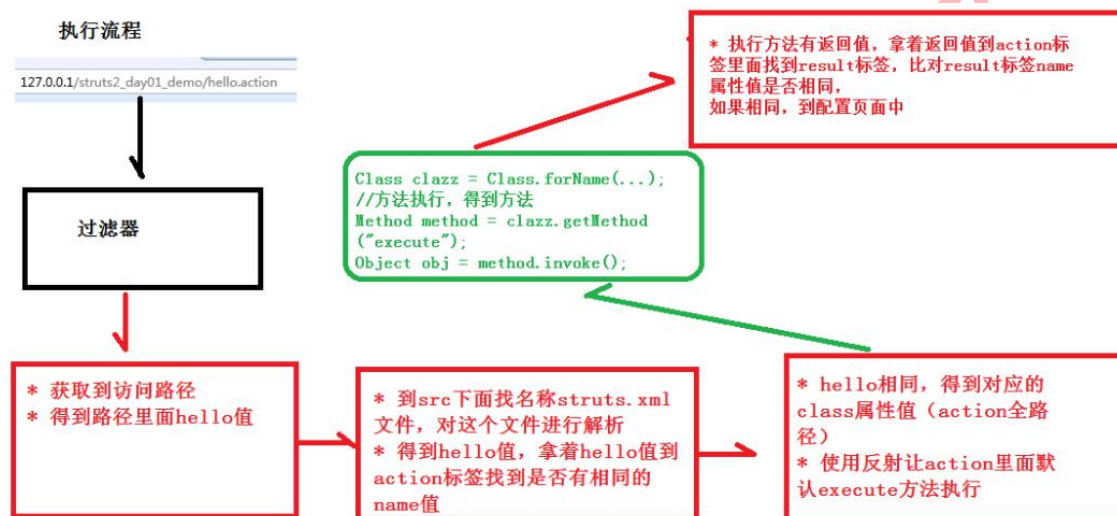
1、谈谈你对 struts2 的理解。

- struts2 框架是一个按照 MVC 设计模式设计的 WEB 层框架，是在 struts 1 和 WebWork 的技术基础上进行了合并的全新的框架。其全新的 Struts 2 的体系结构与 Struts 1 的体系结构差别巨大。Struts 2 以 WebWork 为核心，采用拦截器的机制来处理用户的请求，这样的设计也使得业务逻辑控制器能够与 Servlet API 完全脱离开。
- 我们可以把 struts2 理解为一个大大的 servlet，而这个 servlet 就是 ActionServlet。struts2 在处理客户端请求时，会先读取 web.xml 配置文件，根据前端控制器将符合条件的请求分给各个不同的 Action 处理。在此之前，会把 ActionServlet 会把数据封装成一个 javaBean。
- struts2 框架提供了许多的拦截器，在封装数据的过程中，我们可以对数据进行一些操作，例如：数据校验等等。

- 当 Action 执行完后要返回一个结果视图，这个结果视图可以跟据 struts2 的配置文件中配置，选择转发或者重定向。

2、Struts2 的配置流程。

请求(.action)---->经过 StrutsPrepareAndExecuteFilter 核心控制器---->进入到 Struts2 的拦截器 Interceptor(实现代码功能)----->通过 action 的名称找对应的 Action 类----->执行 Action 类的 execute 方法----->通过 execute 方法中返回的字符串，在 Struts.xml 中找对应的结果页面(result) 【在 action 执行之前，执行了 *defaultStack* 拦截器栈】



3、struts 里边的 Action 配置的注意事项都有哪些？

- ①name 包名称，在 struts2 的配置文件中，包名不能重复，name 并不是真正包名，只是为了管理 Action
- ②namespace 和 <action> 的 name 属性，决定 Action 的访问路径（以/开始）
- ③extends 继承哪个包，通常开发中继承 struts-default 包（struts-default 包在 struts-default.xml 中定义）【可以使用包中默认的拦截器和结果集】

4、拦截器和过滤器有什么区别？

- * 拦截器是基于 java 的反射机制的，而过滤器是基于函数回调
- * 拦截器不依赖与 servlet 容器，而过滤器依赖与 servlet 容器
- * 拦截器只能对 action 请求起作用，而过滤器则可以对几乎所有的请求起作用
- * 拦截器可以访问 action 上下文、值栈里的对象，而过滤器不能

* 在 action 的生命周期中，拦截器可以多次被调用，而过滤器只能在容器初始化时被调用一次

* 拦截器：是在面向切面编程的就是在你的 service 或者一个方法，前调用一个方法，或者在方法后调用一个方法比如动态代理就是拦截器的简单实现，在你调用方法前打印出字符串（或者做其它业务逻辑的操作），也可以在你调用方法后打印出字符串，甚至在你抛出异常的时候做业务逻辑的操作。

5、Struts2 封装获取表单数据的方式有几种？谈谈你熟悉的那种。

(1) 属性封装 (2) 模型驱动封装（重点） (3) 表达式封装

表达式封装和模型驱动封装比较:

- 相同点：都可以把数据封装到实体类对象里面
- 不同点：模型驱动只能封装到一个实体类对象，使用表达式封装可以封装到多个实体类对象里面

使用模型驱动封装

1 使用模型驱动方式，可以直接把表单数据封装到实体类对象里面

2 实现步骤

(1) action 实现接口 ModelDriven (2) 实现接口里面的方法 getModel 方法 把创建对象返回 (3) 在 action 里面创建实体类对象

3 使用模型驱动和属性封装注意问题：

(1) 对同一个表单进行操作，属性封装和模型驱动封装不能同时使用。执行一个封装，执行模型驱动封装，不会执行属性封装。

6、struts2 里边的值栈你是如何使用的？如何向值栈中放值，如何从值栈中取值？

在 struts2 里面提供本身一种存储机制，类似于域对象，是值栈，可以存值和取值。在 action 里面把数据放到值栈里面，在页面中获取到值栈数据。

1 获取值栈对象有多种方式

(1) 常用方式：使用 ActionContext 类里面的方法得到值栈对象

```
ActionContext context = ActionContext.getContext();
```

```
ValueStack stack = context.getValueStack();
```

```
Stack.set("username", "小明");
```

2 每个 action 对象中只有一个值栈对象

如何向值栈放值呢？

放值包括两类 分别是：向值栈中放对象和向值栈中放 list。

放对象的步骤是：

第一步 定义对象变量

第二步 生成变量的 get 方法

第三步 在执行的方法里面面向对象中设置值。

放 list 的步骤是：

第一步 定义 list 集合变量

第二步 生成变量的 get 方法

第三步 在执行的方法里面面向 list 集合设置值

如何从值栈取值呢？

使用 struts2 的标签+ognl 表达式获取值栈数据 <s:property value="ognl 表达式"/>例如 :<s:property value="username"/>

7、Struts2 里边的 OGNL 表达式里边的#、%分别是做什么的？有什么区别？

1. 使用#获取 context 里面数据

```
<s:iterator value = "list" var="user">
```

```
<s:property value = "#user.username">
```

```
</s:iterator>
```

2. (1) 向 request 域放值(获取 context 里面数据，写 ognl 时候，首先添加符号#context 的 key 名称.域对象名称)

(2) 在页面中使用 ognl 获取

```
<s:property value = "#request.req">
```

3. %在 struts2 标签中表单标签

(1) 在 struts2 标签里面使用 ognl 表达式，如果直接在 struts2 表单标签里面使用 ognl 表达式不识别，只有%之后才会识别。

```
<s:textfield name="username" value="%{#request.req}">
```

8、你在开发中，值栈主要有哪些应用？

值栈主要解决 Action 向 JSP 传递数据问题

Action 向 JSP 传递数据处理结果，结果数据有两种形式

1) 消息 String 类型数据

this.addFieldError("msg", "字段错误信息"); (常用于表单校验)

this.addActionError("Action 全局错误信息"); (普通错误信息，不针对某一个字段 登陆失败)

this.addActionMessage("Action 的消息信息"); (通用消息)

在 jsp 中使用 struts2 提供标签 显示消息信息

```
<s:fielderror fieldName="msg"/> <s:actionerror/> <s:actionmessage/>
```

2) 数据 (复杂类型数据)

每次请求，访问 Action 对象会被压入值栈。1) DefaultActionInvocation 的 init 方法

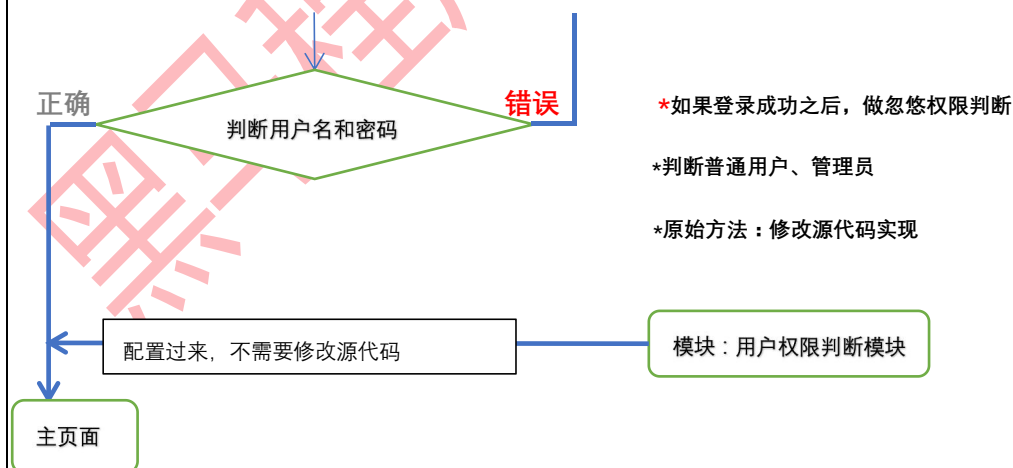
stack.push(action); Action 如果想传递数据给 JSP，只有将数据保存到成员变量，并且提供 get 方法就可以了

9、你会使用 struts2 自定义拦截器么？谈谈其中的原理。

拦截器底层使用了两个原理：

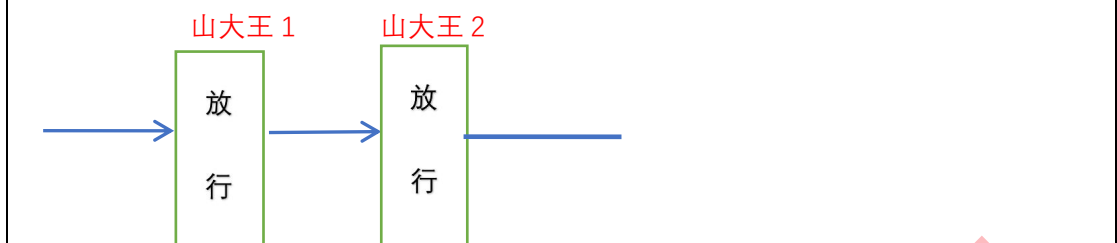
一个是 AOP 思想，一个是责任链模式。

使用 AOP 思想可以达到不修改源码的基础上实现功能的扩展。这一功能的体现如下图所示



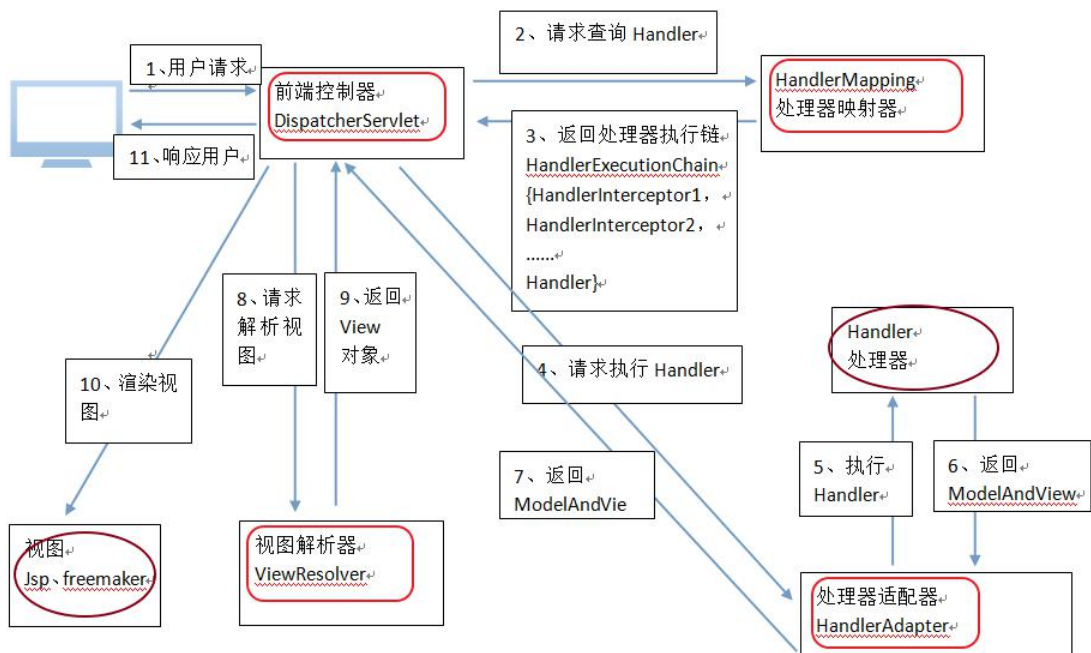
责任链的模式 有点类似于我们的过滤器。比如：要执行多个操作，有添加、修改、删除三个操作。首先执行添加操作，添加操作执行之后 做类似于放行操作，执行修改操作，修改

操作执行之后做类似于放行操作，执行删除操作。过滤链：一个请求可有多个过滤器进行过滤，每个过滤器只有做放行才能到下一个过滤器。如下图：



四：SpringMVC

1、SpringMVC 的执行流程是什么？请简单阐述一下。



2、你常用的 SpringMVC 注解有哪些？如何开启 SpringMVC 的注解扫描？

常用的注解有：@RequestMapping 用于请求 url 映射。

@RequestBody 注解实现接收 http 请求的 json 数据，将 json 数据转换为 java 对象。

@ResponseBody 注解实现将 controller 方法返回对象转化为 json 响应给客户。

其他的注解希望学生根据自己理解情况自行补充。

3、如何开启注解处理器和适配器的配置？

我们在项目中一般会在 springmvc.xml 中通过开启 <mvc:annotation-driven>来实现注解处理器和适配器的开启。

4、使用 springMVC 框架的时候，如何解决 post 和 get 的乱码问题？

解决 post 请求乱码：我们可以在 web.xml 里边配置一个 CharacterEncodingFilter 过滤器。设置为 utf-8。

解决 get 请求的乱码：有两种方法。对于 get 请求中文参数出现乱码解决方法有两个：

1.修改 tomcat 配置文件添加编码与工程编码一致。

2.另外一种方法对参数进行重新编码 String userName = New String(Request.getParameter("userName").getBytes("ISO8859-1"), "utf-8");

5、谈谈你对 restful 的理解及项目中的应用。

RESTful 架构，就是目前最流行的一种互联网软件架构。它结构清晰、符合标准、易于理解、扩展方便，所以正得到越来越多网站的采用。我简单总结一下什么是 RESTful 架构：

(1) 每一个 URI 代表一种资源；

(2) 客户端和服务端之间，传递这种资源的某种表现层；

(3) 客户端通过四个 HTTP 动词，对服务器端资源进行操作，实现"表现层状态转化"。

GET 用来获取资源，POST 用来新建资源（也可以用于更新资源），PUT 用来更新资源，DELETE 用来删除资源。

6、SpringMVC 与 struts2 的区别？

1、springmvc 的入口是一个 servlet 即前端控制器，而 struts2 入口是一个 filter 过滤器。

2、springmvc 是基于方法开发，传递参数是通过方法形参，可以设计为单例或多例(建议单例)，struts2 是基于类开发，传递参数是通过类的属性，只能设计为多例。

3、Struts 采用值栈存储请求和响应的数据，通过 OGNL 存取数据，springmvc 通过参数解析器是将 request 对象内容进行解析成方法形参，将响应数据和页面封装成 ModelAndView 对象，最后又将模型数据通过 request 对象传输到页面。Jsp 视图解析器默认使用 jstl。

4、Struts2 有漏洞，springmvc 目前还没有漏洞出现。如果使用 struts2，建议下载最新包。

五：Mybatis

1、MyBatis 编程步骤是什么样的？

- 1、创建 SqlSessionFactory
- 2、通过 SqlSessionFactory 创建 SqlSession
- 3、通过 sqlSession 执行数据库操作
- 4、调用 session.commit()提交事务
- 5、调用 session.close()关闭会话

2、Mybatis 和 Hibernate 的区别。

mybatis :

1. 入门简单，即学即用，提供了数据库查询的自动对象绑定功能，而且延续了很好的 SQL 使用经验，对于没有那么高的对象模型要求的项目来说，相当完美。
2. 可以进行更为细致的 SQL 优化，可以减少查询字段。
3. 缺点就是框架还是比较简陋，功能尚有缺失，虽然简化了数据绑定代码，但是整个底层数据库查询实际还是要自己写的，工作量也比较大，而且不太容易适应快速数据库修改。
4. 二级缓存机制不佳。

hibernate :

1. 功能强大，数据库无关性好，O/R 映射能力强，如果你对 Hibernate 相当精通，而且对 Hibernate 进行了适当的封装，那么你的项目整个持久层代码会相当简单，需要写的代码很少，开发速度很快，非常爽。
2. 有更好的二级缓存机制，可以使用第三方缓存。
3. 缺点就是学习门槛不低，要精通门槛更高，而且怎么设计 O/R 映射，在性能和对象模型之间如何权衡取得平衡，以及怎样用好 Hibernate 方面需要你的经验和能力都很强才行。

3、JDBC 编程有哪些不足之处 ,MyBatis 是如何解决这些问题的？

① 数据库链接创建、释放频繁造成系统资源浪费从而影响系统性能，如果使用数据库链接池可解决此问题。

解决：在 SqlMapConfig.xml 中配置数据链接池，使用连接池管理数据库链接。

② Sql 语句写在代码中造成代码不易维护，实际应用 sql 变化的可能较大，sql 变动需要改变 java 代码。

解决：将 Sql 语句配置在 XXXXmapper.xml 文件中与 java 代码分离。

③ 向 sql 语句传参数麻烦，因为 sql 语句的 where 条件不一定，可能多也可能少，占位符需要和参数一一对应。

解决：Mybatis 自动将 java 对象映射至 sql 语句。

④ 对结果集解析麻烦，sql 变化导致解析代码变化，且解析前需要遍历，如果能将数据库记录封装成 pojo 对象解析比较方便。

解决：Mybatis 自动将 sql 执行结果映射至 java 对象。

4、简单的说一下 MyBatis 的一级缓存和二级缓存？

Mybatis 首先去缓存中查询结果集，如果没有则查询数据库，如果有则从缓存取出返回结果集就不走数据库。Mybatis 内部存储缓存使用一个 HashMap，key 为 hashCode+sqlId+Sql 语句。value 为从查询出来映射生成的 java 对象

Mybatis 的二级缓存即查询缓存，它的作用域是一个 mapper 的 namespace，即在同一个 namespace 中查询 sql 可以从缓存中获取数据。二级缓存是可以跨 SqlSession 的。

5、使用 MyBatis 的 mapper 接口调用时有哪些要求？

- ① Mapper 接口方法名和 mapper.xml 中定义的每个 sql 的 id 相同
- ② Mapper 接口方法的输入参数类型和 mapper.xml 中定义的每个 sql 的 parameterType 的类型相同
- ③ Mapper 接口方法的输出参数类型和 mapper.xml 中定义的每个 sql 的 resultType 的类型相同
- ④ Mapper.xml 文件中的 namespace 即是 mapper 接口的类路径。

第九部份：传统项目

1、PowerDesigner (p2p,ERP,BOS,医药采购,杰信)

在[数据库建模](#)的过程中，需要运用 PowerDesigner 进行[数据库设计](#)，这个不但可以让人直观的理解模型，而且可以充分的利用数据库技术，优化数据库的设计。在[数据库系统概](#)

论中有涉及到，这个实体关系图中，一个实体对于一个表，实体、属性与联系是进行[系统设计](#)时要考虑的三个要素，也是一个好的数据库设计的核心。

2、plsql (p2p,ERP,BOS,医药采购,杰信)

PL/SQL Developer 是一个集成开发环境，专门面向 Oracle 数据库存储程序单元的开发 PL/SQL Developer 侧重于易用性、代码品质和生产力，充分发挥 Oracle 应用程序开发过程中的主要优势。

3、AngularJS (p2p)

- 1.完全使用 JavaScript 编写的客户端技术。同其他历史悠久的 Web 技术（HTML、CSS 和 JavaScript）配合使用，使 Web 应用开发比以往更简单、更快杰。
- 2 .AngularJS 主要用于构建单页面 Web 应用。它通过增加开发人员和常见 Web 应用开发任务之间的抽象级别，使构建交互式的现代 Web 应用变得更加简单。
- 3、AngularJS 的开发团队将其描述为一种构建动态 Web 应用的结构化框架。
- 4、AngularJS 使开发 Web 应用变得非常简单，同时也降低了构建复杂应用的难度。它提供了开发者在现代 Web 应用中经常要用到的一系列高级功能，例如：
解耦应用逻辑、数据模型和视图；
Ajax 服务；
[依赖注入](#)；
浏览历史（使书签和前进、后退按钮能够像在普通 Web 应用中一样工作）；

测试；

应用场景：

我们在 p2p 中，我们是应用了 AngularJS 的路由，还有解耦应用逻辑、数据模型和视图，我们通过高级路由功能把页面的嵌套，和转发，合理的应用，让页面看起来更加的流畅，在我们写代码的时候能跟快速的理解流程和业务。

4、Spring data jpa (p2p , 医药, Bos)

Spring JPA 通过为用户统一创建和销毁 EntityManager，进行事务管理，简化 JPA 的配置等使用户的开发更加简便。Spring Data JPA 是在 Spring JPA 的基础上，对持久层做了简化。用户只需声明持久层的接口，不需要实现该接口。Spring Data JPA 内部会根据不同的策略、通过不同的方法创建 Query 操作数据库。

应用场景：

我们使用 spring data jpa 很简单，只需要创建一个 dao 接口，来实现指定的 spring data jpa 提供的接口，不需要做任何其它实现就可以完成持久化操作。

5、Bootstrap (p2p)

是一个做网页的框架（目前最流行的 WEB 前端框架），就是说你只需要写 HTML 标签调用它的类你就可以很快速的做一个高大上的网页，你不用担心兼容问题，提供了很多样式供你选择！比如你需要做一个网站的导航对吧，你自己写的话你需要写很多代码，但是如果你使用 bootstrap 框架来写的话，只需要写好 HTML 标签然后调用类名就可以了！

6、Redis

redis 是一个 key-value [存储系统](#)。和 Memcached 类似类型。

应用 Redis 实现数据的读写，同时利用队列处理器定时将数据写入 mysql。

同时要注意避免冲突，在 redis 启动时去 mysql 读取所有表键值存入 redis 中，往 redis 写数据时，对 redis 主键自增并进行读取，若 mysql 更新失败，则需要及时清除缓存及同步 redis 主键。

这样处理，主要是实时读写 redis，而 mysql 数据则通过队列[异步处理](#)，缓解 mysql 压力，不过这种方法应用场景主要基于高并发，而且 redis 的高可用集群架构相对更复杂，一般不是很推荐。

Redis 有五种数据类型

string(字符串)、

list([链表](#))、

set(集合)、

zset(sorted set --有序集合)

hash (哈希类型)

我们使用 jedis 来通过 java 操作 redis，jedis 是一个比较常用 java 连接 redis 插件

应用场景：

当注册页面加载时，会加载验证码，最终我们的图片验证码会存储到 redis 中，存储的方式 uuid 为 key，验证码为 value

7、短信验证（p2p、杰信）

这里我们调用的是吉信通 来作为我们的发送短信接口，这个工具已经有写好的 java 调用代码，我们只需要在使用的時候去官网 copy 下就可以了

8、ActiveMQ（p2p）

应用程序接口是一个 Java 平台中关于面向消息中间件（MOM）的 API，用于在两个应用程序之间，或分布式系统中发送消息，进行异步通信。Java 消息服务是一个与具体平台无关的 API，绝大多数 MOM 提供商都对 JMS 提供支持。

Java 消息服务的规范包括两种消息模式，点对点 and 发布者 / 订阅者。许多提供商支持这一通用框架因此，程序员可以在他们的分布式软件中实现面向消息的操作，这些操作将具有不同面向消息中间件产品的可移植性。

Java 消息服务支持同步和异步的消息处理，在某些场景下，异步消息是必要的；在其他场景下，异步消息比同步消息操作更加便利。

Java 消息服务支持面向事件的方法接收消息，事件驱动的程序设计现在被广泛认为是一种富有成效的程序设计范例在应用系统开发时，Java 消息服务可以推迟选择面对消息中间件产品，也可以在不同的面对消息中间件切换。

应用场景：

在 p2p 项目中所发送的短信，邮件认证都是通过 ActiveMQ 来监听发送的。

9、Webservice

使用 webservice 可以让我们将我们自己程序功能，发布到网络上，提供给别人使用。

简单理财 webservice,就是通过 xml 来进行数据交换

三大核心:

Soap:它是一种协议

WSDL:网络服务的描述语言

Uddi

开发 webservice, 可以使用 java 直接实现:

在实际开发中我们一般使用 webservice 的框架 cxf 框架。

应用场景：

在 p2p 项目中, 我们的充值业务需要调用银行的接口, 我们通过 webservice 来调用银行的接口给我们的客户充值金额到我们的平台上, 完成金额充值业务。

10、Shiro

Shiro 是 apache 的一个权限框架, 它的主要功能认证, 授权.

认证: authentication 可以简单理解认证就是用户是否登录

授权: authorization 用户登录后, 它是否具有权限操作某些功能

注意: shiro 框架它不维护认证及授权数据。

对于一个基本权限操作：

1. 用户表-----一个用户可以具有多个角色
2. 角色表----一个角色可以有多个权限, 一个角色可以给多个用户
3. 权限表---一个权限可以赋予多个角色
4. 用户与角色的关联表
5. 角色与权限的关联表

总结: 用户----角色 角色---权限 都是多对多关系

应用场景：

在项目中我们通过 shiro 的权限控制让项目在访问一些需要经过登录之后才能访问, 才能看到的内容, 给页面形成了一个保护墙, 给我们这个项目系统安全做了保护作用。

11、Easyui (Bos、ERP)

jQuery EasyUI 是一组基于 jQuery 的 UI 插件集合体, 而 jQuery

EasyUI 的目标就是帮助 web 开发者更轻松的打造出功能丰富并且美观的 [UI 界面](#)。开发者不

需要编写复杂的 javascript，也不需要 css 样式

有深入的了解，开发者需要了解的只有一些简单的 [html 标签](#)。

框架的优势我想应该是快速开发，减少项目的开发周期，从而节省项目成本，因为那么多组件，需要一个比较长的开发周期，而且还要投入一些比较牛的技术人员才可以开发。而且框架的 *api* 做得还不错，优势就更加体现出来了。

一般都用在后台管理系统的开发。

应用场景：

我们的 BOS,erp 的后台页面就是 easyui 开发的，我们不需要太复杂跟太漂亮的界面，easyui 完全满足的需求，而且用 easyUi 开发跟快的加速了我们的开发期限。

12、Poi (p2p、BOS、ERP)

项目中经常要解析和生成 Excel 文件，最常用的开源组件有 poi 与 jxl。jxl 是韩国人开发的，发行较早，但是更新的很慢，目前似乎还不支持 excel2007。[poi](#) 是 apache 下的一个子项目，poi 应该是处理 ms 的 office 系列文档最好的组件了。

应用场景：

我们在项目中经常要导出报表，所以我们应用了 poi 来操作。方便，快杰，简单。

13、MD5

MD5，被广泛用于加密和解密技术上，它可以说是文件的“数字指纹”。任何一个文件，无论是可执行程序、图像文件、临时文件或者其他任何类型的文件，也不管它体积多大，都有且只有一个独一无二的 MD5 信息值，并且如果这个文件被修改过，它的 MD5 值也将随之改变。因此，我们可以通过对比同一文件的 MD5 值，来校验这个文件是否被“篡改”过。

应用场景：

在我们这些项目中，文件的登录密码都进行了 MD5 加密，为了保护用户的信息不会被泄露，就算在数据库中我们看到的用户密码也是经过加密的。

14、JavaMail (p2p、杰信)

JavaMail 是由 Sun 定义的一套收发电子邮件的 API，不同的厂商可以提供自己的实现类。但它并没有包含在 JDK 中，而是作为 JavaEE 的一部分。

厂商所提供的 JavaMail 服务程序可以有选择地实现某些邮件协议，常见的邮件协议包括：

SMTP：简单邮件传输协议，用于发送电子邮件的传输协议；

POP3：用于接收电子邮件的标准协议；

IMAP：互联网消息协议，是 POP3 的替代协议。

这三种协议都有对应 SSL 加密传输的协议，分别是 SMTPS，POP3S 和 IMAPS。

除 JavaMail 服务提供程序之外，JavaMail 还需要 JAF(JavaBeans Activation Framework)来处理不是纯文本的邮件内容，这包括 MIME（多用途互联网邮件扩展）、URL 页面和文件附件等内容。

应用场景：

JavaMail 在项目中我们的邮箱验证就是通过它来实现的，每个运营商有自己的运营邮箱，我们在开发的时候只要按要求填好各大邮箱的运营商信息就可以实现邮箱验证功能。

15、Quartz (杰信)

Quartz 是一个开源的作业调度框架，它完全由 Java 写成，并设计用于 J2SE 和 J2EE 应用中。它提供了巨大的灵活性而不牺牲简单性。你能够用它来为执行一个作业而创建简单的或复杂的调度。

应用场景：

当指定的交期到时，给相关人员发送一封邮件进行提醒，需要与工厂进行联系

定时查询库存预警信息，如果存在库存预警信息，发送邮件通知给相关工作人员。

16、Amchart (杰信)

mchart 是一组由 flash 做成的图表组件。这些 flash 没有数据，一旦指定了配置文件和数据文件的地址，flash 就可以显示出你所需要的图表。

在配置文件中可以设这个 flash 的颜色，大小，文字属性，柱子，曲线的粗细颜色，是什么类型的柱子或曲线，是否有气泡，是否有图例，定位，鼠标右键及其事件等，stock 图还要在里面指定数据文件的地址。

在数据文件中存放了要展示的数，数据一般是 XML 类型的数据，也有用 CSV 或 TXT 的类型的数据。

这里所指的文件不一定是一个真实的文件，也可以是同一个域(指访问的这个 flash 的域名)内部的 URL，跨域 URL 要是全名。在我们的系统中大部分是用 flash 所在域的同域中。

应用场景：

在项目中我们使用 Amchart+flash 主要是想把我们一些数据通过动漫效果展示出来。让这些数据在光看的时候跟直观一些。

17、FreeMarker

FreeMarker 是一个模板引擎,一个基于模板生成文本输出的通用工具,使用纯 **Java** 编写。FreeMarker 被设计用来生成 HTMLWeb 页面，特别是基于 MVC 模式的应用程序。

所谓模板，就是一份已经写好了基本内容，有着固定格式的文档，其中空出或者用占位符标识的内容，由使用者来填充，不同的使用者给出的数据是不同的。在模板中的占位符，在模板运行时，由模板引擎来解析模板，并采用动态数据替换占位符部分的内容。

FreeMarker 不是一个 Web 应用框架，而适合作为 Web 应用框架一个组件，FreeMarker 与 Web 容器无关，即在 Web 运行时，它并不知道 Servlet 或 HTTP。它不仅可以用作表现层的实现技术，而且还可以用于生成 XML，JSP 或 Java 文件等。

虽然 FreeMarker 具有一些编程的能力，但通常由 Java 程序准备要显示的数据，由 FreeMarker 生成页面，通过模板显示准备的数据

18、Maven (p2p,ERP,BOS,医药采购,杰信)

Maven 是基于中央仓库的编译，即把编译所需要的资源放在一个中央仓库里，如 jar，tld，pom，等。当编译的时候，maven 会自动在仓库中找到相应的包，如果本地仓库没有，则从设定好的远程仓库中下载到本地。这一切都是自动的

19、分布式开发：

分布式系统往往是把应用拆分成多个应用，每个团队维护一个应用，应用与应用通过远程过程调用或者消息中间件通信。这种系统的优点是能够做到高内聚低耦合，可以支撑业务的快速发展，缺点则是运维成本大大提高了，系统出了问题，需要全链路排查

20、SVN (p2p,BOS,医药采购)

SVN = 版本控制 + 备份服务器简单的说，您可以把 SVN 当成您的备份服务器，更好的是，他可以帮您记住每次上传到这个服务器的档案内容。并且自动的赋予每次的变更一个版本。

版本控制解决了：

- *代码管理混乱
- *解决代码冲突困难
- *在代码整合期间引发 bug
- *无法对代码的拥有者进行权限控制
- *项目不同版本的发布困难

21、Git (BOS)

Git 是分布式版本工具（除了具有远程仓库外，还具有本地仓库 可以在离线情况下进行版本控制）

SVN 要是中央仓库，所有版本控制信息在中央仓库，客户端无任何版本控制信息，SVN 必须基于中央仓库进行控制，如果无法联网，无法进行版本控制

Git 基于中央仓库，进行克隆，本地存在一个仓库，客户端可以基于本地仓库进行版本控制，即使在没有网络情况下，也可以基于本地仓库进行控制

第十部份：电商项目

1、Dubbo

1. Dubbo 是什么？

Dubbo 是一个分布式服务框架，致力于提供高性能和透明化的 **RPC** 远程服务调用方案，以及 **SOA** 服务治理方案。

其核心部分包含：

- **远程通讯**: 提供对多种基于长连接的 NIO 框架抽象封装，包括多种线程模型，序列化，以及“请求-响应”模式的信息交换方式。
- **集群容错**: 提供基于接口方法的透明远程过程调用，包括多协议支持，以及软负载均衡，失败容错，地址路由，动态配置等集群支持。
- **自动发现**: 基于注册中心目录服务，使服务消费方能动态的查找服务提供方，使地址透明，使服务提供方可以平滑增加或减少机器。

2. Dubbo 能做什么？

- 透明化的远程方法调用，就像调用本地方法一样调用远程方法，只需简单配置，没有任何 API 侵入。
- 软负载均衡及容错机制，可在内网替代 F5 等硬件负载均衡器，降低成本，减少单点。
- 服务自动注册与发现，不再需要写死服务提供方地址，注册中心基于接口名查询服务提供者的 IP 地址，并且能够平滑添加或删除服务提供者。

3. Dubbo 服务开发流程，运行流程？Zookeeper 注册中心的作用？

使用流程：

第一步：要在系统中使用 dubbo 应该先搭建一个注册中心，一般使用 zookeeper。

第二步：有了注册中心然后是发布服务，发布服务器要使用 spring 容器和 dubbo 标签来发布任务，并且发布服务时需要指定注册中心的位置。

第三步：服务发布之后就是调用服务。一般调用服务也是使用 spring 容器和 dubbo 标签来引用服务，这样就可以在客户端的容器中生成一个服务的代理对象，在 action 或者 Controller 中直接调用 service 的方法即可。

Zookeeper 注册中心的作用：主要是注册和发现服务的作用。类似于房产中介的作用，在系统中并不参与服务的调用及数据的传输。

2、FastDFS

1. 什么是 FastDFS？

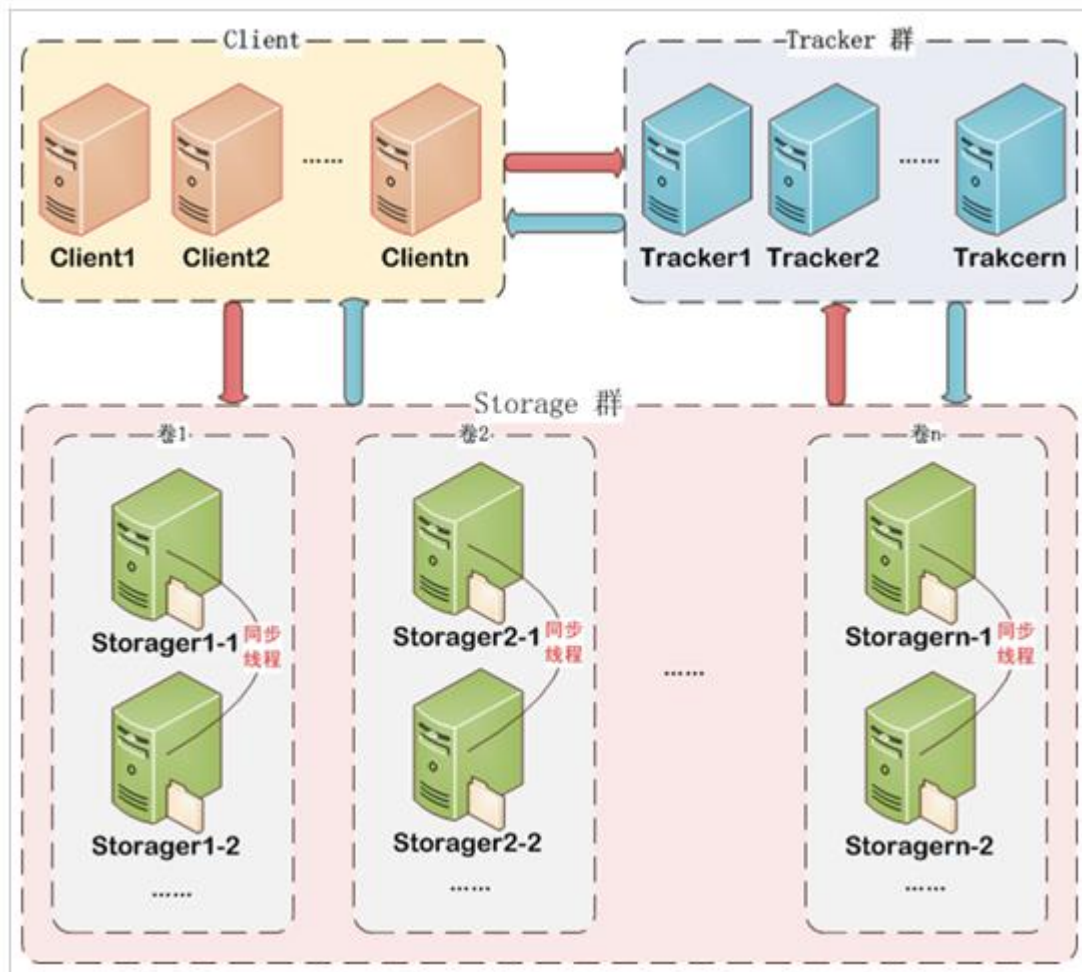
FastDFS 是用 c 语言编写的一款开源的分布式文件系统。FastDFS 为互联网量身定制，充分考虑了冗余备份、负载均衡、线性扩容等机制，并注重高可用、高性能等指标，使用 FastDFS 很容易搭建一套高性能的文件服务器集群提供文件上传、下载等服务。

2. FastDFS 架构

FastDFS 架构包括 Tracker server 和 Storage server。客户端请求 Tracker server 进行文件上传、下载，通过 Tracker server 调度最终由 Storage server 完成文件上传和下载。

Tracker server 作用是负载均衡和调度，通过 Tracker server 在文件上传时可以根据一些策略找到 Storage server 提供文件上传服务。可以将 tracker 称为追踪服务器或调度服务器。

Storage server 作用是文件存储，客户端上传的文件最终存储在 Storage 服务器上，Storage server 没有实现自己的文件系统而是利用操作系统的文件系统来管理文件。可以将 storage 称为存储服务器。



服务端两个角色：

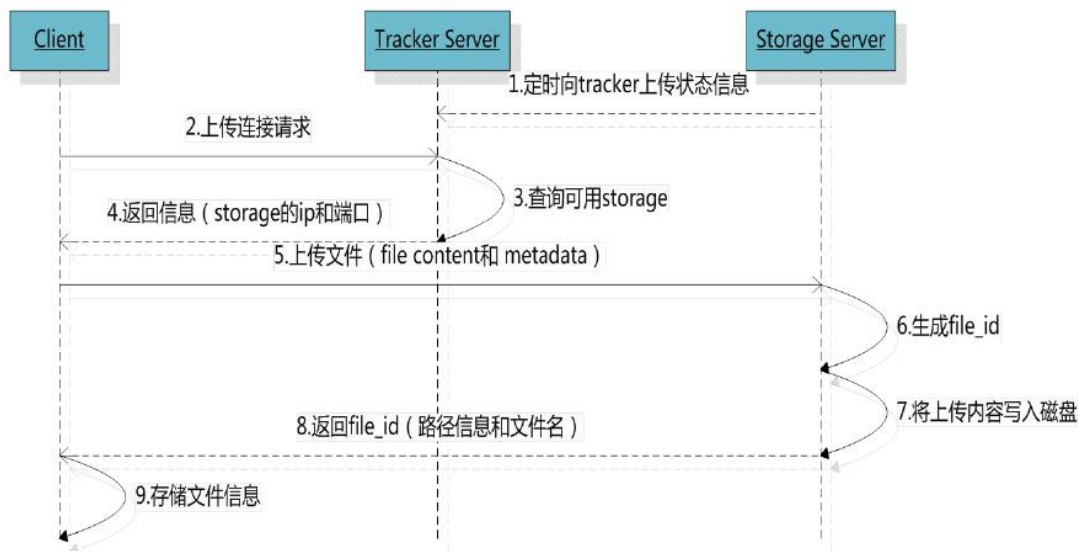
Tracker：管理集群，tracker 也可以实现集群。每个 tracker 节点地位平等。

收集 Storage 集群的状态。

Storage：实际保存文件

Storage 分为多个组，每个组之间保存的文件是不同的。每个组内部可以有多个成员，组成员内部保存的内容是一样的，组成员的地位是一致的，没有主从的概念。

3. 文件上传的流程



客户端上传文件后存储服务器将文件 ID 返回给客户端，此文件 ID 用于以后访问该文件的索引信息。文件索引信息包括：组名，虚拟磁盘路径，数据两级目录，文件名。

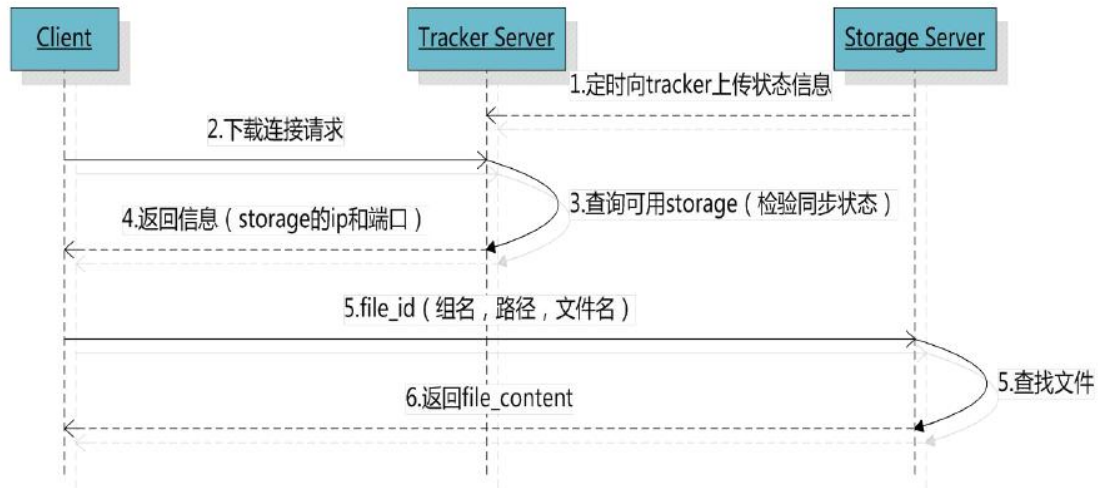
组名：文件上传后所在的 storage 组名称，在文件上传成功后有 storage 服务器返回，需要客户端自行保存。

虚拟磁盘路径：storage 配置的虚拟路径，与磁盘选项 store_path* 对应。如果配置了 store_path0 则是 M00，如果配置了 store_path1 则是 M01，以此类推。

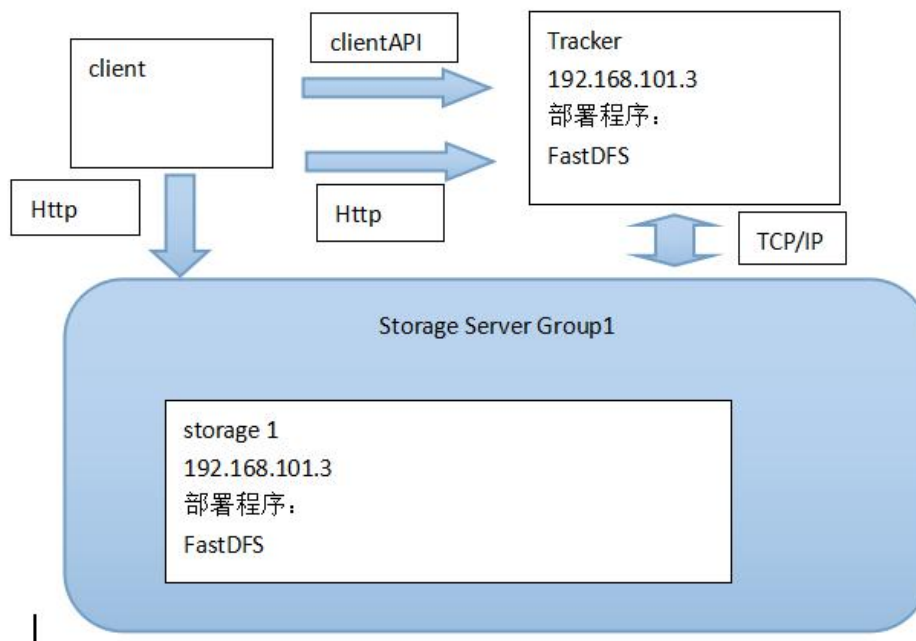
数据两级目录：storage 服务器在每个虚拟磁盘路径下创建的两级目录，用于存储数据文件。

文件名：与文件上传时不同。是由存储服务器根据特定信息生成，文件名包含：源存储服务器 IP 地址、文件创建时间戳、文件大小、随机数和文件拓展名等信息。

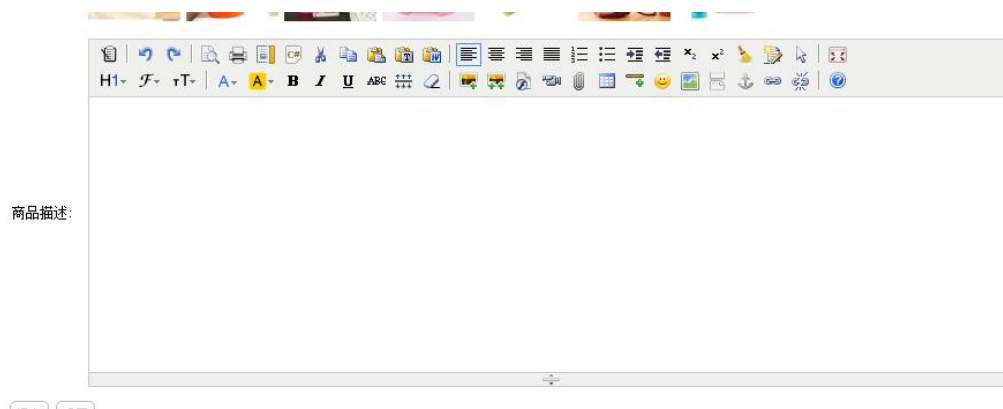
4. 文件下载



5. 最简单的 FastDFS 架构



3、富文本编辑器



1. KindEditor：

KindEditor 使用 JavaScript 编写，可以无缝的于 Java、.NET、PHP、ASP 等程序
接合。KindEditor 非常适合在 CMS、商城、论坛、博客、Wiki、电子邮件等互联网应
用上使用，2006 年 7 月首次发布 2.0 以来，KindEditor 依靠出色的用户体验和领先
的技术不断扩大编辑器市场占有率，目前在国内已经成为最受欢迎的编辑器之一。

主要特点：

1. 体积小，加载速度快，但功能十分丰富。
2. 内置自定义 range，完美地支持 span 标
记。
3. 基于插件的方式设计，所有功能都是插件，增加自定义和扩展功能非常简单。
4. 修改编辑器风格很容易，只需修改一个 CSS 文件。
5. 支持大部分主流浏览器，比如 IE、Firefox、Safari、Chrome、Opera。

2. UEditor：百度编辑器

UEditor 是由百度 WEB 前端研发部开发的所见即所得的开源富文本编辑器，具有轻量、
可定制、用户体验优秀等特点。开源基于 BSD 协议，所有源代码在协议允许范围内可自由
修改和使用。百度 UEditor 的推出，可以帮助不少网站开发者在开发富文本编辑器所遇到的
难题，节约开发者因开发富文本编辑器所需要的大量时间，有效降低了企业的开发成本。

优点：

- 1、体积小巧，性能优良，使用简单
- 2、分层架构，方便定制与扩展

- 3、满足不同层次用户需求，更加适合团队开发
- 4、丰富完善的中文文档
- 5、多个浏览器支持：Mozilla, MSIE, [FireFox](#), [Maxthon](#), Safari 和 Chrome
- 6、更好的使用体验
- 7、拥有专业 QA 团队持续支持，已应用在[百度](#)各大产品线上

注：纯 js 开发，跟后台语言没有关系。

4、Redis

redis 是一个 key-value 存储系统。和 Memcached 类似，它支持存储的 value 类型相对更多，包括 string(字符串)、list(链表)、set(集合)、zset(sorted set --有序集合)和 hash（哈希类型）。这些数据类型都支持 push/pop、add/remove 及取交集并集和差集及更丰富的操作，而且这些操作都是原子性的。在此基础上，redis 支持各种不同方式的排序。与 memcached 一样，为了保证效率，数据都是缓存在内存中。区别的是 redis 会周期性的把更新的数据写入磁盘或者把修改操作写入追加的记录文件，并且在此基础上实现了 master-slave(主从)同步。

Redis 是一个高性能的 key-value 数据库。redis 的出现，很大程度补偿了 memcached 这类 key/value 存储的不足，在部分场合可以对关系数据库起到很好的补充作用。它提供了 Java, C/C++, C#, PHP, JavaScript, Perl, Object-C, Python, Ruby, Erlang 等客户端，使用很方便。

Redis 支持主从同步。数据可以从主服务器向任意数量的从服务器上同步，从服务器可以是关联其他从服务器的主服务器。这使得 Redis 可执行单层树复制。存盘可以有意无意的对数据进行写操作。由于完全实现了发布/订阅机制，使得从数据库在任何地方同步树时，可订阅一个频道并接收主服务器完整的消息发布记录。同步对读取操作的扩展性和数据冗余很有帮助。

1. Redis 的优点？

- 1.性能极高 – Redis 能支持超过 100K+ 每秒的读写频率。
- 2.丰富的数据类型 – Redis 支持二进制案例的 Strings, Lists, Hashes, Sets 及 Ordered Sets 数据类型操作。
- 3.原子 – Redis 的所有操作都是原子性的，同时 Redis 还支持对几个操作全并后的原子性执行。
- 4.丰富的特性 – Redis 还支持 publish/subscribe, 通知, key 过期等等特性

2. Redis 的缺点？

是数据库容量受到物理内存的限制,不能用作海量数据的高性能读写,因此 Redis 适合的场景主要局限在较小数据量的高性能操作和运算上。

3. Redis 持久化

3.1、RDB 持久化：

该机制是指在制定的时间间隔内将内存中的数据快照写入磁盘。

优点：1. 只有一份 rdb 文件，可随时备份

2. 比 AOF 文件小，加载效率高

3. 只提供 fork 子进程，不阻塞主进程，IO 操作比较少

3.2、AOF 持久化：

该机制将以日志的形式记录服务器所处理的每一个写操作，在 Redis 服务器启动之初 会读取该文件来重新构建数据库，以保证启动后数据库中的数据是完整的。

优点：1. 每次改动同步数据安全性好

2. APPEND 方式追加日志，不会对旧日志文件产生影响

3.3、无持久化：

我们可以通过配置的方式禁用 Redis 服务器的持久化功能，这样我们就可以将 Redis 视为一个功能加强版的 memcached 了

3.4、同时应用 AOF 和 RDB

4、Redis 集群

群指的是将几台服务器集中在一起，实现同一业务

1. 目的：高可用、负载均衡、易扩展、数据安全、性能提升

2. 技术：集群地址（虚拟 IP）、网络通信（监控消息）

3. 功能：负载均衡、读写分离、故障转移

5、Memcached 是什么？

Memcached 是一个高性能的分布式内存对象缓存系统，用于动态 Web 应用以减轻数据库负载。它通过在内存中缓存数据和对象来减少读取数据库的次数，从而提高动态、数据库驱动网站的速度。Memcached 基于一个存储键/值对的 [hashmap](#)。其守护进程 (daemon) 是用 C 写的，但是客户端可以用任何语言来编写，并通过 memcached 协议与守护进程通信。

缺点：1：memcached 缺乏认证以及安全管制，这代表应该将 memcached 服务器放置在防火墙后。

2：memcached 的 API 使用 32 位元的[循环冗余校验](#) (CRC-32) 计算键值后，将资料分散在不同的机器上。当表格满了以后，接下来新增的资料会以 LRU 机制替换掉。由于 memcached 通常只是当作快取系统使用，所以使用 memcached 的应用程式在写回较慢的系统时（像是后端的数据库）需要额外的程式码更新 memcached 内的资料。

5、MQ

1. RabbitMQ：

RabbitMQ 是实现 AMQP（高级消息队列协议）的消息中间件的一种，最初起源于金融系统，用于在分布式系统中存储转发消息，在易用性、扩展性、高可用性等方面表现不俗。消息中间件主要用于组件之间的解耦，消息的发送者无需知道消息使用者的存在，反之亦然。

RabbitMQ 本身支持很多的协议：AMQP, XMPP, SMTP, STOMP，也正是如此，使的它变的非常重量级，更适合于企业级的开发。同时实现了一个经纪人(Broker)构架，这意味着消息在发送给客户端时先在中心队列 排队。对路由(Routing)，负载均衡(Load balance)或者数据持久化都有很好的支持。

RabbitMQ 的优点（适用范围）

- 1、基于 erlang 语言开发具有高可用高并发的优点，适合集群服务器。
- 2、健壮、稳定、易用、跨平台、支持多种语言、文档齐全。
- 3、有消息确认机制和持久化机制，可靠性高。
- 4、开源 其他 MQ 的优势：1.2.1 Apache ActiveMQ 曝光率最高，但是可能会丢消息。 5、ZeroMQ 延迟很低、支持灵活拓扑，但是不支持消息持久化和崩溃恢复。

2. ActiveMQ

1. 什么是 ActiveMQ ?

ActiveMQ 是 Apache 出品，最流行的，能力强劲的开源消息总线。ActiveMQ 是一个完全支持 JMS1.1 和 J2EE 1.4 规范的 JMS Provider 实现,尽管 JMS 规范出台已经是很久的事情了,但是 JMS 在当今的 J2EE 应用中间仍然扮演着特殊的地位。

主要特点：

1. 多种语言和协议编写客户端。语言: Java, C, C++, C#, Ruby, Perl, Python, PHP。应用协议: OpenWire, Stomp, REST, WS Notification, XMPP, AMQP
2. 完全支持 JMS1.1 和 J2EE 1.4 规范 (持久化, XA 消息, 事务)
3. 对 Spring 的支持, ActiveMQ 可以很容易内嵌到使用 Spring 的系统里面去, 而且也支持 Spring 2.0 的特性
4. 通过了常见 J2EE 服务器(如 Geronimo, JBoss 4, GlassFish, WebLogic)的测试, 其中通过 JCA 1.5 resource adaptors 的配置, 可以让 ActiveMQ 可以自动的部署到任何兼容 J2EE 1.4 商业服务器上
5. 支持多种传送协议: in-VM, TCP, SSL, NIO, UDP, JGroups, JXTA
6. 支持通过 JDBC 和 journal 提供高速的消息持久化
7. 从设计上保证了高性能的集群, 客户端-服务器, 点对点
8. 支持 Ajax
9. 支持与 Axis 的整合
10. 可以很容易得调用内嵌 JMS provider, 进行测试

2. ActiveMQ 的消息形式

对于消息的传递有两种类型：

一种是点对点的，即一个生产者和一个消费者一一对应；

另一种是发布/订阅模式，即一个生产者产生消息并进行发送后，可以由多个消费者进行接收。

点到点模式：如果消息发送不成功此 消息默认会保存到 activemq 服务端知道有消费者将其消费，所以此时消息是不会丢失的。

发布订阅模式的通信方式：默认情况下只通知一次，如果接收不到此消息就没有了。这种场景只适用于对消息送达率要求不高的情况。如果要求消息必须送达不可以丢失的话，需要

配置持久订阅。每个订阅端定义一个 id，在订阅是向 activemq 注册。发布消息和接收消息时需要配置发送模式为持久化。此时 如果客户端接收不到消息，消息会持久化到服务端，直到客户端正常接收后为止。

JMS 定义了五种不同的消息正文格式，以及调用的消息类型，允许你发送并接收以一些不同形式的数据，提供现有消息格式的一些级别的兼容性。

- StreamMessage -- Java 原始值的数据流
- MapMessage--一套名称-值对
- TextMessage--一个字符串对象
- ObjectMessage--一个序列化的 Java 对象
- BytesMessage--一个字节的数据流

3. ActiveMQ 的作用、原理？（生产者。消费者。 p2p、订阅实现流程）

Activemq 的作用就是系统之间进行通信。当然可以使用其他方式进行系统间通信，如果使用 Activemq 的话可以对系统之间的调用进行解耦，实现系统间的异步通信。原理就是生产者生产消息，把消息发送给 activemq。Activemq 接收到消息，然后查看有多少个消费者，然后把消息转发给消费者，此过程中生产者 无需参与。消费者接收到消息后做相应的处理和生产者没有任何关系。

4. ActiveMQ 在项目中应用场景？

Activemq 在项目中主要是完成系统之间通信，并且将系统之间的调用进行解耦。例如在添加、修改商品信息后，需要将商品信息同步到索引库、同步缓存中的数据以及生成静态页面一系列操作。在此场景下就可 以使用 activemq。一旦后台对商品信息进行修改后，就向 activemq 发送一条消息，然后通过 activemq 将 消息发送给消息的消费端，消费端接收到消息可以进行相应的业务处理。

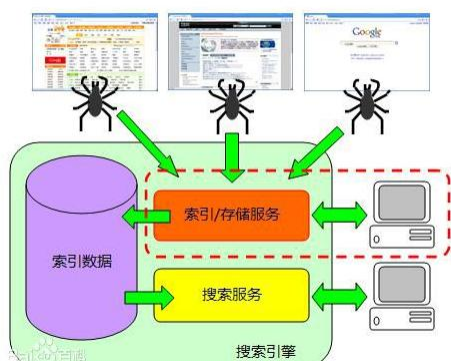
5. ActiveMQ 如果数据提交不成功怎么办？

Activemq 有两种通信方式，点到点形式和发布订阅模式。

6、Solr

1. Solr 是什么

Solr 是一个独立的[企业级搜索](#)应用服务器, 它对外提供类似于 Web-service 的 API 接口。用户可以通过 http 请求, 向搜索引擎服务器提交一定格式的 XML 文件, 生成索引; 也可以通过 Http Get 操作提出查找请求, 并得到 XML 格式的返回结果。

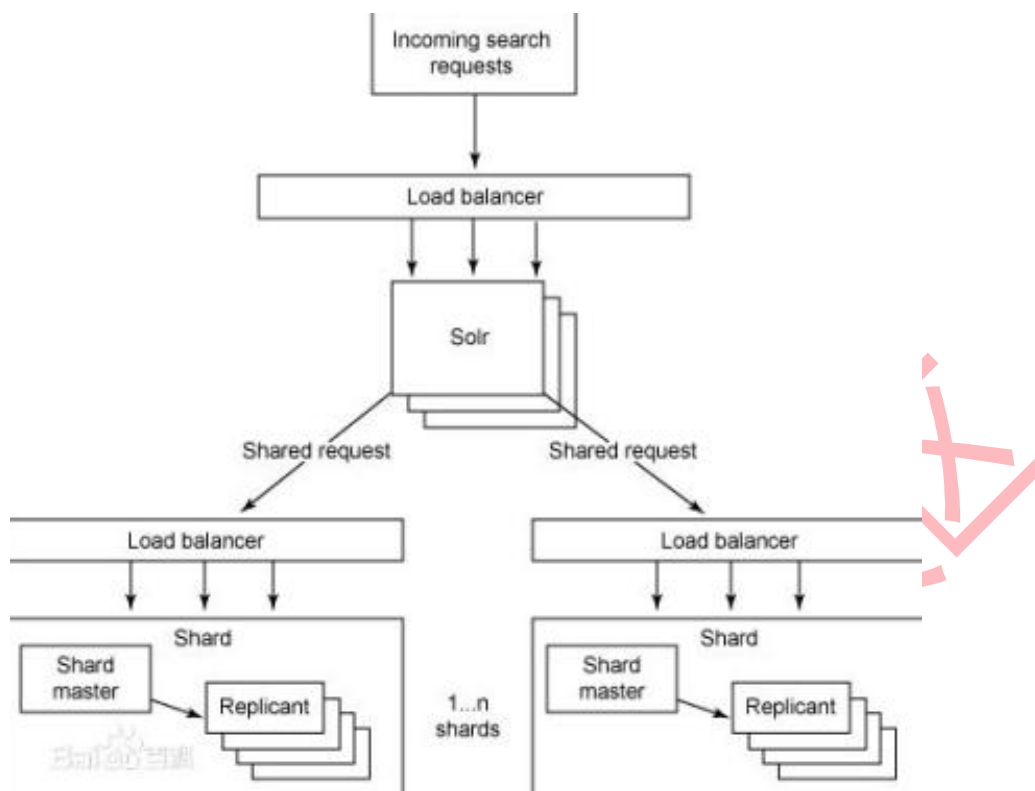


特点：

Solr 是一个高性能, 采用 Java5 开发, 基于 Lucene 的全文搜索服务器。同时对其进行进行了扩展, 提供了比 Lucene 更为丰富的查询语言, 同时实现了可配置、可扩展并对查询性能进行了优化, 并且提供了一个完善的功能管理界面, 是一款非常优秀的[全文搜索引擎](#)。

工作方式：

文档通过 Http 利用 XML 加到一个搜索集合中。查询该集合也是通过 http 收到一个 XML/JSON 响应来实现。它的主要特性包括：高效、灵活的缓存功能, 垂直搜索功能, 高亮显示搜索结果, 通过索引复制来提高可用性, 提供一套强大 Data Schema 来定义字段, 类型和设置文本分析, 提供基于 Web 的管理界面等。



2. Solr 里面的 IK 分词器的原理

IK 分词器的分词原理本质上是词典分词。现在内存中初始化一个词典，然后在分词过程中挨个读取字符，和字典中的字符相匹配，把文档中的所有的词语拆分出来的过程。

3. Solr 怎么设置搜索结果排名靠前（得分）？

可以设置文档中域的 boost 值，boost 值越高，计算出来的相关度得分就越高，排名也就越靠前。此方法可以把热点商品或者推广商品的排名提高。

7、Zookeeper（淘淘商城）

1. Zookeeper 是什么？

ZooKeeper 是一个分布式的，开放源码的分布式应用程序协调服务，是 Google 的 Chubby 一个开源的实现，是 Hadoop 和 Hbase 的重要组成部分。它是一个为分布式应用提供一致性服务的软件，提供的功能包括：配置维护、域名服务、分布式同步、组服务等。

ZooKeeper 的目标就是封装好复杂易出错的关键服务，将简单易用的接口和性能高效、功能稳定的系统提供给用户。

ZooKeeper 包含一个简单的原语集，提供 Java 和 C 的接口。

ZooKeeper 代码版本中，提供了分布式独享锁、选举、队列的接口，代码在 zookeeper-3.4.3\src\recipes。其中分布锁和队列有 Java 和 C 两个版本，选举只有 Java 版本。

2. 原理：

ZooKeeper 是以 Fast Paxos 算法为基础的，Paxos 算法存在活锁的问题，即当有多个 proposer 交错提交时，有可能互相排斥导致没有一个 proposer 能提交成功，而 Fast Paxos 作了一些优化，通过选举产生一个 leader (领导者)，只有 leader 才能提交 proposer，具体算法可见 Fast Paxos。因此，要想看懂 ZooKeeper 首先得对 Fast Paxos 有所了解。[3]

ZooKeeper 的基本运转流程：

- 1、选举 Leader。
- 2、同步数据。
- 3、选举 Leader 过程中算法有很多，但要达到的选举标准是一致的。
- 4、Leader 要具有最高的执行 ID，类似 root 权限。
- 5、集群中大多数的机器得到响应并 follow 选出的 Leader。[3]

8、Freemarker

1. 什么是 freemarker ?

FreeMarker 是一个用 Java 语言编写的模板引擎，它基于模板来生成文本输出。FreeMarker 与 Web 容器无关，即在 Web 运行时，它并不知道 Servlet 或 HTTP。它不仅可以用于表现层的实现技术，而且还可以用于生成 XML，JSP 或 Java 等。

目前企业中:主要用 Freemarker 做静态页面或是页面展示

2. Freemarker 的使用方法

把 freemarker 的 jar 包添加到工程中。

Maven 工程添加依赖

```
<dependency>

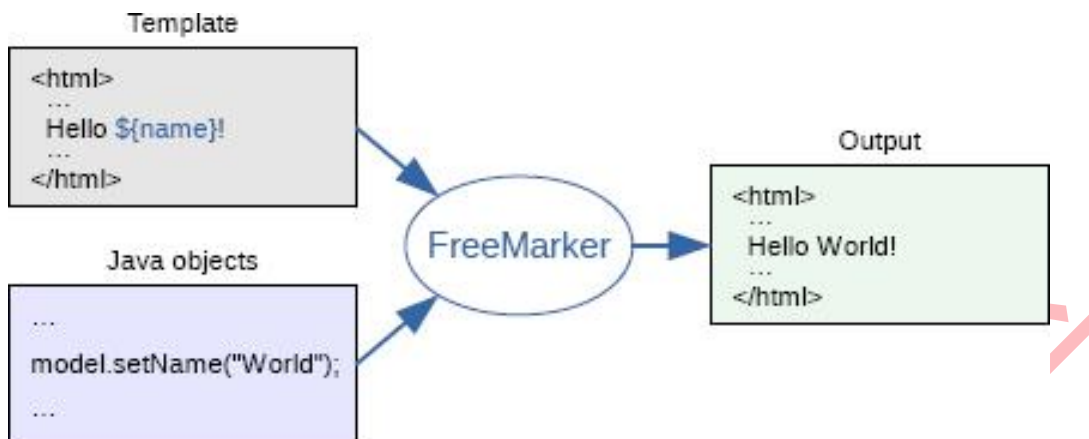
    <groupId>org.freemarker</groupId>

    <artifactId>freemarker</artifactId>

    <version>2.3.23</version>
```

```
</dependency>
```

原理：



使用步骤：

第一步：创建一个 Configuration 对象，直接 new 一个对象。构造方法的参数就是 freemarker 对于的版本号。

第二步：设置模板文件所在的路径。

第三步：设置模板文件使用的字符集。一般就是 utf-8.

第四步：加载一个模板，创建一个模板对象。

第五步：创建一个模板使用的数据集，可以是 pojo 也可以是 map。一般是 Map。

第六步：创建一个 Writer 对象，一般创建一 FileWriter 对象，指定生成的文件名。

第七步：调用模板对象的 process 方法输出文件。

第八步：关闭流。

3. 网页的静态化方案

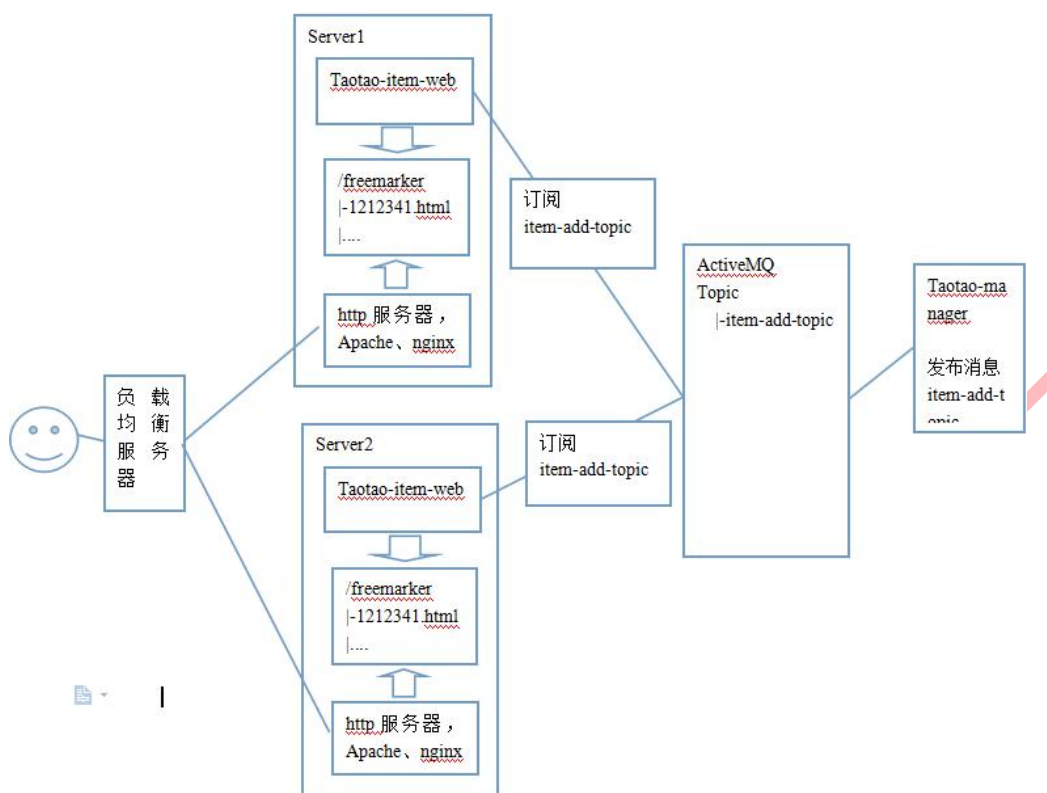
输出文件的名称：商品 id+“.html”

输出文件的路径：工程外部的任意目录。

网页访问：使用 nginx 访问网页。在此方案下 tomcat 只有一个作用就是生成静态页面。

工程部署：可以把 taotao-item-web 部署到多个服务器上。

生成静态页面的时机：商品添加后，生成静态页面。可以使用 Activemq，订阅 topic（商品添加）



9、Nginx

1. Nginx 是什么？

Nginx 是一款高性能的 http 服务器/反向代理服务器及电子邮件（IMAP/POP3）代理服务器。由俄罗斯的程序设计师 Igor Sysoev 所开发，官方测试 nginx 能够支撑 5 万并发链接，并且 cpu、内存等资源消耗却非常低，运行非常稳定。

2. 应用场景

- 1、**http 服务器**。Nginx 是一个 http 服务可以独立提供 http 服务。可以做网页静态服务器。
- 2、**虚拟主机**。可以实现在一台服务器虚拟出多个网站。例如个人网站使用的虚拟主机。
- 3、**反向代理，负载均衡**。当网站的访问量达到一定程度后，单台服务器不能满足用户的需求时，需要用多台服务器集群可以使用 nginx 做反向代理。并且多台服务器可以平均分担负载，不会因为某台服务器负载高宕机而某台服务器闲置的情况。

3. Nginx 与 Apache 的对比

1.1、nginx 相对于 apache 的优点：

轻量级，同样起 web 服务，比 apache 占用更少的内存及资源抗并发，nginx 处理请求是异步非阻塞的，而 apache 则是阻塞型的，在高并发下 nginx 能保持低资源低消耗高性能高度模块化的设计，编写模块相对简单社区活跃，各种高性能模块出品迅速啊

apache 相对于 nginx 的优点：

rewrite，比 nginx 的 rewrite 强大，动态页面，模块超多，基本想到的都可以找到，少 bug，nginx 的 bug 相对较多，超稳定

存在就是理由，一般来说，需要性能的 web 服务，用 nginx。如果不需要性能只求稳定，那就 apache 吧。后者的各种功能模块实现得比前者，例如 ssl 的模块就比前者好，可配置项多。这里要注意一点，epoll(freebsd 上是 kqueue)网络 IO 模型是 nginx 处理性能高的根本理由，但并不是所有的情况下都是 epoll 大获全胜的，如果本身提供静态服务的就只有寥寥几个文件，apache 的 select 模型或许比 epoll 更高性能。

1.2、作为 Web 服务器：相比 Apache，Nginx 使用更少的资源，支持更多的并发连接，

体现更高的效率，这点使 Nginx 尤其受到虚拟主机提供商的欢迎。在高连接并发的情况下，Nginx 是 Apache 服务器不错的替代品：Nginx 在美国是做虚拟主机生意的老板们经常选择的软件平台之一。能够支持高达 50,000 个并发连接数的响应，感谢 Nginx 为我们选择了 epoll and kqueue 作为开发模型。

Nginx 作为负载均衡服务器：Nginx 既可以在内部直接支持 Rails 和 PHP 程序对外进行服务，也可以支持作为 HTTP 代理 服务器对外进行服务。Nginx 采用 C 进行编写，不论是系统资源开销还是 CPU 使用效率都比 Perlbal 要好很多。

作为邮件代理服务器：Nginx 同时也是一个非常优秀的邮件代理服务器（最早开发这个产品的目的之一也是作为邮件代理服务器），Last.fm 描述了成功并且美妙的使用经验。

Nginx 是一个安装非常的简单，配置文件非常简洁（还能够支持 perl 语法），Bugs 非常的少：Nginx 启动特别容易，并且几乎可以做到 7*24 不间断运行，即使运行数个月也不需要重新启动。你还能够不间断服务的情况下进行软件版本的升级。

1.3 nginx 处理静态文件好，耗费内存少。但无疑 apache 仍然是目前的主流，有很多丰富的特性。所以还需要搭配着来。当然如果能确定 nginx 就适合需求，那么使用 nginx 会是更经济的方式。

apache 有先天不支持多核心处理负载鸡肋的缺点，建议使用 nginx 做前端，后端用 apache。大型网站建议用 nginx 自带的集群功能

1.4 nginx 处理动态请求是鸡肋，一般动态请求要 apache 去做，nginx 只适合静态和反向。

4. Nginx 反向代理为什么可以提高网站性能？

对于后端是动态服务来说，比如 Java 和 PHP。这类服务器（如 JBoss 和 PHP-FPM）的 IO 处理能力往往不高。Nginx 有个好处是它会把 **Request** 在读取完整之前 buffer 住，这样交给后端的就是一个完整的 HTTP 请求，从而提高后端的效率，而不是断断续续的传递（互联网上连接速度一般比较慢）。同样，Nginx 也可以把 **response** 给 buffer 住，同样也是减轻后端的压力。

10、HttpClient

1. HttpClient 是什么？

HttpClient 不是一个浏览器。它是一个客户端的 HTTP 通信实现库。HttpClient 的目标是发送和接收 HTTP 报文。HttpClient 不会去缓存内容，执行嵌入在 HTML 页面中的 javascript 代码，猜测内容类型，重新格式化 请求/重定向 URI，或者其它和 HTTP 运输无关的功能。它主要就是支持 HTTP 传输协议的。

2. HttpClient 的使用

我们知道，HTTP 协议的连接方法有 GET、POST、PUT 和 HEAD 方式，在创建 Method 实例的时候可以更具具体的方法来创建。HttpClient 的使用一般分如下几步：

- 1、创建 HttpClient 实例。
- 2、创建具体连接方法的实例。如 POST 方法创建 PostMethod 的实例，在实例化时从构造函数中传入待连接的 URL 地址。
- 3、对 post 的发送内容等信息进行配置
- 4、执行 HttpClient 的 execute 方法
- 5、如果返回的状态码正常，表明连接成功，可以读取 response 的内容

11、Jsonp

jsonp 到底是什么？

JSONP 的原理非常简单，**为了克服跨域问题，利用没有跨域限制的 script 标签加载预设的 callback 将内容传递给 js。**一般来说我们约定通过一个参数来告诉服务器 JSONP 返回时应该调用的回调函数名，然后拼接出对应的 js。

12、Quartz

1. Quartz 是什么？

Quartz 作为一个优秀的开源调度框架，

Quartz 具有以下特点：

强大的调度功能，

支持立即调度、定时调度、周期调度、并发调度；

灵活的应用方式，支持 job 间通过 listener 实现依赖调度，可以方便的进行调度组合

2. 用 Quartz 做定时任务调度

需求是这样的，以整点时间戳为文件名，每隔一小时创建一个文件，在这一小时内不断的写文件，达到下一小时关闭当前文件句柄和流，并以当前整点小时创建新文件！

写一个单例服务类，服务类两个成员变量，一个是文件句柄，一个是流，可 get 可 set。把它作为 spring 的一个 bean，在 quartz 和你的读写线程都注入这个 bean。quartz 里面用 set 来改句柄(百度上有很好的 解释)和流，读写线程用 get 来读

3. 如何监控 Quartz 的 job 执行状态：运行中，暂停中，等待中？

目前我能想到的解决办法是通过往表(新建一个操作日志表)里插入日志的形式：

运行中：通过 JobListener 监听器来实现运行时更改表信息

暂停中：调用 scheduler.pauseTrigger() 方法时，更改表中 job 信息

等待中：新添加的 job 默认给其等待中的状态，也是更改表中的 job 信息但是上面这种形式的麻烦之处是得频繁的往表里插入数据。

13、Mycat 中间件（新巴巴项目）

《MyCat》是代替昂贵的 oracle 的 MySQL 集群中间件。

1. 为什么需要 MyCat？

虽然云计算时代，传统数据库存在着先天性的弊端，但是 NoSQL 数据库又无法将其替代。如果传统数据易于扩展，可切分，就可以避免单机（单库）的性能缺陷。

MyCat 的目标就是：低成本地将现有的单机数据库和应用平滑迁移到“云”端，解决数据存储和业务规模迅速增长情况下的数据瓶颈问题。2014 年 MyCat 首次在上海的《中华架构师》大会上对外宣讲引发围观，更多的人参与进来，随后越来越多的项目采用了 MyCat。

2. MyCat 是什么？

从定义和分类来看，它是一个开源的分布式数据库系统，是一个实现了 MySQL 协议的服务器，前端用户可以把它看作是一个数据库代理，用 MySQL 客户端工具和命令行访问，而其后端可以用 MySQL 原生协议与多个 MySQL 服务器通信，也可以用 JDBC 协议与大多数主流数据库服务器通信，其核心功能是分表分库，即将一个大表水平分割为 N 个小表，存储在后端 MySQL 服务器里或者其他数据库里。

MyCat 发展到目前的版本，已经不是一个单纯的 MySQL 代理了，它的后端可以支持 MySQL、SQL Server、Oracle、DB2、PostgreSQL 等主流数据库，也支持 MongoDB 这种新型 NoSQL 方式的存储，未来还会支持更多类型的存储。而在最终用户看来，无论是那种存储方式，在 MyCat 里，都是一个传统的数据库表，支持标准的 SQL 语句进行数据的操作，这样一来，对前端业务系统来说，可以大幅降低开发难度，提升开发速度

黑马程序员·顺义校区