Java 工程师（程序员）面题
Struts,Spring,Hibernate 三大框架

1.Hibernate 工作原理及为什么要用？
原理： 1.读取并解析配置文件 2.读取并解析映射信息，创建 SessionFactory 3.打开 Session 4.创建事务 Transation 5.持久化操作 6.提交事务 7.关闭 Session 8.关闭 SesstionFactory
为什么要用： 1. 对 JDBC 访问数据库的代码做了封装，大大简化了数据访问层繁琐的重复性代码。 2. Hibernate 是一个基于 JDBC 的主流持久化框架，是一个优秀的 ORM 实现。他很大程度的简化 DAO 层的编码工作 3. hibernate 使用 Java 反射机制，而不是字节码增强程序来实现透明性。 4. hibernate 的性能非常好，因为它是个轻量级框架。映射的灵活性很出色。它支持各种关系数据库，从一对一到多对多的各种复杂关系。

2．Hibernate 是如何延迟加载?
1. Hibernate2 延迟加载实现：a)实体对象 b)集合（Collection）
2. Hibernate3 提供了属性的延迟加载功能 当 Hibernate 在查询数据的时候，数据并没有存在与内存中，当程序真正对数据的操作时，对象才存在与内存中，就实现了延迟加载，他节省了服务器的内存开销，从而提高了服务器的性能。

3．Hibernate 中怎样实现类之间的关系?(如：一对多、多对多的关系)
类与类之间的关系主要体现在表与表之间的关系进行操作，它们都是对对象进行操作，我们程序中把所有的表与类都映射在一起，它们通过配置文件中的 many-to-one、one-to-many、many-to-many

4.Struts1 流程：
1、客户端浏览器发出 HTTP 请求。2、根据 web.xml 配置，该请求被 ActionServlet 接收。3、根据 struts-config.xml 配置， ActionServlet 先将请求中的参数填充到 ActionForm 中，然后 ActionServlet 再将请求发送到 Action 进行处理。4、是否验证，需要验证则调用 ActionForm 的 validate 方法，验证失败则跳转到 input，成功则继续。5、Action 从 ActionForm 获得数据，调用 javabean 中的业务方法处理数据。6、Action 返回 ActionForward 对象，跳转到相应 JSP 页面或 Action。7、返回 HTTP 响应到客户端浏览器。

MVC 设计模式：modal：“模型” 也称业务逻辑，是正真完成任务的代码,相当与 JavaBeanview：视图，其实就是显示界面，相当于 JSPcontroller：控制器，他控制模型和视图的交互过程，相当于 servletstruts1 是基于 MVC 设计模式 hibernate 是基于 ORM 对象关系映射

5.struts 是什么？
struts1 是基于 JSP 和 servlet 的一个开源的 Web 应用框架，使用的是 MVC 的设计模式 struts2 是基于 webwork 技术的框架，是 sun 和 webwork 公司联手开发的一个功能非常齐全的框架，struts2 和 struts1 没有任何关系，是一个全新的框架

6.spring 是什么？

spring 是一个集成了许多第三方框架的大杂烩，其核心技术是 IOC（控制反转，也称依赖注入）和 AOP（面向切面编程）

7.hibernate 是什么？
hibernate 是基于 ORM 对象关系映射（完成对象数据到关系数据映射的机制）实现的,做数据持久化的工具

8.JSF 是什么？
JavaServer Face 是基于组件的 web 开发框架，跟 sturts 差不多的框架

9.数据库里面的索引和约束是什么?
索引是为了提高数据的检索速度，索引是建立在数据表上，根据一个或多个字段建立的约束是为了保持数据的完整性，约束有非空约束，主键约束，外键约束等等。

10.spring 是什么
这个问题，往往可以通过我们为什么要使用 spring 这个问题来切入：AOP 让开发人员可以创建非行为性的关注点，称为横切关注点，并将它们插入到应用程序代码中。使用 AOP 后，公共服务 （比 如日志、持久性、事务等）就可以分解成方面并应用到域对象上，同时不会增加域对象的对象模型的复杂性。 IOC 允许创建一个可以构造对象的应用环境，然后向这些对象传递它们的协作对象。正如单词 倒置 所表明的，IOC 就像反 过来的 JNDI。没有使用一堆抽象工厂、服务定位器、单元素（singleton）和直接构造（straight construction），每一个对象都是用其协作对象构造的。因此是由容器管理协作对象（collaborator）。 Spring 即使一个 AOP 框架，也是一 IOC 容器。 Spring 最好的地方是它有助于您替换对象。有了 Spring，只要用 JavaBean 属性和配置文件加入依赖性（协作对象）。然后可以很容易地在需要时替换具有类似接口的协作对象。

11.用自己的话简要阐述 struts2 的执行流程。
Struts 2 框架本身大致可以分为 3 个部分：核心控制器 FilterDispatcher、业务控制器 Action 和用户实现的企业业务逻辑组件。核心控制器 FilterDispatcher 是 Struts 2 框架的基础，包含了框架内部的控制流程和处理机制。业务控制器 Action 和业务逻辑组件是需要用户来自己实现的。用户在开发 Action 和业务逻辑组件的同时，还需要编写相关的配置文件，供核心控制器 FilterDispatcher 来使用。

Struts 2 的工作流程相对于 Struts 1 要简单，与 WebWork 框架基本相同，所以说 Struts 2 是 WebWork 的升级版本。基本简要流程如下：1、客户端浏览器发出 HTTP 请求。2、根据 web.xml 配置，该请求被 FilterDispatcher 接收。3、根据 struts.xml 配置，找到需要调用的 Action 类和方法， 并通过 IoC 方式，将值注入给 Aciton。4、Action 调用业务逻辑组件处理业务逻辑，这一步包含表单验证。5、Action 执行完毕，根据 struts.xml 中的配置找到对应的返回结果 result，并跳转到相应页面。6、返回 HTTP 响应到客户端浏览器。
1 Action 是不是线程安全的？如果不是 有什么方式可以保证 Action 的线程安全？如果是，说明原因
不是
声明局部变量，或者扩展 RequestProcessor，让每次都创建一个 Action，或者在 spring 中用 scope="prototype"来管理

2.MVC，分析一下 struts 是如何实现 MVC 的
m：JavaBean 或结合 EJB 组件或者 pojo 构成
c：Action 来实现
v：一组 JSP 文件及其标签构成。

3.struts 中的几个关键对象的作用(说说几个关键对象的作用)
Action：控制器类，ActionForm：表单对象，DynaValidatorForm：动态 form，ActonMapping：配置文件中 action 节点的信息.....

4.说说 AOP 和 IOC 的概念以及在 spring 中是如何应用的
AOP:面向方面编程，ioc：依赖注入；声明式事务和编程式事务积极一些通用部分

5.Hibernate 有哪几种查询数据的方式
hql 查询，sql 查询，条件查询

6.load()和 get()的区别
hibernate 对于 load 方法认为该数据在数据库中一定存在，可以放心的使用代理来延迟加载，load 默认支持延迟加载，在用到对象中的其他属性数 据时才查询数据库，但是万一数据库中不存在该记录，只能抛异常 ObjectNotFoundEcception；所说的 load 方法抛异常是指在使用该对 象的数据时，数据库中不存在该数据时抛异常，而不是在创建这个对象时。由于 session 中的缓存对于 hibernate 来说是个相当廉价的资源，所以在 load 时会先查一下 session 缓存看看该 id 对应的对象是否存在，不存在则创建代理（load 时候之查询一级缓存，不存在则创建代理）。get() 现在一级缓存找，没有就去二级缓存找，没有就去数据库找，没有就返回 null ；而对于 get 方法，hibernate 一定要获取到真实的数据，否则返回 null。

7.谈谈 hibernate 的延迟加载和 openSessionInView
延迟加载要在 session 范围内，用到的时候再加载；opensessioninview 是在 web 层写了一个 filter 来打开和关闭 session，这样就表示在一次 request 过程中 session 一直开着，保证了延迟
加载在 session 中的这个前提。

8.spring 的事务有几种方式？谈谈 spring 事务的隔离级别和传播行为。
声明事务和编程事务
隔离级别：
- DEFAULT 使用数据库默认的隔离级别
- READ_UNCOMMITTED 会出现脏读，不可重复读和幻影读问题
- READ_COMMITTED 会出现重复读和幻影读
- REPEATABLE_READ 会出现幻影读
- SERIALIZABLE 最安全，但是代价最大，性能影响极其严重
和传播行：
- REQUIRED 存在事务就融入该事务，不存在就创建事务
- SUPPORTS 存在事务就融入事务，不存在则不创建事务
- MANDATORY 存在事务则融入该事务，不存在，抛异常

- REQUIRES_NEW 总是创建新事务
- NOT_SUPPORTED 存在事务则挂起，一直执行非事务操作
- NEVER 总是执行非事务，如果当前存在事务则抛异常
- NESTED 嵌入式事务

9.Hibernate 中的 update()和 saveOrUpdate()的区别.
摘自 hibernate 说明文档：
saveOrUpdate()做下面的事:
如果对象已经在本 session 中持久化了，不做任何事
如果另一个与本 session 关联的对象拥有相同的持久化标识(identifier)，抛出一个异常
如果对象没有持久化标识(identifier)属性，对其调用 save()
如果对象的持久标识(identifier)表明其是一个新实例化的对象，对其调用 save()
如果对象是附带版本信息的（通过 <version>或 <timestamp>） 并且版本属性的值表明其是一个新实例化的对象，save()它。 否则 update() 这个对象
10.Spring 对多种 ORM 框架提供了很好的支持，简单描述在 Spring 中使用 Hibernate 的方法，并结合事务管理。
getHiberanteTemplate 里面提供了 save，update，delete，find 等方法。
简单说一个：如果配置了声明式事务，当执行 getHibernateTemplate 的各种方法的时候，事务会
自动被加载
如果没有配置事务，那么以上操作不会真正的被同步到数据库，除非配置了 hibernate 的 autocommit=true
8.spring 的事务有几种方式？谈谈 spring 事务的隔离级别和传播行为。
spring 事务分两种形式，声明式事务和编程式事务，spring 提供了一个事务的接口 PaltformTractionManager 接口，针对不同的事务，spring 进行了不同的实现,对 hibernate 事务的实现 HIbernateTractionManager,对 JDBC 的 JdbcTractionManager，DataSourceTractionManager 以及 JdoTractionManager。接口 platformTractionManager 提供了三
个方法，获取事务，提交和回滚的方法。
****
分享面试题二】Spring,hibernate,struts 的面试笔试题（含答案）
（声明：这里不是为其他商业利益，是为学习讨论使用）
【郑重声明】：单纯接分将被删帖，希望大家有自己的感触
Hibernate 工作原理及为什么要用？
原理：
1.读取并解析配置文件
2.读取并解析映射信息，创建 SessionFactory
3.打开 Sesssion
4.创建事务 Transation
5.持久化操作
6.提交事务
7.关闭 Session
8.关闭 SesstionFactory
为什么要用：

1. 对 JDBC 访问数据库的代码做了封装，大大简化了数据访问层繁琐的重复性代码。
2. Hibernate 是一个基于 JDBC 的主流持久化框架，是一个优秀的 ORM 实现。他很大程度的简化 DAO 层的编码工作
3. hibernate 使用 Java 反射机制，而不是字节码增强程序来实现透明性。
4. hibernate 的性能非常好，因为它是个轻量级框架。映射的灵活性很出色。它支持各种关系数据库，从一对一到多对多的各种复杂关系。

2． Hibernate 是如何延迟加载?

1. Hibernate2 延迟加载实现：a)实体对象 b)集合（Collection）

2. Hibernate3 提供了属性的延迟加载功能

当 Hibernate 在查询数据的时候，数据并没有存在与内存中，当程序真正对数据的操作时，对象才存在与内存中，就实现了延迟加载，他节省了服务器的内存开销，从而提高了服务器的性能。

3．Hibernate 中怎样实现类之间的关系?(如：一对多、多对多的关系)

类与类之间的关系主要体现在表与表之间的关系进行操作，它们都市对对象进行操作，我们程序中把所有的表与类都映射在一起，它们通过配置文件中的 many-to-one、one-to-many、many-to-many、

4． 说下 Hibernate 的缓存机制

1. 内部缓存存在 Hibernate 中又叫一级缓存，属于应用事物级缓存

2. 二级缓存：

a) 应用及缓存

b) 分布式缓存

条件：数据不会被第三方修改、数据大小在可接受范围、数据更新频率低、同一数据被系统频繁使用、非 关键数据

c) 第三方缓存的实现

5． Hibernate 的查询方式

Sql、Criteria,object comptosition

Hql：

1、 属性查询

2、 参数查询、命名参数查询

3、 关联查询

4、 分页查询

5、 统计函数

6． 如何优化 Hibernate？

1.使用双向一对多关联，不使用单向一对多

2.灵活使用单向一对多关联

3.不用一对一，用多对一取代

4.配置对象缓存，不使用集合缓存

5.一对多集合使用 Bag,多对多集合使用 Set

6. 继承类使用显式多态

7. 表字段要少，表关联不要怕多，有二级缓存撑腰

7． Struts 工作机制？为什么要使用 Struts？

工作机制：

Struts 的工作流程:

在 web 应用启动时就会加载初始化 ActionServlet,ActionServlet 从
struts-config.xml 文件中读取配置信息,把它们存放到各种配置对象
当 ActionServlet 接收到一个客户请求时,将执行如下流程.

-(1)检索和用户请求匹配的 ActionMapping 实例,如果不存在,就返回请求路径无效信息;

-(2)如果 ActionForm 实例不存在,就创建一个 ActionForm 对象,把客户提交的表单数据保存到 ActionForm 对象中;

-(3)根据配置信息决定是否需要表单验证.如果需要验证,就调用 ActionForm 的 validate()方法;

-(4)如果 ActionForm 的 validate()方法返回 null 或返回一个不包含 ActionMessage 的 ActuibErrors 对象, 就表示表单验证成功;

-(5)ActionServlet 根据 ActionMapping 所包含的映射信息决定将请求转发给哪个 Action,如果相应 的 Action 实例不存在,就先创建这个实例,然后调用 Action 的 execute()方法;

-(6)Action 的 execute()方法返回一个 ActionForward 对象,ActionServlet 在把客户请求转发给 ActionForward 对象指向的 JSP 组件;

-(7)ActionForward 对象指向 JSP 组件生成动态网页,返回给客户;

为什么要用：

JSP、Servlet、JavaBean 技术的出现给我们构建强大的企业应用系统提供了可能。但用这些技术构建的系统非常的繁乱，所以在此之上，我们需要一个规则、一个把这些技术组织起来的规则，这就是框架，Struts 便应运而生。

基于 Struts 开发的应用由 3 类组件构成：控制器组件、模型组件、视图组件

8． Struts 的 validate 框架是如何验证的？

在 struts 配置文件中配置具体的错误提示，再在 FormBean 中的 validate()方法具体调用。

9． 说下 Struts 的设计模式

MVC 模式: web 应用程序启动时就会加载并初始化 ActionServler。用户提交表单时，一个配置好的 ActionForm 对象被创建，并被填入表单相应的数 据，ActionServler 根据 Struts-config.xml 文件配置好的设置决定是否需要表单验证，如果需要就调用 ActionForm 的 Validate（）验证后选择将请求发送到哪个 Action，如果 Action 不存在，ActionServlet 会先创建这个对象，然后调用 Action 的 execute（）方法。Execute（）从 ActionForm 对象中获取数据，完成业务逻辑，返回一个 ActionForward 对 象，ActionServlet 再把客户请求转发给 ActionForward 对象指定的 jsp 组件，ActionForward 对象指定的 jsp 生成动 态的网页，返回给客户。

10． spring 工作机制及为什么要用?

1.spring mvc 请所有的请求都提交给 DispatcherServlet,它会委托应用系统的其他模块负责负责对请求进行真正的处理工作。

2.DispatcherServlet 查询一个或多个 HandlerMapping,找到处理请求的 Controller.

3.DispatcherServlet 请请求提交到目标 Controller

4.Controller 进行业务逻辑处理后，会返回一个 ModelAndView

5.Dispathcher 查询一个或多个 ViewResolver 视图解析器,找到 ModelAndView 对象指定的视图对象

6.视图对象负责渲染返回给客户端。

为什么用：

{AOP 让开发人员可以创建非行为性的关注点，称为横切关注点，并将它们插入到应用程序代码中。使用 AOP 后，公共服务 （比 如日志、持久性、事务等）就可以分解成方面并应

用到域对象上，同时不会增加域对象的对象模型的复杂性。

IOC 允许创建一个可以构造对象的应用环境，然后向这些对象传递它们的协作对象。正如单词 倒置 所表明的，IOC 就像反 过来的 JNDI。没有使用一堆抽象工厂、服务定位器、单元素（singleton）和直接构造（straight construction），每一个对象都是用其协作对象构造的。因此是由容器管理协作对象（collaborator）。

Spring 即使一个 AOP 框架，也是一 IOC 容器。 Spring 最好的地方是它有助于您替换对象。有了 Spring，只要用 JavaBean 属性和配置文件加入依赖性（协作对象）。然后可以很容易地在需要时替换具有类似接口的协作对象。}

Struts,Spring,Hibernate 优缺点

Struts 跟 Tomcat、Turbine 等诸 多 Apache 项目一样，是开源软件，这是它的一大优点。使开发者能更深入的了解其内部实现机制。 Struts 开放源码框架的创建是为了使开发者在构建基于 Java Servlet 和 JavaServer Pages（JSP）技术的 Web 应用时更加容易。Struts 框架为开放者提供了一个统一的标准框架，通过使用 Struts 作为基础，开发者能够更专注 于应用程序的商业逻辑。Struts 框架本身是使用 Java Servlet 和 JavaServer Pages 技术的一种 Model-View-Controller（MVC）实现.

具体来讲,

Struts 的优点有:

1. 实现 MVC 模式，结构清晰,使开发者只关注业务逻辑的实现.
2. 有丰富的 tag 可以用 ,Struts 的标记库(Taglib)，如能灵活动用，则能大大提高开发效率。另外，就目前国内的 JSP 开发者而言，除了使用 JSP 自带的常用标记外，很少开发自己的标记，或许 Struts 是一个很好的起点。
3. 页面导航.页面导航将是今后的一个发展方向，事实上，这样做，使系统的脉络更加清晰。通过一个配置文件，即可把握整个系统各部分之间的联系，这对于后期的维护有着莫大的好处。尤其是当另一批开发者接手这个项目时，这种优势体现得更加明显。
4. 提供 Exception 处理机制 .
5. 数据库链接池管理
6. 支持 I18N

缺点:

一、转到展示层时，需要配置 forward，每一次转到展示层，相信大多数都是直接转到 jsp，而涉及到转向， 需要配置 forward，如果有十个展示层的 jsp，需要配置十次 struts，而且还不包括有时候目录、文件变更，需要重新修改 forward，注意， 每次修改配置之后，要求重新部署整个项目，而 tomcate 这样的服务器，还必须重新启动服务器，如果业务变更复杂频繁的系统，这样的操作简单不可想象。 现在就是这样，几十上百个人同时在线使用我们的系统，大家可以想象一下，我的烦恼有多大。

二、 Struts 的 Action 必需是 thread－safe 方式，它仅仅允许一个实例去处理所有的请求。所以 action 用到的所有的资源都必需统一同步，这个就引起了线程安全的问题。

三、 测试不方便. Struts 的每个 Action 都同 Web 层耦合在一起，这样它的测试依赖于 Web 容器，单元测试也很难实现。不过有一个 Junit 的扩展工具 Struts TestCase 可以实现它的单元测试。

四、 类型的转换. Struts 的 FormBean 把所有的数据都作为 String 类型，它可以使用工具 Commons-Beanutils 进行类型转化。但它的转化都是在 Class 级别，而且转化的类型是不可配置的。类型转化时的错误信息返回给用户也是非常困难的。

五、 对 Servlet 的依赖性过强. Struts 处理 Action 时必需要依赖 ServletRequest 和 ServletResponse，所有它摆脱不了 Servlet 容器。

六、前端表达式语言方面.Struts 集成了 JSTL，所以它主要使用 JSTL 的表达式语言来获取数据。可是 JSTL 的表达式语言在 Collection 和索引属性方面处理显得很弱。
七、 对 Action 执行的控制困难. Struts 创建一个 Action，如果想控制它的执行顺序将会非常困难。甚至你要重新去写 Servlet 来实现你的这个功能需求。
八、 对 Action 执行前和后的处理. Struts 处理 Action 的时候是基于 class 的 hierarchies，很难在 action 处理前和后进行操作。
九、 对事件支持不够. 在 struts 中，实际是一个表单 Form 对应一个 Action 类(或 DispatchAction)，换一句话说：在 Struts 中实际是一个表单只能 对应一个事件，struts 这种事件方式称为 application event，application event 和 component event 相比是一种粗粒度的事件。

Struts 重要的表单对象 ActionForm 是一种对象，它代表了一种应用，这个对象中至少包含几个字段， 这些字段是 Jsp 页面表单中的 input 字段，因为一个表单对应一个事件，所以，当我们需要将事件粒度细化到表单中这些字段时，也就是说，一个字段对应一 个事件时，单纯使用 Struts 就不太可能，当然通过结合 JavaScript 也是可以转弯实现的。
2．Hibernate
Hibernate 是一个开放源代码的对象关系映射框架，它对 JDBC 进行了非常轻量级的对象封装，使得 Java 程序员可以随心所欲的使用对象编程思维来操纵数据库。
Hibernate 可以应用在任何使用 JDBC 的场合，既可以在 Java 的客户端程序实用，也可以在 Servlet/JSP 的 Web 应用中使用，最具革命意义的是，Hibernate 可以在应用 EJB 的 J2EE 架构中取代 CMP，完成数据持久化的重任。
大多数开发机构经常采取创建各自独立的数据持久层。一旦底层的数据结构发生改变，那么修改应用的其余部分使之适应这种改变的代价将是十分巨大的。 Hibernate 适时的填补了这一空白，它为 Java 应用提供了一个易用的、高效率的对象关系映射框架。hibernate 是个轻量级的持久性框架，功 能却非常丰富。
优点：
a.Hibernate 使用 Java 反射机制而不是字节码增强程序来实现透明性。
b.Hibernate 的性能非常好，因为它是个轻量级框架。映射的灵活性很出色。
c.它支持各种关系数据库，从一对一到多对多的各种复杂关系。
缺点：
它限制您所使用的对象模型。(例如，一个持久性类不能映射到多个表)其独有的界面和可怜的市场份额也让人不安，尽管如此，Hibernate 还是以其强大的发展动力减轻了这些风险。其他的开源持久性框架也有一些，不过都没有 Hibernate 这样有市场冲击力。
上面回贴情绪有点激动，希望谅解，我不是因为有人批评 Hibernate 而感到不快，而是因为帖子里面的观点实在让我觉得荒谬。不管觉得 Hibernate 好也吧，不好也吧，我唯一觉得遗憾的是，在中文论坛里面找不到一个对 Hibernate 的真正高水平的评价。在 TSS 上有一个关于 Hibernate 的 hot thread，跟了几百贴，其中包括 Hibernate 作者 Gavin 和 LiDO JDO 的 CTO，对于 JDO 和 Hibernate 有过一些激烈的争论，我曾经耐心的看了一遍，仍然没有发现针对 Hibernate 真正有力的攻击，那些所 谓的攻击无非针对 Hibernate 没有一个 GUI 的配置工具，没有商业公司支持，没有标准化等等这些站不住脚的理由。
补充几点我的意见：
一、Hibernate 是 JDBC 的轻量级的对象封装，它是一个独立的对象持久层框架，和 App Server，和 EJB 没有什么必然的联系。Hibernate 可以用在任何 JDBC 可以使用的场合，例如 Java 应用程序的数据库访问代码，DAO 接口 的实现类，甚至可以是 BMP 里面的访问数据库的代

码。从这个意义上来说，Hibernate 和 EB 不是一个范畴的东西，也不存在非此即彼的关系。
二、Hibernate 是一个和 JDBC 密切关联的框架，所以 Hibernate 的兼容性和 JDBC 驱动，和数据库都有一定的关系，但是和使用它的 Java 程序，和 App Server 没有任何关系，也不存在兼容性问题。
三、Hibernate 不能用来直接和 Entity Bean 做对比，只有放在整个 J2EE 项目的框架中才能比较。并且即使是放在软件整体框架中来看，Hibernate 也是做为 JDBC 的替代者出现的，而 不是 Entity Bean 的替代者出现的，让我再列一次我已经列 n 次的框架结构：
传统的架构：
1) Session Bean <-> Entity Bean <-> DB
为了解决性能障碍的替代架构：
2) Session Bean <-> DAO <-> JDBC <-> DB
使用 Hibernate 来提高上面架构的开发效率的架构：
3) Session Bean <-> DAO <-> Hibernate <-> DB
就上面 3 个架构来分析：
1、内存消耗：采用 JDBC 的架构 2 无疑是最省内存的，Hibernate 的架构 3 次之，EB 的架构 1 最差。
2、运行效率：如果 JDBC 的代码写的非常优化，那么 JDBC 架构运行效率最高，但是实际项目中，这一点几乎做不到，这需要程序员非常精通 JDBC，运用 Batch 语句，调整 PreapredStatement 的 Batch Size 和 Fetch Size 等参数，以及在必要的情况下采用结果集 cache 等等。而一般情况下程序员是做不到这一点的。因此 Hibernate 架构表现出最快的运行效率。EB 的架构效率会差的很远。
3、开发效率：在有 JBuilder 的支持下以及简单的项目，EB 架构开发效率最高，JDBC 次之，Hibernate 最差。但是在大的项目，特别是持久层关系映射很复杂的情况下，Hibernate 效率高的惊人，JDBC 次之，而 EB 架构很可能会失败。
4、分布式，安全检查，集群，负载均衡的支持
由于有 SB 做为 Facade，3 个架构没有区别。
四、EB 和 Hibernate 学习难度在哪里？
EB 的难度在哪里？不在复杂的 XML 配置文件上，而在于 EB 运用稍微不慎，就有严重的性能障碍。所以难在你需要学习很多 EJB 设计模式来避开性能问题，需 要学习 App Server 和 EB 的配置来优化 EB 的运行效率。做 EB 的开发工作，程序员的大部分精力都被放到了 EB 的性能问题上了，反而没有更多的精力关注本身就主要 投入精力去考虑的对象持久层的设计上来。
Hibernate 难在哪里？不在 Hibernate 本身的复杂，实际上 Hibernate 非常的简单，难在 Hibernate 太灵活了。
当你用 EB 来实现持久层的时候，你会发现 EB 实在是太笨拙了，笨拙到你根本没有什么可以选择的余地，所以你根本就不用花费精力去设计方案，去平衡方案的好坏，去费脑筋考虑选择哪个方案，因为只有唯一的方案摆在你面前，你只能这么做，没得选择。
Hibernate 相反，它太灵活了，相同的问题，你至少可以设计出十几种方案来解决，所以特别的犯难，究竟用这个，还是用那个呢？这些方案之间到底有什 么区别呢？他们的运行原理有什么不同？运行效率哪个比较好？光是主键生成，就有七八种方案供你选择，你为难不为难？集合属性可以用 Set，可以用 List，还可以用 Bag，到底哪个效率高，你为难不为难？查询可以用 iterator，可以用 list，哪个好，有什么区别？你为难不为难？复合主键你 可以直接在 hbm 里面配置，也可以自定义 CustomerType，哪种比较好些？你为难不为难？对于一个表，你可以选择单一映射一个对象，也可以映射成 父子对象，还可以映射成两个 1:1

的对象，在什么情况下用哪种方案比较好，你为难不为难？
这个列表可以一直开列下去，直到你不想再看下去为止。当你面前摆着无数的眼花缭乱的方案的时候，你会觉得幸福呢？还是悲哀呢？如果你是一个负责的程序员， 那么你一定会仔细研究每种方案的区别，每种方案的效率，每种方案的适用场合，你会觉得你已经陷入进去拔不出来了。如果是用 EB，你第一秒种就已经做出了决 定，根本没得选择，比如说集合属性，你只能用 Collection，如果是 Hibernate，你会在 Bag，List 和 Set 之间来回犹豫不决，甚至搞 不清楚的话，程序都没有办法写。
3． Spring
它是一个开源的项目，而且目前非常活跃；它基于 IoC（Inversion of Control，反向控制）和 AOP 的构架多层 j2ee 系统的框架，但它不强迫你必须在每一层中必须使用 Spring，因为它模块化的很好，允许你根据自 己的需要选择使用它的某一个模块；它实现了很优雅的 MVC，对不同的数据访问技术提供了统一的接口，采用 IoC 使得可以很容易的实现 bean 的装配，提供 了简洁的 AOP 并据此实现 Transcation Managment，等等
优点：
a. Spring 能有效地组织你的中间层对象，不管你是否选择使用了 EJB。如果你仅仅使用了 Struts 或其他为 J2EE 的 API 特制的 framework，Spring 致力于解决剩下的问题。
b. Spring 能消除在许多工程中常见的对 Singleton 的过多使用。根据我的经验，这是一个很大的问题，它降低了系统的可测试性和面向对象的程度。
c. 通过一种在不同应用程序和项目间一致的方法来处理配置文件， Spring 能消除各种各样自定义格式的属性文件的需要。曾经对某个类要寻找的是哪个魔法般的 属性项或系统属性感到不解，为此不得不去读 Javadoc 甚至源编码？有了 Spring，你仅仅需要看看类的 JavaBean 属性。Inversion of Control 的使用（在下面讨论）帮助完成了这种简化。
d.通过把对接口编程而不是对类编程的代价几乎减少到没有，Spring 能够促进养成好的编程习惯。
e. Spring 被设计为让使用它创建的应用尽可能少的依赖于他的 APIs。在 Spring 应用中的大多数业务对象没有依赖于 Spring。
f. 使用 Spring 构建的应用程序易于单元测试。
g.Spring 能使 EJB 的使用成为一个实现选择,而不是应用架构的必然选择。你能选择用 POJOs 或 local EJBs 来实现业务接口，却不会影响调用代码。
h. Spring 帮助你解决许多问题而无需使用 EJB。Spring 能提供一种 EJB 的替换物，它们适用于许多 web 应用。例如，Spring 能使用 AOP 提供声明性事务管理而不通过 EJB 容器，如果你仅仅需要与单个数据库打交道，甚至不需要一个 JTA 实现。
i. Spring 为数据存取提供了一个一致的框架,不论是使用的是 JDBC 还是 O/R mapping 产品（如 Hibernate）。
Spring 确实使你能通过最简单可行的解决办法来解决你的问题。而这是有有很大价值的。
缺点：
使用人数不多、jsp 中要写很多代码、控制器过于灵活，缺少一个公用控制器。

6. Java web 部分
1、Tomcat 的优化经验
答: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，加大 tomcat 使用的 jvm 的内存

1、HTTP 请求的 GET 与 POST 方式的区别
（1）get 是从服务器上获取数据，post 是向服务器传送数据。
在客户端，Get 方式在通过 URL 提交数据，数据在 URL 中可以看到；POST 方式，数据放置在 HTML HEADER 内提交。
（2） 对于 get 方式，服务器端用 Request.QueryString 获取变量的值，对于 post 方式，服务器端用 Request.Form 获取提交的数据。
（3） GET 方式提交的数据最多只能有 1024 字节，而 POST 则没有此限制。
（4）安全性问题。正如在（1）中提到，使用 Get 的时候，参数会显示在地址栏上，而 Post 不会。所以，如果这些数据是中文数据而且是非敏感数据，那么使用 get；如果用户输入的数据不是中文字符而且包含敏感数据，那么还是使用 post 为好。
62、解释一下什么是 servlet;
答:servlet 有良好的生存期的定义，包括加载和实例化、初始化、处理请求以及服务结束。这个生存期由 javax.servlet.Servlet 接口的 init,service 和 destroy 方法表达。
1、说一说 Servlet 的生命周期?
答:servlet 有良好的生存期的定义，包括加载和实例化、初始化、处理请求以及服务结束。这个生存期由 javax.servlet.Servlet 接口的 init,service 和 destroy 方法表达。

Servlet 被服务器实例化后，容器运行其 init 方法，请求到达时运行其 service 方法，service 方法自动派遣运行与请求对应的 doXXX 方法（doGet，doPost）等，当服务器决定将实例销毁的时候调用其 destroy 方法。
web 容器加载 servlet，生命周期开始。通过调用 servlet 的 init()方法进行 servlet 的初始化。通过调用 service()方法实现，根据请求的不同调用不同的 do***()方法。结束服务，web 容器调用 servlet 的 destroy()方法。

4、Servlet 的基本架构
```
public class ServletName extends HttpServlet {
public void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
}
public void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
}
}
```
3、SERVLET API 中 forward() 与 redirect()的区别？
答:前者仅是容器中控制权的转向，在客户端浏览器地址栏中不会显示出转向后的地址；后者则是完全的跳转，浏览器将会得到跳转的地址，并重新发送请求链接。这样，从浏览器的地址栏中可以看到跳转后的链接地址。所以，前者更加高效，在前者可以满足需要时，尽量使用 forward()方法，并且，这样也有助于隐藏实际的链接。在有些情况下，比如，需要跳转到一个其它服务器上的资源，则必须使用
sendRedirect()方法。

60、什么情况下调用 doGet()和 doPost()？
Jsp 页面中的 FORM 标签里的 method 属性为 get 时调用 doGet()，为 post 时调用 doPost()。

66、Request 对象的主要方法：
setAttribute(String name,Object)：设置名字为 name 的 request 的参数值
getAttribute(String name)：返回由 name 指定的属性值
getAttributeNames()：返回 request 对象所有属性的名字集合，结果是一个枚举的实例
getCookies()：返回客户端的所有 Cookie 对象，结果是一个 Cookie 数组
getCharacterEncoding()：返回请求中的字符编码方式
getContentLength()：返回请求的 Body 的长度
getHeader(String name)：获得 HTTP 协议定义的文件头信息
getHeaders(String name)：返回指定名字的 request Header 的所有值，结果是一个枚举的实例
getHeaderNames()：返回所以 request Header 的名字，结果是一个枚举的实例
getInputStream()：返回请求的输入流，用于获得请求中的数据
getMethod()：获得客户端向服务器端传送数据的方法
getParameter(String name)：获得客户端传送给服务器端的有 name 指定的参数值
getParameterNames()：获得客户端传送给服务器端的所有参数的名字，结果是一个枚举的实例
getParametervalues(String name)：获得有 name 指定的参数的所有值
getProtocol()：获取客户端向服务器端传送数据所依据的协议名称
getQueryString()：获得查询字符串
getRequestURI()：获取发出请求字符串的客户端地址
getRemoteAddr()：获取客户端的 IP 地址
getRemoteHost()：获取客户端的名字
getSession([Boolean create])：返回和请求相关 Session
getServerName()：获取服务器的名字
getServletPath()：获取客户端所请求的脚本文件的路径
getServerPort()：获取服务器的端口号
removeAttribute(String name)：删除请求中的一个属性

19、forward 和 redirect 的区别
forward 是服务器请求资源，服务器直接访问目标地址的 URL，把那个 URL 的响应内容读取过来，然后把这些内容再发给浏览器，浏览器根本不知道服务器发送的内容是从哪儿来的，所以它的地址栏中还是原来的地址。

redirect 就是服务端根据逻辑,发送一个状态码,告诉浏览器重新去请求那个地址，一般来说浏览器会用刚才请求的所有参数重新请求，所以 session,request 参数都可以获取。

4、request.getAttribute() 和 request.getParameter() 有何区别?
1.getAttribute 是取得 jsp 中 用 setAttribute 設定的 attribute
2.parameter 得到的是 string；attribute 得到的是 object
3.request.getParameter ()方法传递的数据，会从 Web 客户端传到 Web 服务器端，代表 HTTP 请求数据；request.setAttribute()和 getAttribute()方法传递的数据只会存在于 Web 容器内部，在具有转发关系的 Web 组件之间共享。即 request.getAttribute()方法返回 request 范围内存在的对象，而 request.getParameter()方法是获取 http 提交过来的数据。

1. jsp 有哪些内置对象?作用分别是什么？分别有什么方法？
答:JSP 共有以下 9 个内置的对象：
request 用户端请求，此请求会包含来自 GET/POST 请求的参数
response 网页传回用户端的回应
pageContext 网页的属性是在这里管理
session 与请求有关的会话期
application servlet 正在执行的内容
out 用来传送回应的输出
config servlet 的构架部件
page JSP 网页本身
exception 针对错误网页，未捕捉的例外

request 表示 HttpServletRequest 对象。它包含了有关浏览器请求的信息，并且提供了几个用于获取 cookie, header, 和 session 数据的有用的方法。

response 表示 HttpServletResponse 对象，并提供了几个用于设置送回 浏览器的响应的方法（如 cookies,头信息等）

out 对象是 javax.jsp.JspWriter 的一个实例，并提供了几个方法使你能用于向浏览器回送输出结果。

pageContext 表示一个 javax.servlet.jsp.PageContext 对象。它是用于方便存取各种范围的名字空间、servlet 相关的对象的 API，并且包装了通用的 servlet 相关功能的方法。

session 表示一个请求的 javax.servlet.http.HttpSession 对象。Session 可以存贮用户的状态信息

applicaton 表示一个 javax.servle.ServletContext 对象。这有助于查找有关 servlet 引擎和 servlet 环境的信息

config 表示一个 javax.servlet.ServletConfig 对象。该对象用于存取 servlet 实例的初始化参数。

page 表示从该页面产生的一个 servlet 实例

2. jsp 有哪些动作?作用分别是什么?
（这个问题似乎不重要，不明白为何有此题）
答:JSP 共有以下 6 种基本动作
jsp:include：在页面被请求的时候引入一个文件。
jsp:useBean：寻找或者实例化一个 JavaBean。
jsp:setProperty：设置 JavaBean 的属性。
jsp:getProperty：输出某个 JavaBean 的属性。
jsp:forward：把请求转到一个新的页面。
jsp:plugin：根据浏览器类型为 Java 插件生成 OBJECT 或 EMBED 标记
59、JSP 的常用指令
isErrorPage(是否能使用 Exception 对象)，isELIgnored(是否忽略表达式)

3. JSP 中动态 INCLUDE 与静态 INCLUDE 的区别？
答：动态 INCLUDE 用 jsp:include 动作实现
<jsp:include page=included.jsp flush=true />它总是会检查所含文件中的变化，适合用于包含动态页面，并且可以带参数 静态 INCLUDE 用 include 伪码实现,定不会检查所含文件的变化，

适用于包含静态页面 <%@ include file=included.htm %>

4、两种跳转方式分别是什么?有什么区别?
（下面的回答严重错误，应该是想问 forward 和 sendRedirect 的区别，毕竟出题的人不是专业搞文字艺术的人，可能表达能力并不见得很强，用词不一定精准，加之其自身的技术面也可能存在一些问题，不一定真正将他的意思表达清楚了，严格意思上来讲，一些题目可能根本就无人能答，所以，答题时要掌握主动，只要把自己知道的表达清楚就够了，而不要去推敲原始题目的具体含义是什么，不要一味想着是在答题）
答：有两种，分别为：
<jsp:include page=included.jsp flush=true>
<jsp:forward page= nextpage.jsp/>
前者页面不会转向 include 所指的页面，只是显示该页的结果，主页面还是原来的页面。执行完后还会回来，相当于函数调用。并且可以带参数.后者完全转向新页面，不会再回来。相当于 go to 语句。

63、页面间对象传递的方法
request，session，application，cookie 等
64、JSP 和 Servlet 有哪些相同点和不同点，他们之间的联系是什么？
JSP 是 Servlet 技术的扩展，本质上是 Servlet 的简易方式，更强调应用的外表表达。JSP 编译后是"类 servlet"。Servlet 和 JSP 最主要的不同点在于，Servlet 的应用逻辑是在 Java 文件中，并且完全从表示层中的 HTML 里分离开来。而 JSP 的情况是 Java 和 HTML 可以组合成一个扩展名为.jsp 的文件。JSP 侧重于视图，Servlet 主要用于控制逻辑。

1、MVC 的各个部分都有那些技术来实现?如何实现?
答:MVC 是 Model－View－Controller 的简写。Model 代表的是应用的业务逻辑（通过 JavaBean，EJB 组件实现）， View 是应用的表示面（由 JSP 页面产生），Controller 是提供应用的处理过程控制（一般是一个 Servlet），通过这种设计模型把应用逻辑，处理过程和显示逻辑分成不同的组件实现。这些组件可以进行交互和重用。

68、我们在 web 应用开发过程中经常遇到输出某种编码的字符，如 iso8859-1 等，如何输出一个某种编码的字符串？

```
Public String translate (String str) {
    String tempStr = "";
    try {
       tempStr = new String(str.getBytes("ISO-8859-1"), "GBK");
       tempStr = tempStr.trim();
    }
    catch (Exception e) {
       System.err.println(e.getMessage());
    }
    return tempStr;
}
```

1．现在输入 n 个数字，以逗号，分开；然后可选择升或者降序排序；按提交键就在另一页面显示按什么排序，结果为，提供 reset

7. 实际项目开发

1、在 eclipse 中调试时，怎样查看一个变量的值？

在要查看的变量前先设置断点，然后选中变量，右键选 debug as-->Java Application，打开 debug 透视图，这时在 Variables 窗口中可以看到变量当前的值。

如果是局部变量，也可以在局部变量窗口中查看。

要知道一个方法被调用的方法调用链，可以在方法栈中查看。

2、你们公司使用的代码配置管理工具是什么？

除了说以前使用 cvs，现在新项目使用 svn 了，还简要说一下使用的过程，如果有可能，还说说仓库的概念和如何使用锁之类的细节。

3、你们的项目总金额多少，多少人开发，总共花了多少个月？

像巴巴运动网这种规模的项目，可以说是 4、5 个人、开发了 4、5 个月，费用则是 4、50 万。按每人每月两万收入去计算，就差不多了。

7. 数据库部分

1、用两种方式根据部门号从高到低，工资从低到高列出每个员工的信息。

employee:

eid,ename,salary,deptid;

select * from employee order by deptid desc,salary

2、列出各个部门中工资高于本部门的平均工资的员工数和部门号，并按部门号排序

创建表：

mysql> create table employee921(id int primary key auto_increment,name varchar(50),salary bigint,deptid int);

插入实验数据：

mysql> insert into employee921 values(null,'zs',1000,1),(null,'ls',1100,1),(null,'ww',1100,1),(null,'zl',900,1) ,(null,'zl',1000,2), (null,'zl',900,2) ,(null,'zl',1000,2) , (null,'zl',1100,2);

编写 sql 语句：

（）select avg(salary) from employee921 group by deptid;

（）mysql> select employee921.id,employee921.name,employee921.salary,employee921.deptid tid from employee921 where salary > (select avg(salary) from employee921 where deptid = tid);

效率低的一个语句，仅供学习参考使用（在 group by 之后不能使用 where，只能使用 having，在 group by 之前可以使用 where，即表示对过滤后的结果分组）：

mysql> select employee921.id,employee921.name,employee921.salary,employee921.deptid tid from employee921 where salary > (select avg(salary) from employee921 group by deptid having deptid = tid);

（）select count(*) ,tid

```
    from (
        select employee921.id,employee921.name,employee921.salary,employee921.deptid tid
        from    employee921
        where salary >
            (select avg(salary) from employee921 where  deptid = tid)
    ) as t
    group by tid ;
```

另外一种方式：关联查询

```
select a.ename,a.salary,a.deptid
 from emp a,
    (select deptd,avg(salary) avgsal from emp group by deptid ) b
 where a.deptid=b.deptid and a.salary>b.avgsal;
```

1、存储过程与触发器必须讲，经常被面试到?

```
create procedure insert_Student (_name varchar(50),_age int ,out _id int)
begin
    insert into student value(null,_name,_age);
    select max(stuId) into _id from student;
end;

call insert_Student('wfz',23,@id);
select @id;
```

```
mysql> create trigger update_Student BEFORE update on student FOR EACH ROW
-> select * from student;
```

触发器不允许返回结果

```
create trigger update_Student BEFORE update on student FOR EACH ROW
insert into  student value(null,'zxx',28);
```

mysql 的触发器目前不能对当前表进行操作

```
create trigger update_Student BEFORE update on student FOR EACH ROW
delete from articles  where id=8;
```

这个例子不是很好，最好是用删除一个用户时，顺带删除该用户的所有帖子

这里要注意使用 OLD.id

触发器用处还是很多的，比如校内网、开心网、Facebook，你发一个日志，自动通知好友，其实就是在增加日志时做一个后触发，再向通知表中写入条目。因为触发器效率高。而 UCH 没有用触发器，效率和数据处理能力都很低。

存储过程的实验步骤：

```
mysql> delimiter |
mysql> create procedure insertArticle_Procedure (pTitle varchar(50),pBid int,out
 pId int)
    -> begin
```

```
    -> insert into article1 value(null,pTitle,pBid);
    -> select max(id) into pId from article1;
    -> end;
    -> |
Query OK, 0 rows affected (0.05 sec)

mysql> call insertArticle_Procedure('传智播客',1,@pid);
    -> |
Query OK, 0 rows affected (0.00 sec)

mysql> delimiter ;
mysql> select @pid;
+------+
| @pid |
+------+
| 3    |
+------+
1 row in set (0.00 sec)

mysql> select * from article1;
+----+--------------+------+
| id | title        | bid  |
+----+--------------+------+
| 1  | test         | 1    |
| 2  | chuanzhiboke | 1    |
| 3  | 传智播客     | 1    |
+----+--------------+------+
3 rows in set (0.00 sec)
```

触发器的实验步骤：

```
create table board1(id int primary key auto_increment,name varchar(50),ar
ticleCount int);

create table article1(id int primary key auto_increment,title varchar(50)
,bid int references board1(id));

delimiter |

create trigger insertArticle_Trigger after insert on article1 for each ro
w begin
    -> update board1 set articleCount=articleCount+1 where id= NEW.bid;
    -> end;
    -> |
```

delimiter ;

insert into board1 value (null,'test',0);

insert into article1 value(null,'test',1);
还有，每插入一个帖子，都希望将版面表中的最后发帖时间，帖子总数字段进行同步更新，用触发器做效率就很高。下次课设计这样一个案例，写触发器时，对于最后发帖时间可能需要用 declare 方式声明一个变量，或者是用 NEW.posttime 来生成。

1、数据库三范式是什么?
第一范式（1NF）：字段具有原子性,不可再分。所有关系型数据库系统都满足第一范式）
数据库表中的字段都是单一属性的，不可再分。例如，姓名字段，其中的姓和名必须作为一个整体，无法区分哪部分是姓，哪部分是名，如果要区分出姓和名，必须设计成两个独立的字段。

第二范式（2NF）：
第二范式（2NF）是在第一范式（1NF）的基础上建立起来的，即满足第二范式（2NF）必须先满足第一范式（1NF）。
要求数据库表中的每个实例或行必须可以被惟一地区分。通常需要为表加上一个列，以存储各个实例的惟一标识。这个惟一属性列被称为主关键字或主键。

第二范式（2NF）要求实体的属性完全依赖于主关键字。所谓完全依赖是指不能存在仅依赖主关键字一部分的属性，如果存在，那么这个属性和主关键字的这一部分应该分离出来形成一个新的实体，新实体与原实体之间是一对多的关系。为实现区分通常需要为表加上一个列，以存储各个实例的惟一标识。简而言之，第二范式就是非主属性非部分依赖于主关键字。

第三范式的要求如下：
满足第三范式（3NF）必须先满足第二范式（2NF）。简而言之，第三范式（3NF）要求一个数据库表中不包含已在其它表中已包含的非主关键字信息。
所以第三范式具有如下特征：
1，每一列只有一个值
2，每一行都能区分。
3，每一个表都不包含其他表已经包含的非主关键字信息。
例如，帖子表中只能出现发帖人的 id，而不能出现发帖人的 id，还同时出现发帖人姓名，否则，只要出现同一发帖人 id 的所有记录，它们中的姓名部分都必须严格保持一致，这就是数据冗余。

1、说出一些数据库优化方面的经验?
用 PreparedStatement 一般来说比 Statement 性能高：一个 sql 发给服务器去执行，涉及步骤：语法检查、语义分析， 编译，缓存
“inert into user values(1,1,1)”-?二进制
“inert into user values(2,2,2)”-?二进制
“inert into user values(?,?,?)”-?二进制

有外键约束会影响插入和删除性能，如果程序能够保证数据的完整性，那在设计数据库时就去掉外键。（比喻：就好比免检产品，就是为了提高效率，充分相信产品的制造商）
（对于 hibernate 来说，就应该有一个变化：empleyee->Deptment 对象，现在设计时就成了 employee?deptid）

看 mysql 帮助文档子查询章节的最后部分，例如，根据扫描的原理，下面的子查询语句要比第二条关联查询的效率高：
1. select e.name,e.salary where e.managerid=(select id from employee where name='zxx');

2. select e.name,e.salary,m.name,m.salary from employees e,employees m where e.managerid = m.id and m.name='zxx';

表中允许适当冗余，譬如，主题帖的回复数量和最后回复时间等
将姓名和密码单独从用户表中独立出来。这可以是非常好的一对一的案例哟！

sql 语句全部大写，特别是列名和表名都大写。特别是 sql 命令的缓存功能，更加需要统一大小写，sql 语句?发给 oracle 服务器?语法检查和编译成为内部指令?缓存和执行指令。根据缓存的特点，不要拼凑条件，而是用?和 PreparedStatment

还有索引对查询性能的改进也是值得关注的。

备注：下面是关于性能的讨论举例

4 航班 3 个城市

m*n

select * from flight,city where flight.startcityid=city.cityid and city.name='beijing';

m + n

select * from flight where startcityid = (select cityid from city where cityname='beijing');

select flight.id,'beijing',flight.flightTime from flight where startcityid = (select cityid from city where cityname='beijing')
2、union 和 union all 有什么不同?
假设我们有一个表 Student，包括以下字段与数据：
drop table student;
create table student
(
id int primary key,

```
name nvarchar2(50) not null,
score number not null
);
insert into student values(1,'Aaron',78);
insert into student values(2,'Bill',76);
insert into student values(3,'Cindy',89);
insert into student values(4,'Damon',90);
insert into student values(5,'Ella',73);
insert into student values(6,'Frado',61);
insert into student values(7,'Gill',99);
insert into student values(8,'Hellen',56);
insert into student values(9,'Ivan',93);
insert into student values(10,'Jay',90);
commit;
```

Union 和 Union All 的区别。

```
select *
from student
where id < 4
union
select *
from student
where id > 2 and id < 6
```

结果将是

```
1    Aaron    78
2    Bill    76
3    Cindy    89
4    Damon    90
5    Ella    73
```

如果换成 Union All 连接两个结果集，则返回结果是：

```
1    Aaron    78
2    Bill    76
3    Cindy    89
3    Cindy    89
4    Damon    90
5    Ella    73
```

可以看到，Union 和 Union All 的区别之一在于对重复结果的处理。

UNION 在进行表链接后会筛选掉重复的记录，所以在表链接后会对所产生的结果集进行排序运算，删除重复的记录再返回结果。实际大部分应用中是不会产生重复的记录，最常见的是过程表与历史表 UNION。如：

```
select * from gc_dfys
union
select * from ls_jg_dfys
```

这个 SQL 在运行时先取出两个表的结果，再用排序空间进行排序删除重复的记录，最

后返回结果集，如果表数据量大的话可能会导致用磁盘进行排序。

而 UNION ALL 只是简单的将两个结果合并后就返回。这样，如果返回的两个结果集中有重复的数据，那么返回的结果集就会包含重复的数据了。

从效率上说，UNION ALL 要比 UNION 快很多，所以，如果可以确认合并的两个结果集中不包含重复的数据的话，那么就使用 UNION ALL，

3.分页语句

取出 sql 表中第 31 到 40 的记录（以自动增长 ID 为主键）

sql server 方案 1：

```
select top 10 * from t where id not in (select top 30 id from t order by id ) orde by id
```

sql server 方案 2：

```
select top 10 * from t where id in (select top 40 id from t order by id) order by id desc
```

mysql 方案：select * from t order by id limit 30,10

oracle 方案：select * from (select rownum r,* from t where r<=40) where r>30

----------待整理进去的内容----------

pageSize=20;

pageNo = 5;

1.分页技术 1（直接利用 sql 语句进行分页，效率最高和最推荐的）

```
mysql:sql = "select * from articles limit " + (pageNo-1)*pageSize + "," + pageSize;
oracle: sql = "select * from " +
                                        "(select rownum r,* from " +
                                            "(select * from articles order by postime desc)" +
                                        "where rownum<= " + pageNo*pageSize +") tmp " +
                                    "where r>" + (pageNo-1)*pageSize;
```

注释：第 7 行保证 rownum 的顺序是确定的，因为 oracle 的索引会造成 rownum 返回不同的值

简洋提示：没有 order by 时，rownum 按顺序输出，一旦有了 order by，rownum 不按顺序输出了，这说明 rownum 是排序前的编号。如果对 order by 从句中的字段建立了索引，那么，rownum 也是按顺序输出的，因为这时候生成原始的查询结果集时会参照索引表的顺序来构建。

```
sqlserver:sql = "select top 10 * from id not id(select top " + (pageNo-1)*pageSize + "id from articles)"

DataSource ds = new InitialContext().lookup(jndiurl);
Connection cn = ds.getConnection();
//"select * from user where id=?"    --->binary directive
PreparedStatement pstmt = cn.prepareSatement(sql);
ResultSet rs = pstmt.executeQuery()
while(rs.next())
```

```
{
	out.println(rs.getString(1));
}
```

2.不可滚动的游标

```
pageSize=20;
pageNo = 5;
cn = null
stmt = null;
rs = null;
try
{
sqlserver:sql = "select    * from articles";

DataSource ds = new InitialContext().lookup(jndiurl);
Connection cn = ds.getConnection();
//"select * from user where id=?"    --->binary directive
PreparedStatement pstmt = cn.prepareSatement(sql);
ResultSet rs = pstmt.executeQuery()
for(int j=0;j<(pageNo-1)*pageSize;j++)
{
	rs.next();
}

int i=0;

while(rs.next() && i<10)
{
	i++;
	out.println(rs.getString(1));
}
}
cacth(){}
finnaly
{
	if(rs!=null)try{rs.close();}catch(Exception e){}
	if(stm.....
	if(cn.....
}
```

3.可滚动的游标

```
pageSize=20;
pageNo = 5;
cn = null
```

```
stmt = null;
rs = null;
try
{
sqlserver:sql = "select    * from articles";

DataSource ds = new InitialContext().lookup(jndiurl);
Connection cn = ds.getConnection();
//"select * from user where id=?"    --->binary directive
PreparedStatement pstmt = cn.prepareSatement(sql,ResultSet.TYPE_SCROLL_INSENSITIVE,...);
//根据上面这行代码的异常 SQLFeatureNotSupportedException，就可判断驱动是否支持可滚动游标

ResultSet rs = pstmt.executeQuery()
rs.absolute((pageNo-1)*pageSize)
int i=0;
while(rs.next() && i<10)
{
      i++;
      out.println(rs.getString(1));
}
}
cacth(){}
finnaly
{
      if(rs!=null)try{rs.close();}catch(Exception e){}
      if(stm.....
      if(cn.....
}
```

3.用一条 SQL 语句 查询出每门课都大于 80 分的学生姓名

name	kecheng	fenshu
张三	语文	81
张三	数学	75
李四	语文	76
李四	数学	90
王五	语文	81
王五	数学	100
王五	英语	90

准备数据的 sql 代码：

```
create table score(id int primary key auto_increment,name varchar(20),subject varchar(20),score int);
insert into score values
```

(null,'张三','语文',81),
(null,'张三','数学',75),
(null,'李四','语文',76),
(null,'李四','数学',90),
(null,'王五','语文',81),
(null,'王五','数学',100),
(null,'王五 ','英语',90);

提示：当百思不得其解时，请理想思维，把小变成大做，把大变成小做，

答案：
A: select distinct name from score where name not in (select distinct name from score where score<=80)

B:select distince name t1 from score where 80< all (select score from score where name=t1);

4.所有部门之间的比赛组合
一个叫 department 的表，里面只有一个字段 name,一共有 4 条纪录，分别是 a,b,c,d,对应四个球对，现在四个球对进行比赛，用一条 sql 语句显示所有可能的比赛组合.

答：select a.name, b.name
from team a, team b
where a.name < b.name

4.每个月份的发生额都比 101 科目多的科目
请用 SQL 语句实现：从 TestDB 数据表中查询出所有月份的发生额都比 101 科目相应月份的发生额高的科目。请注意：TestDB 中有很多科目，都有 1－12 月份的发生额。
AccID：科目代码，Occmonth：发生额月份，DebitOccur：发生额。
数据库名：JcyAudit，数据集：Select * from TestDB
准备数据的 sql 代码：
drop table if exists TestDB;
create table TestDB(id int primary key auto_increment,AccID varchar(20), Occmonth date, DebitOccur bigint);
insert into TestDB values
(null,'101','1988-1-1',100),
(null,'101','1988-2-1',110),
(null,'101','1988-3-1',120),
(null,'101','1988-4-1',100),
(null,'101','1988-5-1',100),
(null,'101','1988-6-1',100),
(null,'101','1988-7-1',100),
(null,'101','1988-8-1',100);
--复制上面的数据，故意把第一个月份的发生额数字改小一点
insert into TestDB values

(null,'102','1988-1-1',90),
(null,'102','1988-2-1',110),
(null,'102','1988-3-1',120),
(null,'102','1988-4-1',100),
(null,'102','1988-5-1',100),
(null,'102','1988-6-1',100),
(null,'102','1988-7-1',100),
(null,'102','1988-8-1',100);
--复制最上面的数据，故意把所有发生额数字改大一点
insert into TestDB values
(null,'103','1988-1-1',150),
(null,'103','1988-2-1',160),
(null,'103','1988-3-1',180),
(null,'103','1988-4-1',120),
(null,'103','1988-5-1',120),
(null,'103','1988-6-1',120),
(null,'103','1988-7-1',120),
(null,'103','1988-8-1',120);
--复制最上面的数据，故意把所有发生额数字改大一点
insert into TestDB values
(null,'104','1988-1-1',130),
(null,'104','1988-2-1',130),
(null,'104','1988-3-1',140),
(null,'104','1988-4-1',150),
(null,'104','1988-5-1',160),
(null,'104','1988-6-1',170),
(null,'104','1988-7-1',180),
(null,'104','1988-8-1',140);
--复制最上面的数据，故意把第二个月份的发生额数字改小一点
insert into TestDB values
(null,'105','1988-1-1',100),
(null,'105','1988-2-1',80),
(null,'105','1988-3-1',120),
(null,'105','1988-4-1',100),
(null,'105','1988-5-1',100),
(null,'105','1988-6-1',100),
(null,'105','1988-7-1',100),
(null,'105','1988-8-1',100);
答案：
```
select distinct AccID from TestDB
where AccID not in
    (select TestDB.AccIDfrom TestDB,
        (select * from TestDB where AccID='101') as db101
    where TestDB.Occmonth=db101.Occmonth and TestDB.DebitOccur<=db101.DebitOccur
```

);

4.统计每年每月的信息
year month amount
1991 1 1.1
1991 2 1.2
1991 3 1.3
1991 4 1.4
1992 1 2.1
1992 2 2.2
1992 3 2.3
1992 4 2.4
查成这样一个结果
year m1 m2 m3 m4
1991 1.1 1.2 1.3 1.4
1992 2.1 2.2 2.3 2.4

提示：这个与工资条非常类似，与学生的科目成绩也很相似。

准备 sql 语句：
drop table if exists sales;
create table sales(id int auto_increment primary key,year varchar(10), month varchar(10), amount float(2,1));
insert into sales values
(null,'1991','1',1.1),
(null,'1991','2',1.2),
(null,'1991','3',1.3),
(null,'1991','4',1.4),
(null,'1992','1',2.1),
(null,'1992','2',2.2),
(null,'1992','3',2.3),
(null,'1992','4',2.4);

答案一、
select sales.year ,
(select t.amount from sales t where t.month='1' and t.year= sales.year) '1',
(select t.amount from sales t where t.month='1' and t.year= sales.year) '2',
(select t.amount from sales t where t.month='1' and t.year= sales.year) '3',
(select t.amount from sales t where t.month='1' and t.year= sales.year) as '4'
from sales group by year;

5.显示文章标题，发帖人、最后回复时间
表：id,title,postuser,postdate,parentid
准备 sql 语句：

```
drop table if exists articles;
create table articles(id int auto_increment primary key,title varchar(50), postuser varchar(10), postdate datetime,parentid int references articles(id));
insert into articles values
(null,'第一条','张三','1998-10-10 12:32:32',null),
(null,'第二条','张三','1998-10-10 12:34:32',null),
(null,'第一条回复 1','李四','1998-10-10 12:35:32',1),
(null,'第二条回复 1','李四','1998-10-10 12:36:32',2),
(null,'第一条回复 2','王五','1998-10-10 12:37:32',1),
(null,'第一条回复 3','李四','1998-10-10 12:38:32',1),
(null,'第二条回复 2','李四','1998-10-10 12:39:32',2),
(null,'第一条回复 4','王五','1998-10-10 12:39:40',1);
```

答案：

```
select a.title,a.postuser,
    (select max(postdate) from articles where parentid=a.id) reply
from articles a where a.parentid is null;
```

注释：子查询可以用在选择列中，也可用于 where 的比较条件中，还可以用于 from 从句中。

3.删除除了 id 号不同,其他都相同的学生冗余信息

2.学生表 如下:

id 号	学号	姓名	课程编号	课程名称	分数
1	2005001	张三	0001	数学	69
2	2005002	李四	0001	数学	89
3	2005001	张三	0001	数学	69

A: delete from tablename where id 号 not in(select min(id 号) from tablename group by 学号,姓名,课程编号,课程名称,分数)

实验：

```
create table student2(id int auto_increment primary key,code varchar(20),name varchar(20));
insert into student2 values(null,'2005001','张三'),(null,'2005002','李四'),(null,'2005001','张三');
```

//如下语句，mysql 报告错误，可能删除依赖后面统计语句，而删除又导致统计语句结果不一致。

```
delete from student2 where id not in(select min(id) from student2 group by name);
```

//但是，如下语句没有问题：

```
select *  from student2 where id not in(select min(id) from student2 group by name);
```

//于是，我想先把分组的结果做成虚表，然后从虚表中选出结果，最后再将结果作为删除的条件数据。

```
delete from student2 where id not in(select mid from (select min(id) mid
from student2 group by name) as t);
```

或者：

```
delete from student2 where id not in(select min(id) from (select * from s
tudent2) as t group by t.name);
```

4.航空网的几个航班查询题：
表结构如下：
flight{flightID,StartCityID ,endCityID,StartTime}
city{cityID, CityName)
实验环境：

```
create table city(cityID int auto_increment primary key,cityName varchar(20));
create table flight (flightID int auto_increment primary key,
        StartCityID int references city(cityID),
        endCityID    int references city(cityID),
        StartTime timestamp);
//航班本来应该没有日期部分才好，但是下面的题目当中涉及到了日期
insert into city values(null,'北京'),(null,'上海'),(null,'广州');
insert into flight values
        (null,1,2,'9:37:23'),(null,1,3,'9:37:23'),(null,1,2,'10:37:23'),(null,2,3,'10:37:23');
```

1、查询起飞城市是北京的所有航班，按到达城市的名字排序

参与运算的列是我起码能够显示出来的那些列，但最终我不一定把它们显示出来。各个表组合出来的中间结果字段中必须包含所有运算的字段。

```
   select    * from flight f,city c
        where f.endcityid = c.cityid and startcityid =
        (select c1.cityid from city c1 where c1.cityname = "北京")
        order by c.cityname asc;
```

```
mysql> select flight.flightid,'北京' startcity, e.cityname from flight,city e wh
ere flight.endcityid=e.cityid and flight.startcityid=(select cityid from city wh
ere cityname='北京');
```

```
mysql> select flight.flightid,s.cityname,e.cityname from flight,city s,city e wh
ere flight.startcityid=s.cityid and s.cityname='北京' and flight.endCityId=e.cit
yID order by e.cityName desc;
```

2、查询北京到上海的所有航班纪录（起飞城市，到达城市，起飞时间，航班号）

```
select c1.CityName,c2.CityName,f.StartTime,f.flightID
from city c1,city c2,flight f
where f.StartCityID=c1.cityID
and f.endCityID=c2.cityID
and c1.cityName='北京'
and c2.cityName='上海'
```

3、查询具体某一天（2005-5-8）的北京到上海的的航班次数

```
select count(*) from
(select c1.CityName,c2.CityName,f.StartTime,f.flightID
from city c1,city c2,flight f
where f.StartCityID=c1.cityID
and f.endCityID=c2.cityID
and c1.cityName='北京'
and c2.cityName='上海'
and  查帮助获得的某个日期处理函数(startTime) like '2005-5-8%'
```

mysql 中提取日期部分进行比较的示例代码如下：

```
select * from flight where date_format(starttime,'%Y-%m-%d')='1998-01-02'
```

5.查出比经理薪水还高的员工信息：

```
Drop table if not exists employees;
create table employees(id int primary key auto_increment,name varchar(50)
,salary int,managerid int references employees(id));
insert into employees values (null,' lhm',10000,null), (null,' zxx',15000,1
),(null,'flx',9000,1),(null,'tg',10000,2),(null,'wzg',10000,3);
```

Wzg 大于 flx,lhm 大于 zxx

解题思路：

根据 sql 语句的查询特点，是逐行进行运算，不可能两行同时参与运算。

涉及了员工薪水和经理薪水，所有，一行记录要同时包含两个薪水，所有想到要把这个表自关联组合一下。

首先要组合出一个包含有各个员工及该员工的经理信息的长记录，譬如，左半部分是员工，右半部分是经理。而迪卡尔积会组合出很多垃圾信息，先去除这些垃圾信息。

```
select e.* from employees e,employees m where e.managerid=m.id and e.sala
ry>m.salary;
```

6、求出小于 45 岁的各个老师所带的大于 12 岁的学生人数

数据库中有 3 个表 teacher 表，student 表，tea_stu 关系表。

teacher 表 teaID name age

student 表 stuID name age

teacher_student 表 teaID stuID

要求用一条 sql 查询出这样的结果

1.显示的字段要有老师 name, age 每个老师所带的学生人数

2 只列出老师 age 为 40 以下，学生 age 为 12 以上的记录

预备知识：

1.sql 语句是对每一条记录依次处理，条件为真则执行动作（select,insert,delete,update）

2.只要是迪卡尔积，就会产生“垃圾”信息，所以，只要迪卡尔积了，我们首先就要想到清除“垃圾”信息

实验准备：

```
drop table if exists tea_stu;
drop table if exists teacher;
```

```
drop table if exists student;
    create table teacher(teaID int primary key,name varchar(50),age int);
    create table student(stuID int primary key,name varchar(50),age int);
    create table tea_stu(teaID int references teacher(teaID),stuID int references student(stuID));
insert into teacher values(1,'zxx',45), (2,'lhm',25) , (3,'wzg',26) , (4,'tg',27);
insert into student values(1,'wy',11), (2,'dh',25) , (3,'ysq',26) , (4,'mxc',27);
insert into tea_stu values(1,1), (1,2), (1,3);
insert into tea_stu values(2,2), (2,3), (2,4);
 insert into tea_stu values(3,3), (3,4), (3,1);
insert into tea_stu values(4,4), (4,1), (4,2) , (4,3);
```

结果：2?3,3?2,4?3

解题思路：（真实面试答题时，也要写出每个分析步骤，如果纸张不够，就找别人要）
1 要会统计分组信息，统计信息放在中间表中：

```
select teaid,count(*) from tea_stu group by teaid;
```

2 接着其实应该是筛除掉小于 12 岁的学生，然后再进行统计，中间表必须与 student 关联才能得到 12 岁以下学生和把该学生记录从中间表中剔除，代码是：

```
select tea_stu.teaid,count(*) total from student,tea_stu
where student.stuid=tea_stu.stuid and student.age>12 group by tea_stu.teaid
```

3.接着把上面的结果做成虚表与 teacher 进行关联，并筛除大于 45 的老师

```
select teacher.teaid,teacher.name,total from teacher ,(select tea_stu.tea
id,count(*) total from student,tea_stu where student.stuid=tea_stu.stuid and stu
dent.age>12 group by tea_stu.teaid) as tea_stu2 where teacher.teaid=tea_stu2.tea
id and teacher.age<45;
```

7.求出发帖最多的人：

```
select authorid,count(*) total from articles
group by authorid
having total=
(select max(total2) from (select count(*) total2 from articles group by authorid) as t);

select t.authorid,max(t.total) from
（select authorid,count(*) total from articles ）as t
```

这条语句不行，因为 max 只有一列，不能与其他列混淆。

```
select authorid,count(*) total from articles
group by authorid having total=max(total)
```

也不行。

10、一个用户表中有一个积分字段，假如数据库中有 100 多万个用户，若要在每年第一天凌

晨将积分清零，你将考虑什么，你将想什么办法解决?
alter table drop column score;
alter table add colunm score int;
可能会很快，但是需要试验，试验不能拿真实的环境来操刀，并且要注意，
这样的操作时无法回滚的，在我的印象中，只有 inert update delete 等 DML 语句才能回滚，
对于 create table,drop table ,alter table 等 DDL 语句是不能回滚。

解决方案一，update user set score=0;
解决方案二，假设上面的代码要执行好长时间，超出我们的容忍范围，那我就 alter table user drop column score;alter table user add column score int。

下面代码实现每年的那个凌晨时刻进行清零。

```
Runnable runnable =
      new Runnable(){
            public void run(){
                  clearDb();
                  schedule(this,new Date(new Date().getYear()+1,0,0));
                  }
                  };

schedule(runnable,
      new Date(new Date().getYear()+1,0,1));
```

10、一个用户具有多个角色，请查询出该表中具有该用户的所有角色的其他用户。

```
select count(*) as num,tb.id
from
  tb,
  (select role from tb where id=xxx) as t1
where
  tb.role = t1.role and tb.id != t1.id
group by tb.id
having
      num = select count(role) from tb where id=xxx;
```

8. xxx 公司的 sql 面试

```
Table EMPLOYEES Structure:
EMPLOYEE_ID          NUMBER              Primary Key,
FIRST_NAME           VARCHAR2(25),
LAST_NAME            VARCHAR2(25),
Salary number(8,2),
HiredDate DATE,
Departmentid number(2)
Table Departments Structure:
Departmentid number(2)              Primary Key,
```

DepartmentName　VARCHAR2(25).

（2）基于上述 EMPLOYEES 表写出查询：写出雇用日期在今年的，或者工资在[1000,2000]之间的，或者员工姓名（last_name）以’Obama’打头的所有员工，列出这些员工的全部个人信息。（4 分）

```
select * from employees
where Year(hiredDate) = Year(date())
      or (salary between 1000 and 200)
      or left(last_name,3)='abc';
```

(3) 基于上述 EMPLOYEES 表写出查询：查出部门平均工资大于 1800 元的部门的所有员工，列出这些员工的全部个人信息。（4 分）

```
mysql> select id,name,salary,deptid did from employee1 where (select avg(salary)
  from employee1 where deptid = did) > 1800;
```

(4) 基于上述 EMPLOYEES 表写出查询：查出个人工资高于其所在部门平均工资的员工，列出这些员工的全部个人信息及该员工工资高出部门平均工资百分比。（5 分）

```
select employee1.*,(employee1.salary-t.avgSalary)*100/employee1.salary
from employee1,
      (select deptid,avg(salary) avgSalary from employee1 group by deptid) as t
where employee1.deptid = t.deptid and employee1.salary>t.avgSalary;
```

1、注册 Jdbc 驱动程序的三种方式

1、用 JDBC 如何调用存储过程

代码如下：

```
package com.huawei.interview.lym;

import java.sql.CallableStatement;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Types;

public class JdbcTest {

      /**
        * @param args
        */
      public static void main(String[] args) {
            // TODO Auto-generated method stub
            Connection cn = null;
            CallableStatement cstmt = null;
            try {
```

```
            //这里最好不要这么干，因为驱动名写死在程序中了
            Class.forName("com.mysql.jdbc.Driver");
            //实际项目中，这里应用 DataSource 数据，如果用框架，
            //这个数据源不需要我们编码创建，我们只需 Datasource ds = context.lookup()
            //cn = ds.getConnection();
            cn = DriverManager.getConnection("jdbc:mysql:///test","root","root");
            cstmt = cn.prepareCall("{call insert_Student(?,?,?)}");
            cstmt.registerOutParameter(3,Types.INTEGER);
            cstmt.setString(1, "wangwu");
            cstmt.setInt(2, 25);
            cstmt.execute();
            //get 第几个，不同的数据库不一样，建议不写
            System.out.println(cstmt.getString(3));
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        finally
        {

            /*try{cstmt.close();}catch(Exception e){}
            try{cn.close();}catch(Exception e){}*/
            try {
                if(cstmt != null)
                    cstmt.close();
                if(cn != null)
                    cn.close();
            } catch (SQLException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }
```

1、JDBC 中的 PreparedStatement 相比 Statement 的好处

答：一个 sql 命令发给服务器去执行的步骤为：语法检查，语义分析，编译成内部指令，缓存指令，执行指令等过程。

select * from student where id =3----缓存--?xxxxx 二进制命令

select * from student where id =3----直接取-?xxxxx 二进制命令

select * from student where id =4--- -?会怎么干？

如果当初是 select * from student where id =?--- -?又会怎么干？

上面说的是性能提高

可以防止 sql 注入。

1. 写一个用 jdbc 连接并访问 oracle 数据的程序代码

2、Class.forName 的作用?为什么要用?

答：按参数中指定的字符串形式的类名去搜索并加载相应的类，如果该类字节码已经被加载过，则返回代表该字节码的 Class 实例对象，否则，按类加载器的委托机制去搜索和加载该类，如果所有的类加载器都无法加载到该类，则抛出 ClassNotFoundException。加载完这个 Class 字节码后，接着就可以使用 Class 字节码的 newInstance 方法去创建该类的实例对象了。有时候，我们程序中所有使用的具体类名在设计时（即开发时）无法确定，只有程序运行时才能确定，这时候就需要使用 Class.forName 去动态加载该类，这个类名通常是在配置文件中配置的，例如，spring 的 ioc 中每次依赖注入的具体类就是这样配置的，jdbc 的驱动类名通常也是通过配置文件来配置的，以便在产品交付使用后不用修改源程序就可以更换驱动类名。

3、大数据量下的分页解决方法。

答：最好的办法是利用 sql 语句进行分页，这样每次查询出的结果集中就只包含某页的数据内容。再 sql 语句无法实现分页的情况下，可以考虑对大的结果集通过游标定位方式来获取某页的数据。

sql 语句分页，不同的数据库下的分页方案各不一样，下面是主流的三种数据库的分页 sql：

sql server:

```
String sql =
"select top " + pageSize + " * from students where id not in" +

 "(select top " + pageSize * (pageNumber-1) + " id from students order by id)" +

 "order by id";
```

mysql:

```
String sql =
"select * from students order by id limit " + pageSize*(pageNumber-1) + "," + pageSize;
```

oracle:

```
String sql =
  "select * from " +
  (select *,rownum rid from (select * from students order by postime desc) where rid<=" +
pagesize*pagenumber + ") as t" +
  "where t>" + pageSize*(pageNumber-1);
```

4、用 JDBC 查询学生成绩单, 把主要代码写出来（考试概率极大）.

```
Connection cn = null;
PreparedStatement pstmt =null;
Resultset rs = null;
try
{
    Class.forname(driveClassName);
    cn =   DriverManager.getConnection(url,username,password);
    pstmt = cn.prepareStatement("select   score.* from score ,student " +
          "where score.stuId = student.id and student.name = ?");
```

```
        pstmt.setString(1,studentName);
        Resultset rs = pstmt.executeQuery();
        while(rs.next())
        {
              system.out.println(rs.getInt("subject")   +   "       " + rs.getFloat("score") );
        }
}catch(Exception e){e.printStackTrace();}
finally
{
        if(rs != null) try{ rs.close() }catch(exception e){}
        if(pstmt != null) try{pstmt.close()}catch(exception e){}
        if(cn != null) try{ cn.close() }catch(exception e){}
}
```

5、这段代码有什么不足之处?

```
try {
Connection conn = ...;
Statement stmt = ...;
ResultSet rs = stmt.executeQuery("select * from table1");
while(rs.next()) {
}
} catch(Exception ex) {
}
```

答：没有 finally 语句来关闭各个对象，另外，使用 finally 之后，要把变量的定义放在 try 语句块的外面，以便在 try 语句块之外的 finally 块中仍可以访问这些变量。

36、说出数据连接池的工作机制是什么?

J2EE 服务器启动时会建立一定数量的池连接，并一直维持不少于此数目的池连接。客户端程序需要连接时，池驱动程序会返回一个未使用的池连接并将其表记为忙。如果当前没有空闲连接，池驱动程序就新建一定数量的连接，新建连接的数量有配置参数决定。当使用的池连接调用完成后，池驱动程序将此连接表记为空闲，其他调用就可以使用这个连接。

实现方式，返回的 Connection 是原始 Connection 的代理，代理 Connection 的 close 方法不是真正关连接，而是把它代理的 Connection 对象还回到连接池中。

4、为什么要用 ORM? 和 JDBC 有何不一样?

orm 是一种思想，就是把 object 转变成数据库中的记录，或者把数据库中的记录转变成 objecdt，我们可以用 jdbc 来实现这种思想，其实，如果我们的项目是严格按照 oop 方式编写的话，我们的 jdbc 程序不管是有意还是无意，就已经在实现 orm 的工作了。

现在有许多 orm 工具，它们底层调用 jdbc 来实现了 orm 工作，我们直接使用这些工具，就省去了直接使用 jdbc 的繁琐细节，提高了开发效率，现在用的较多的 orm 工具是 hibernate。也听说一些其他 orm 工具，如 toplink,ojb 等。

8. XML 部分

1、xml 有哪些解析技术?区别是什么?

答:有 DOM,SAX,STAX 等
DOM:处理大型文件时其性能下降的非常厉害。这个问题是由 DOM 的树结构所造成的，这种结构占用的内存较多，而且 DOM 必须在解析文件之前把整个文档装入内存,适合对 XML 的随机访问 SAX:不现于 DOM,SAX 是事件驱动型的 XML 解析方式。它顺序读取 XML 文件，不需要一次全部装载整个文件。当遇到像文件开头，文档结束，或者标签开头与标签结束时，它会触发一个事件，用户通过在其回调事件中写入处理代码来处理 XML 文件，适合对 XML 的顺序访问
STAX:Streaming API for XML (StAX)
讲解这些区别是不需要特别去比较，就像说传智播客与其他培训机构的区别时，我们只需说清楚传智播客有什么特点和优点就行了，这就已经间接回答了彼此的区别。

2、你在项目中用到了 xml 技术的哪些方面?如何实现的?
答:用到了数据存贮，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收者，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 可以很方便的进行，软件的各种配置参数都存贮在 XML 文件中。
3、用 jdom 解析 xml 文件时如何解决中文问题?如何解析?
答:看如下代码,用编码方式加以解决

```
package test;
import java.io.*;
public class DOMTest
{
private String inFile = "c:\\people.xml"
private String outFile = "c:\\people.xml"
public static void main(String args[])
{
new DOMTest();
}
public DOMTest()
{
try
{
javax.xml.parsers.DocumentBuilder builder =
javax.xml.parsers.DocumentBuilderFactory.newInstance().newDocumentBuilder();
org.w3c.dom.Document doc = builder.newDocument();
org.w3c.dom.Element root = doc.createElement("老师");
org.w3c.dom.Element wang = doc.createElement("王");
org.w3c.dom.Element liu = doc.createElement("刘");
wang.appendChild(doc.createTextNode("我是王老师"));
root.appendChild(wang);
doc.appendChild(root);
javax.xml.transform.Transformer transformer =
javax.xml.transform.TransformerFactory.newInstance().newTransformer();
transformer.setOutputProperty(javax.xml.transform.OutputKeys.ENCODING, "gb2312");
```

```
transformer.setOutputProperty(javax.xml.transform.OutputKeys.INDENT, "yes");
transformer.transform(new javax.xml.transform.dom.DOMSource(doc),
new
javax.xml.transform.stream.StreamResult(outFile));
}
catch (Exception e)
{
System.out.println (e.getMessage());
}
}
}
```

4、编程用 JAVA 解析 XML 的方式.

答:用 SAX 方式解析 XML，XML 文件如下：

```
<?xml version=1.0 encoding=gb2312?>
<person>
<name>王小明</name>
<college>信息学院</college>
<telephone>6258113</telephone>
<notes>男,1955 年生,博士，95 年调入海南大学</notes>
</person>
```

事件回调类 SAXHandler.java

```
import java.io.*;
import java.util.Hashtable;
import org.xml.sax.*;
public class SAXHandler extends HandlerBase
{
private Hashtable table = new Hashtable();
private String currentElement = null;
private String currentValue = null;
public void setTable(Hashtable table)
{
this.table = table;
}
public Hashtable getTable()
{
return table;
}
public void startElement(String tag, AttributeList attrs)
throws SAXException
{
currentElement = tag;
}
public void characters(char[] ch, int start, int length)
throws SAXException
```

```
{
currentValue = new String(ch, start, length);
}
public void endElement(String name) throws SAXException
{
if (currentElement.equals(name))
table.put(currentElement, currentValue);
}

}
```

JSP 内容显示源码,SaxXml.jsp:

```
<HTML>
<HEAD>
<TITLE>剖析 XML 文件 people.xml</TITLE>
</HEAD>
<BODY>
<%@ page errorPage=ErrPage.jsp
contentType=text/html;charset=GB2312 %>
<%@ page import=java.io.* %>
<%@ page import=java.util.Hashtable %>
<%@ page import=org.w3c.dom.* %>
<%@ page import=org.xml.sax.* %>
<%@ page import=javax.xml.parsers.SAXParserFactory %>
<%@ page import=javax.xml.parsers.SAXParser %>
<%@ page import=SAXHandler %>
<%
File file = new File(c:\people.xml);
FileReader reader = new FileReader(file);
Parser parser;
SAXParserFactory spf = SAXParserFactory.newInstance();
SAXParser sp = spf.newSAXParser();
SAXHandler handler = new SAXHandler();
sp.parse(new InputSource(reader), handler);
Hashtable hashTable = handler.getTable();
out.println(<TABLE BORDER=2><CAPTION>教师信息表</CAPTION>);
out.println(<TR><TD>姓名</TD> + <TD> +
(String)hashTable.get(new String(name)) + </TD></TR>);
out.println(<TR><TD>学院</TD> + <TD> +
(String)hashTable.get(new String(college))+</TD></TR>);
out.println(<TR><TD>电话</TD> + <TD> +
(String)hashTable.get(new String(telephone)) + </TD></TR>);
out.println(<TR><TD>备注</TD> + <TD> +
(String)hashTable.get(new String(notes)) + </TD></TR>);
out.println(</TABLE>);
```

%>
</BODY>
</HTML>
70、XML 文档定义有几种形式？它们之间有何本质区别？解析 XML 文档有哪几种方式？
a: 两种形式 dtd　schema，b: 本质区别:schema 本身是 xml 的，可以被 XML 解析器解析(这也是从 DTD 上发展 schema 的根本目的)，c:有 DOM,SAX,STAX 等

DOM:处理大型文件时其性能下降的非常厉害。这个问题是由 DOM 的树结构所造成的，这种结构占用的内存较多，而且 DOM 必须在解析文件之前把整个文档装入内存,适合对 XML 的随机访问

SAX:不现于 DOM,SAX 是事件驱动型的 XML 解析方式。它顺序读取 XML 文件，不需要一次全部装载整个文件。当遇到像文件开头，文档结束，或者标签开头与标签结束时，它会触发一个事件，用户通过在其回调事件中写入处理代码来处理 XML 文件，适合对 XML 的顺序访问

STAX:Streaming API for XML (StAX)

9. 流行的框架与新技术

1、谈谈你对 Struts 的理解。
答:
1. struts 是一个按 MVC 模式设计的 Web 层框架，其实它就是一个大大的 servlet，这个 Servlet 名为 ActionServlet，或是 ActionServlet 的子类。我们可以在 web.xml 文件中将符合某种特征的所有请求交给这个 Servlet 处理，这个 Servlet 再参照一个配置文件（通常为/WEB-INF/struts-config.xml）将各个请求分别分配给不同的 action 去处理。
一个扩展知识点：struts 的配置文件可以有多个，可以按模块配置各自的配置文件，这样可以防止配置文件的过度膨胀；
2. ActionServlet 把请求交给 action 去处理之前，会将请求参数封装成一个 formbean 对象（就是一个 java 类，这个类中的每个属性对应一个请求参数），封装成一个什么样的 formbean 对象呢？看配置文件。
3.要说明的是， ActionServlet 把 formbean 对象传递给 action 的 execute 方法之前，可能会调用 formbean 的 validate 方法进行校验，只有校验通过后才将这个 formbean 对象传递给 action 的 execute 方法，否则，它将返回一个错误页面，这个错误页面由 input 属性指定，（看配置文件）作者为什么将这里命名为 input 属性，而不是 error 属性，我们后面结合实际的运行效果进行分析。
4.action 执行完后要返回显示的结果视图，这个结果视图是用一个 ActionForward 对象来表示的，actionforward 对象通过 struts-config.xml 配置文件中的配置关联到某个 jsp 页面，因为程序中使用的是在 struts-config.xml 配置文件为 jsp 页面设置的逻辑名，这样可以实现 action 程序代码与返回的 jsp 页面名称的解耦。

你对 struts 可能还有自己的应用方面的经验，那也要一并说出来。
2、谈谈你对 Hibernate 的理解。
答:
1. 面向对象设计的软件内部运行过程可以理解成就是在不断创建各种新对象、建立对象之间的关系，调用对象的方法来改变各个对象的状态和对象消亡的过程，不管程序运行的过程和操作怎么样，本质上都是要得到一个结果，程序上一个时刻和下一个时刻的运行结果的差

异就表现在内存中的对象状态发生了变化。
2.为了在关机和内存空间不够的状况下，保持程序的运行状态，需要将内存中的对象状态保存到持久化设备和从持久化设备中恢复出对象的状态，通常都是保存到关系数据库来保存大量对象信息。从 Java 程序的运行功能上来讲，保存对象状态的功能相比系统运行的其他功能来说，应该是一个很不起眼的附属功能，java 采用 jdbc 来实现这个功能，这个不起眼的功能却要编写大量的代码，而做的事情仅仅是保存对象和恢复对象，并且那些大量的 jdbc 代码并没有什么技术含量，基本上是采用一套例行公事的标准代码模板来编写，是一种苦活和重复性的工作。
3.通过数据库保存 java 程序运行时产生的对象和恢复对象，其实就是实现了 java 对象与关系数据库记录的映射关系，称为 ORM（即 Object Relation Mapping），人们可以通过封装 JDBC 代码来实现了这种功能，封装出来的产品称之为 ORM 框架，Hibernate 就是其中的一种流行 ORM 框架。使用 Hibernate 框架，不用写 JDBC 代码，仅仅是调用一个 save 方法，就可以将对象保存到关系数据库中，仅仅是调用一个 get 方法，就可以从数据库中加载出一个对象。
4.使用 Hibernate 的基本流程是：配置 Configuration 对象、产生 SessionFactory、创建 session 对象，启动事务，完成 CRUD 操作，提交事务，关闭 session。
5.使用 Hibernate 时，先要配置 hibernate.cfg.xml 文件，其中配置数据库连接信息和方言等，还要为每个实体配置相应的 hbm.xml 文件，hibernate.cfg.xml 文件中需要登记每个 hbm.xml 文件。
6.在应用 Hibernate 时，重点要了解 Session 的缓存原理，级联，延迟加载和 hql 查询。
3、AOP 的作用。
3、你对 Spring 的理解。
1.Spring 实现了工厂模式的工厂类（在这里有必要解释清楚什么是工厂模式），这个类名为 BeanFactory（实际上是一个接口），在程序中通常 BeanFactory 的子类 ApplicationContext。Spring 相当于一个大的工厂类，在其配置文件中通过<bean>元素配置用于创建实例对象的类名和实例对象的属性。
2. Spring 提供了对 IOC 良好支持，IOC 是一种编程思想，是一种架构艺术，利用这种思想可以很好地实现模块之间的解耦。IOC 也称为 DI（Depency Injection），什么叫依赖注入呢？譬如，Class Programmer

```
{
    Computer computer = null;
    public void code()
    {
        //Computer computer = new IBMComputer();
        //Computer computer = beanfacotry.getComputer();
        computer.write();
    }
    public void setComputer(Computer computer)
    {
        this.computer = computer;
    }
}
```

另外两种方式都由依赖，第一个直接依赖于目标类，第二个把依赖转移到工厂上，第三个彻底与目标和工厂解耦了。在 spring 的配置文件中配置片段如下：

```
<bean id=”computer” class=”cn.itcast.interview.Computer”>
```

</bean>

<bean id=”programmer” class=”cn.itcast.interview.Programmer”>
<property name=”computer” ref=”computer”></property>
</bean>

3. Spring 提供了对 AOP 技术的良好封装， AOP 称为面向切面编程，就是系统中有很多各不相干的类的方法，在这些众多方法中要加入某种系统功能的代码，例如，加入日志，加入权限判断，加入异常处理，这种应用称为 AOP。实现 AOP 功能采用的是代理技术，客户端程序不再调用目标，而调用代理类，代理类与目标类对外具有相同的方法声明，有两种方式可以实现相同的方法声明，一是实现相同的接口，二是作为目标的子类在，JDK 中采用 Proxy 类产生动态代理的方式为某个接口生成实现类，如果要为某个类生成子类，则可以用 CGLIB。在生成的代理类的方法中加入系统功能和调用目标类的相应方法，系统功能的代理 以 Advice 对象进行提供，显然要创建出代理对象，至少需要目标类和 Advice 类。spring 提供了这种支持，只需要在 spring 配置文件中配置这两个元素即可实现代理和 aop 功能，例如，

<bean id=”proxy” type=”org.spring.framework.aop.ProxyBeanFactory”>
<property name=”target” ref=””></property>
<property name=”advisor” ref=””></property>

</bean>

11、谈谈 Struts 中的 Action servlet。

12、Struts 优缺点

优点：

1. 实现 MVC 模式，结构清晰,使开发者只关注业务逻辑的实现.
2．有丰富的 tag 可以用 ,Struts 的标记库(Taglib)，如能灵活动用，则能大大提高开发效率
3. 页面导航

使系统的脉络更加清晰。通过一个配置文件，即可把握整个系统各部分之间的联系，这对于后期的维护有着莫大的好处。尤其是当另一批开发者接手这个项目时，这种优势体现得更加明显。

4. 提供 Exception 处理机制 .
5. 数据库链接池管理
6. 支持 I18N

缺点

一、 转到展示层时，需要配置 forward，如果有十个展示层的 jsp，需要配置十次 struts，而且还不包括有时候目录、文件变更，需要重新修改 forward，注意，每次修改配置之后，要求重新部署整个项目，而 tomcate 这样的服务器，还必须重新启动服务器

二、 二、 Struts 的 Action 必需是 thread－safe 方式，它仅仅允许一个实例去处理所有的请求。所以 action 用到的所有的资源都必需统一同步，这个就引起了线程安全的问题。

三、 测试不方便. Struts 的每个 Action 都同 Web 层耦合在一起，这样它的测试依赖于 Web 容器，单元测试也很难实现。不过有一个 Junit 的扩展工具 Struts TestCase 可以实现它的单元测试。

四、 类型的转换. Struts 的 FormBean 把所有的数据都作为 String 类型，它可以使用工具 Commons-Beanutils 进行类型转化。但它的转化都是在 Class 级别，而且转化的类型是不

可配置的。类型转化时的错误信息返回给用户也是非常困难的。
五、 对 Servlet 的依赖性过强. Struts 处理 Action 时必需要依赖 ServletRequest 和 ServletResponse，所有它摆脱不了 Servlet 容器。
六、 前端表达式语言方面.Struts 集成了 JSTL，所以它主要使用 JSTL 的表达式语言来获取数据。可是 JSTL 的表达式语言在 Collection 和索引属性方面处理显得很弱。
七、 对 Action 执行的控制困难. Struts 创建一个 Action，如果想控制它的执行顺序将会非常困难。甚至你要重新去写 Servlet 来实现你的这个功能需求。
八、 对 Action 执行前和后的处理. Struts 处理 Action 的时候是基于 class 的 hierarchies，很难在 action 处理前和后进行操作。
九、 对事件支持不够. 在 struts 中，实际是一个表单 Form 对应一个 Action 类(或 DispatchAction)，换一句话说：在 Struts 中实际是一个表单只能 对应一个事件，struts 这种事件方式称为 application event，application event 和 component event 相比是一种粗粒度的事件

119、STRUTS 的应用(如 STRUTS 架构)
Struts 是采用 Java Servlet/JavaServer Pages 技术，开发 Web 应用程序的开放源码的 framework。采用 Struts 能开发出基于 MVC(Model-View-Controller)设计模式的应用构架。 Struts 有如下的主要功能： 一.包含一个 controller servlet，能将用户的请求发送到相应的 Action 对象。二.JSP 自由 tag 库，并且在 controller servlet 中提供关联支持，帮助开发员创建交互式表单应用。 三.提供了一系列实用对象：XML 处理、通过 Java reflection APIs 自动处理 JavaBeans 属性、国际化的提示和消息。

110、说说 struts1 与 struts2 的区别。
1.都是 MVC 的 WEB 框架,
2 struts1 的老牌框架，应用很广泛，有很好的群众基础，使用它开发风险很小，成本更低！struts2 虽然基于这个框架，但是应用群众并多，相对不成熟，未知的风险和变化很多，开发人员相对不好招，使用它开发项目的风险系数更大，用人成本更高！
3.struts2 毕竟是站在前辈的基础设计出来，它会改善和完善 struts1 中的一些缺陷，struts1 中一些悬而未决问题在 struts2 得到了解决。
4.struts1 的前端控制器是一个 Servlet，名称为 ActionServlet，struts2 的前端控制器是一个 filter，在 struts2.0 中叫 FilterDispatcher，在 struts2.1 中叫 StrutsPrepareAndExecuteFilter。
5.struts1 的 action 需要继承 Action 类，struts2 的 action 可以不继承任何类；struts1 对同一个路径的所有请求共享一个 Action 实例，struts2 对同一个路径的每个请求分别使用一个独立 Action 实例对象，所有对于 struts2 的 Action 不用考虑线程安全问题。
6.在 struts1 中使用 formbean 封装请求参数，在 struts2 中直接使用 action 的属性来封装请求参数。
7.struts1 中的多个业务方法放在一个 Action 中时（即继承 DispatchAction 时），要么都校验，要么都不校验；对于 struts2，可以指定只对某个方法进行校验，当一个 Action 继承了 ActionSupport 且在这个类中只编写了 validateXxx()方法，那么则只对 Xxx()方法进行校验。

（一个请求来了的执行流程进行分析，struts2 是自动支持分模块开发，并可以不同模块设置不同的 url 前缀，这是通过 package 的 namespace 来实现的；struts2 是支持多种类型的视图，struts2 的视图地址可以是动态的，即视图的名称是支持变量方式的，举例，论坛发帖失败后

回来还要传递 boardid。视图内容显示方面：它的标签用 ognl，要 el 强大很多，在国际化方面支持分模块管理，两个模块用到同样的 key，对应不同的消息；）

与 Struts1 不同，Struts2 对用户的每一次请求都会创建一个 Action，所以 Struts2 中的 Action 是线程安全的。

给我印象最深刻的是：struts 配置文件中的 redirect 视图的 url 不能接受参数，而 struts2 配置文件中的 redirect 视图可以接受参数。

110、hibernate 中的 update()和 saveOrUpdate()的区别，session 的 load()和 get()的区别。
110、简述 Hibernate 和 JDBC 的优缺点？如何书写一个 one to many 配置文件.
7、iBatis 与 Hibernate 有什么不同?
相同点：屏蔽 jdbc api 的底层访问细节，使用我们不用与 jdbc api 打交道，就可以访问数据。jdbc api 编程流程固定，还将 sql 语句与 java 代码混杂在了一起，经常需要拼凑 sql 语句，细节很繁琐。
ibatis 的好处：屏蔽 jdbc api 的底层访问细节；将 sql 语句与 java 代码进行分离;提供了将结果集自动封装称为实体对象和对象的集合的功能， queryForList 返回对象集合，用 queryForObject 返回单个对象；提供了自动将实体对象的属性传递给 sql 语句的参数。

Hibernate 是一个全自动的 orm 映射工具，它可以自动生成 sql 语句,ibatis 需要我们自己在 xml 配置文件中写 sql 语句，hibernate 要比 ibatis 功能负责和强大很多。因为 hibernate 自动生成 sql 语句，我们无法控制该语句，我们就无法去写特定的高效率的 sql。对于一些不太复杂的 sql 查询，hibernate 可以很好帮我们完成，但是，对于特别复杂的查询，hibernate 就很难适应了，这时候用 ibatis 就是不错的选择，因为 ibatis 还是由我们自己写 sql 语句。

7、写 Hibernate 的一对多和多对一双向关联的 orm 配置?
9、hibernate 的 inverse 属性的作用?
解决方案一，按照 Object[]数据取出数据，然后自己组 bean
解决方案二，对每个表的 bean 写构造函数，比如表一要查出 field1,field2 两个字段，那么有一个构造函数就是 Bean(type1 filed1,type2
field2) ，然后在 hql 里面就可以直接生成这个 bean 了。

10、在 DAO 中如何体现 DAO 设计模式?
解决方案一，按照 Object[]数据取出数据，然后自己组 bean
解决方案二，对每个表的 bean 写构造函数，比如表一要查出 field1,field2 两个字段，那么有一个构造函数就是 Bean(type1 filed1,type2
field2) ，然后在 hql 里面就可以直接生成这个 bean 了。

10、spring+Hibernate 中委托方案怎么配置?
解决方案一，按照 Object[]数据取出数据，然后自己组 bean
解决方案二，对每个表的 bean 写构造函数，比如表一要查出 field1,field2 两个字段，那么有

一个构造函数就是 Bean(type1 filed1,type2
field2) ，然后在 hql 里面就可以直接生成这个 bean 了。

10、spring+Hibernate 中委托方案怎么配置?
解决方案一，按照 Object[]数据取出数据，然后自己组 bean
解决方案二，对每个表的 bean 写构造函数，比如表一要查出 field1,field2 两个字段，那么有一个构造函数就是 Bean(type1 filed1,type2
field2) ，然后在 hql 里面就可以直接生成这个 bean 了。

8. hibernate 进行多表查询每个表中各取几个字段，也就是说查询出来的结果集没有一个实体类与之对应如何解决；

解决方案一，按照 Object[]数据取出数据，然后自己组 bean
解决方案二，对每个表的 bean 写构造函数，比如表一要查出 field1,field2 两个字段，那么有一个构造函数就是 Bean(type1 filed1,type2
field2) ，然后在 hql 里面就可以直接生成这个 bean 了。
8.介绍一下 Hibernate 的二级缓存
按照以下思路来回答：（1）首先说清楚什么是缓存，（2）再说有了 hibernate 的 Session 就是一级缓存，即有了一级缓存，为什么还要有二级缓存，（3）最后再说如何配置 Hibernate 的二级缓存。
（1)缓存就是把以前从数据库中查询出来和使用过的对象保存在内存中(一个数据结构中)，这个数据结构通常是或类似 Hashmap，当以后要使用某个对象时，先查询缓存中是否有这个对象，如果有则使用缓存中的对象，如果没有则去查询数据库，并将查询出来的对象保存在缓存中，以便下次使用。下面是缓存的伪代码：
引出 hibernate 的第二级缓存，用下面的伪代码分析了 Cache 的实现原理

```
Dao
{
    hashmap map = new map();
    User getUser(integer id)
    {
        User user = map.get(id)
        if(user == null)
        {
            user = session.get(id);
            map.put(id,user);
        }
        return user;
    }
}

Dao
{
    Cache cache = null
    setCache(Cache cache)
```

```
    {
        this.cache = cache
    }

    User getUser(int id)
    {
        if(cache!=null)
        {
            User user = cache.get(id);
            if(user ==null)
            {
                user = session.get(id);
                cache.put(id,user);
            }
            return user;
        }

        return session.get(id);
    }
}
```

（2）Hibernate 的 Session 就是一种缓存，我们通常将之称为 Hibernate 的一级缓存，当想使用 session 从数据库中查询出一个对象时，Session 也是先从自己内部查看是否存在这个对象，存在则直接返回，不存在才去访问数据库，并将查询的结果保存在自己内部。由于 Session 代表一次会话过程，一个 Session 与一个数据库连接相关连，所以 Session 最好不要长时间保持打开，通常仅用于一个事务当中，在事务结束时就应关闭。并且 Session 是线程不安全的，被多个线程共享时容易出现问题。通常只有那种全局意义上的缓存才是真正的缓存应用，才有较大的缓存价值，因此，Hibernate 的 Session 这一级缓存的缓存作用并不明显，应用价值不大。Hibernate 的二级缓存就是要为 Hibernate 配置一种全局缓存，让多个线程和多个事务都可以共享这个缓存。我们希望的是一个人使用过，其他人也可以使用， session 没有这种效果。

（3）二级缓存是独立于 Hibernate 的软件部件，属于第三方的产品，多个厂商和组织都提供有缓存产品，例如，EHCache 和 OSCache 等等。在 Hibernate 中使用二级缓存，首先就要在 hibernate.cfg.xml 配置文件中配置使用哪个厂家的缓存产品，接着需要配置该缓存产品自己的配置文件，最后要配置 Hibernate 中的哪些实体对象要纳入到二级缓存的管理中。明白了二级缓存原理和有了这个思路后，很容易配置起 Hibernate 的二级缓存。扩展知识：一个 SessionFactory 可以关联一个二级缓存，也即一个二级缓存只能负责缓存一个数据库中的数据，当使用 Hibernate 的二级缓存后，注意不要有其他的应用或 SessionFactory 来更改当前数据库中的数据，这样缓存的数据就会与数据库中的实际数据不一致。

111、Spring 的依赖注入是什么意思? 给一个 Bean 的 message 属性, 字符串类型, 注入值为 "Hello" 的 XML 配置文件该怎么写?

120、Jdo 是什么?

JDO 是 Java 对象持久化的新的规范，为 java data object 的简称,也是一个用于存取某种数据

仓库中的对象的标准化 API。JDO 提供了透明的对象存储，因此对开发人员来说，存储数据对象完全不需要额外的代码（如 JDBC API 的使用）。这些繁琐的例行工作已经转移到 JDO 产品提供商身上，使开发人员解脱出来，从而集中时间和精力在业务逻辑上。另外，JDO 很灵活，因为它可以在任何数据底层上运行。JDBC 只是面向关系数据库（RDBMS）JDO 更通用，提供到任何数据底层的存储功能，比如关系数据库、文件、XML 以及对象数据库（ODBMS）等等，使得应用可移植性更强。

什么是 spring 的 IOC　AOP
STRUTS 的工作流程！
spring 与 EJB 的区别！！
Hibernate 工作原理及为什么要用？
原理：
1.读取并解析配置文件
2.读取并解析映射信息，创建 SessionFactory
3.打开 Sesssion
4.创建事务 Transation
5.持久化操作
6.提交事务
7.关闭 Session
8.关闭 SesstionFactory

为什么要用：
1. 对 JDBC 访问数据库的代码做了封装，大大简化了数据访问层繁琐的重复性代码。

2. Hibernate 是一个基于 JDBC 的主流持久化框架，是一个优秀的 ORM 实现。他很大程度的简化 DAO 层的编码工作

3. hibernate 使用 Java 反射机制，而不是字节码增强程序来实现透明性。

4. hibernate 的性能非常好，因为它是个轻量级框架。映射的灵活性很出色。它支持各种关系数据库，从一对一到多对多的各种复杂关系。

2． Hibernate 是如何延迟加载?
1. Hibernate2 延迟加载实现：a)实体对象 b)集合（Collection）

2. Hibernate3 提供了属性的延迟加载功能

当 Hibernate 在查询数据的时候，数据并没有存在与内存中，当程序真正对数据的操作时，对象才存在与内存中，就实现了延迟加载，他节省了服务器的内存开销，从而提高了服务器的性能。

3．Hibernate 中怎样实现类之间的关系?(如：一对多、多对多的关系)

类与类之间的关系主要体现在表与表之间的关系进行操作，它们都市对对象进行操作，我们

程序中把所有的表与类都映射在一起，它们通过配置文件中的 many-to-one、one-to-many、many-to-many、

4． 说下 Hibernate 的缓存机制

1. 内部缓存存在 Hibernate 中又叫一级缓存，属于应用事物级缓存

2. 二级缓存：
a) 应用及缓存
b) 分布式缓存
条件：数据不会被第三方修改、数据大小在可接受范围、数据更新频率低、同一数据被系统频繁使用、非 关键数据
c) 第三方缓存的实现

5． Hibernate 的查询方式
Sql、Criteria,object comptosition
Hql：
1、 属性查询
2、 参数查询、命名参数查询
3、 关联查询
4、 分页查询
5、 统计函数

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

7． Struts 工作机制？为什么要使用 Struts？
工作机制：
Struts 的工作流程:
在 web 应用启动时就会加载初始化 ActionServlet,ActionServlet 从
struts-config.xml 文件中读取配置信息,把它们存放到各种配置对象
当 ActionServlet 接收到一个客户请求时,将执行如下流程.
-(1)检索和用户请求匹配的 ActionMapping 实例,如果不存在,就返回请求路径无效信息;
-(2)如果 ActionForm 实例不存在,就创建一个 ActionForm 对象,把客户提交的表单数据保存到 ActionForm 对象中;
-(3)根据配置信息决定是否需要表单验证.如果需要验证,就调用 ActionForm 的 validate()方法;
-(4)如果 ActionForm 的 validate()方法返回 null 或返回一个不包含 ActionMessage 的

ActuibErrors 对象, 就表示表单验证成功;
-(5)ActionServlet 根据 ActionMapping 所包含的映射信息决定将请求转发给哪个 Action,如果相应的 Action 实例不存在,就先创建这个实例,然后调用 Action 的 execute()方法;
-(6)Action 的 execute()方法返回一个 ActionForward 对象,ActionServlet 在把客户请求转发给 ActionForward 对象指向的 JSP 组件;
-(7)ActionForward 对象指向 JSP 组件生成动态网页,返回给客户;

为什么要用：
JSP、Servlet、JavaBean 技术的出现给我们构建强大的企业应用系统提供了可能。但用这些技术构建的系统非常的繁乱，所以在此之上，我们需要一个规则、一个把这些技术组织起来的规则，这就是框架，Struts 便应运而生。

基于 Struts 开发的应用由 3 类组件构成：控制器组件、模型组件、视图组件

8． Struts 的 validate 框架是如何验证的？
在 struts 配置文件中配置具体的错误提示，再在 FormBean 中的 validate()方法具体调用。

9． 说下 Struts 的设计模式
MVC 模式: web 应用程序启动时就会加载并初始化 ActionServler。用户提交表单时，一个配置好的 ActionForm 对象被创建，并被填入表单相应的数 据，ActionServler 根据 Struts-config.xml 文件配置好的设置决定是否需要表单验证，如果需要就调用 ActionForm 的 Validate（）验证后选择将请求发送到哪个 Action，如果 Action 不存在，ActionServlet 会先创建这个对象，然后调用 Action 的 execute（）方法。Execute（）从 ActionForm 对象中获取数据，完成业务逻辑，返回一个 ActionForward 对 象，ActionServlet 再把客户请求转发给 ActionForward 对象指定的 jsp 组件，ActionForward 对象指定的 jsp 生成动 态的网页，返回给客户。

10． spring 工作机制及为什么要用?
1.spring mvc 请所有的请求都提交给 DispatcherServlet,它会委托应用系统的其他模块负责负责对请求进行真正的处理工作。
2.DispatcherServlet 查询一个或多个 HandlerMapping,找到处理请求的 Controller.
3.DispatcherServlet 请请求提交到目标 Controller
4.Controller 进行业务逻辑处理后，会返回一个 ModelAndView
5.Dispathcher 查询一个或多个 ViewResolver 视图解析器,找到 ModelAndView 对象指定的视图对象
6.视图对象负责渲染返回给客户端。

为什么用：
{AOP 让开发人员可以创建非行为性的关注点，称为横切关注点，并将它们插入到应用程序代码中。使用 AOP 后，公共服务 （比如日志、持久性、事务等）就可以分解成方面并应用到域对象上，同时不会增加域对象的对象模型的复杂性。
IOC 允许创建一个可以构造对象的应用环境，然后向这些对象传递它们的协作对象。正如单词 倒置 所表明的，IOC 就像反 过来的 JNDI。没有使用一堆抽象工厂、服务定位器、单元素（singleton）和直接构造（straight construction），每一个对象都是用其协作对象构造

的。因此是由容器管理协作对象（collaborator）。
Spring 即使一个 AOP 框架，也是一 IOC 容器。 Spring 最好的地方是它有助于您替换对象。有了 Spring，只要用 JavaBean 属性和配置文件加入依赖性（协作对象）。然后可以很容易地在需要时替换具有类似接口的协作对象。}

Spring 框架是一个分层架构，由 7 个定义良好的模块组成。Spring 模块构建在核心容器之上，核心容器定义了创建、配置和管理 bean 的方式，如图 1 所示。

组成 Spring 框架的每个模块（或组件）都可以单独存在，或者与其他一个或多个模块联合实现。每个模块的功能如下：

☆ 核心容器：核心容器提供 Spring 框架的基本功能。核心容器的主要组件是 BeanFactory，它是工厂模式的实现。BeanFactory 使用控制反转 （IOC）模式将应用程序的配置和依赖性规范与实际的应用程序代码分开。

☆ Spring 上下文：Spring 上下文是一个配置文件，向 Spring 框架提供上下文信息。Spring 上下文包括企业服务，例如 JNDI、EJB、电子邮件、国际化、校验和调度功能。

☆ Spring AOP：通过配置管理特性，Spring AOP 模块直接将面向方面的编程功能集成到了 Spring 框架中。所以，可以很容易地使 Spring 框架管理的任何对象支持 AOP。Spring AOP 模块为基于 Spring 的应用程序中的对象提供了事务管理服务。通过使用 Spring AOP，不用依赖 EJB 组件，就可以将声明性事务管理集成到应用程序中。

☆ Spring DAO：JDBC DAO 抽象层提供了有意义的异常层次结构，可用该结构来管理异常处理和不同数据库供应商抛出的错误消息。异常层次结构简化了错误处理，并且极大地降低了需要编写 的异常代码数量（例如打开和关闭连接）。Spring DAO 的面向 JDBC 的异常遵从通用的 DAO 异常层次结构。

☆ Spring ORM：Spring 框架插入了若干个 ORM 框架，从而提供了 ORM 的对象关系工具，其中包括 JDO、Hibernate 和 iBatis SQL Map。所有这些都遵从 Spring 的通用事务和 DAO 异常层次结构。

☆ Spring Web 模块：Web 上下文模块建立在应用程序上下文模块之上，为基于 Web 的应用程序提供了上下文。所以，Spring 框架支持与 Jakarta Struts 的集成。Web 模块还简化了处理多部分请求以及将请求参数绑定到域对象的工作。

☆ Spring MVC 框架：MVC 框架是一个全功能的构建 Web 应用程序的 MVC 实现。通过策略接口，MVC 框架变成为高度可配置的，MVC 容纳了大量视图技术，其中包括 JSP、Velocity、Tiles、iText 和 POI。

Spring 框架的功能可以用在任何 J2EE 服务器中，大多数功能也适用于不受管理的环境。

Spring 的核心要点是：支持不绑定到特定 J2EE 服务的可重用业务和数据访问对象。毫无疑问，这样的对象可以在不同 J2EE 环境 （Web 或 EJB）、独立应用程序、测试环境之间重用。

IOC 和 AOP

控制反转模式（也称作依赖性介入）的基本概念是：不创建对象，但是描述创建它们的方式。在代码中不直接与对象和服务连接，但在配置文件中描述哪一个组件需要哪一项服务。容器（在 Spring 框架中是 IOC 容器） 负责将这些联系在一起。

在典型的 IOC 场景中，容器创建了所有对象，并设置必要的属性将它们连接在一起，决定什么时间调用方法。下表列出了 IOC 的一个实现模式。

Spring 框架的 IOC 容器采用类型 2 和类型 3 实现。

面向方面的编程

面向方面的编程，即 AOP，是一种编程技术，它允许程序员对横切关注点或横切典型的职责分界线的行为（例如日志和事务管理）进行模块化。AOP 的核心构造是方面，它将那些影响多个类的行为封装到可重用的模块中。

AOP 和 IOC 是补充性的技术，它们都运用模块化方式解决企业应用程序开发中的复杂问题。在典型的面向对象开发方式中，可能要将日志记录语句放在所有方法和 Java 类中才能实现日志功能。在 AOP 方式中，可以反过来将日志服务模块化，并以声明的方式将它们应用到需要日志的组件上。当然，优势就是 Java 类不需要知道日志服务的存在，也不需要考虑相关的代码。所以，用 Spring AOP 编写的应用程序代码是松散耦合的。

AOP 的功能完全集成到了 Spring 事务管理、日志和其他各种特性的上下文中。

IOC 容器

Spring 设计的核心是 org.springframework.beans 包，它的设计目标是与 JavaBean 组件一起使用。这个包通常不是由用户直接使用，而是由服务器将其用作其他多数功能的底层中介。下一个最高级抽象是 BeanFactory 接口，它是工厂设计模式的实现，允许通过名称创建和检索对象。BeanFactory 也可以管理对象之间的关系。

BeanFactory 支持两个对象模型。

□ 单态 模型提供了具有特定名称的对象的共享实例，可以在查询时对其进行检索。Singleton 是默认的也是最常用的对象模型。对于无状态服务对象很理想。

□ 原型 模型确保每次检索都会创建单独的对象。在每个用户都需要自己的对象时，原型模型最适合。

bean 工厂的概念是 Spring 作为 IOC 容器的基础。IOC 将处理事情的责任从应用程序代码转移到框架。正如我将在下一个示例中演示的那样，Spring 框架使用 JavaBean 属性和配置数据来指出必须设置的依赖关系。

BeanFactory 接口

因为 org.springframework.beans.factory.BeanFactory 是一个简单接口，所以可以针对各种底层存储方法实现。最常用的 BeanFactory 定义是 XmlBeanFactory，它根据 XML 文件中的定义装入 bean，如清单 1 所示。

清单 1. XmlBeanFactory

```
BeanFactory factory = new XMLBeanFactory(new FileInputSteam("mybean.xml"));
```

在 XML 文件中定义的 Bean 是被消极加载的，这意味在需要 bean 之前，bean 本身不会被初始化。要从 BeanFactory 检索 bean，只需调用 getBean() 方法，传入将要检索的 bean 的名称即可，如清单 2 所示。

清单 2. getBean()

```
MyBean mybean = (MyBean) factory.getBean("mybean");
```

每个 bean 的定义都可以是 POJO （用类名和 JavaBean 初始化属性定义） 或 FactoryBean。FactoryBean 接口为使用 Spring 框架构建的应用程序添加了一个间接的级别。

IOC 示例

理解控制反转最简单的方式就是看它的实际应用。在对由三部分组成的 Spring 系列 的第 1 部分进行总结时，我使用了一个示例，演示了如何通过 Spring IOC 容器注入应用程序的依赖关系（而不是将它们构建进来）。

我用开启在线信用帐户的用例作为起点。对于该实现，开启信用帐户要求用户与以下服务进行交互：

☆ 信用级别评定服务，查询用户的信用历史信息。

☆ 远程信息链接服务，插入客户信息，将客户信息与信用卡和银行信息连接起来，以进行自动借记（如果需要的话）。

☆ 电子邮件服务，向用户发送有关信用卡状态的电子邮件。

三个接口

对于这个示例，我假设服务已经存在，理想的情况是用松散耦合的方式把它们集成在一起。以下清单显示了三个服务的应用程序接口。

清单 3. CreditRatingInterface

```
public interface CreditRatingInterface {
public boolean getUserCreditHistoryInformation(ICustomer iCustomer);
}
```

清单 3 所示的信用级别评定接口提供了信用历史信息。它需要一个包含客户信息的 Customer 对象。该接口的实现是由 CreditRating 类提供的。

清单 4. CreditLinkingInterface

```
public interface CreditLinkingInterface {

public String getUrl();
public void setUrl(String url);
public void linkCreditBankAccount() throws Exception ;

}
```

信用链接接口将信用历史信息与银行信息（如果需要的话）连接在一起，并插入用户的信用卡信息。信用链接接口是一个远程服务，它的查询是通过 getUrl() 方法进行的。URL 由 Spring 框架的 bean 配置机制设置，我稍后会讨论它。该接口的实现是由 CreditLinking 类提供的。

清单 5. EmailInterface

```
public interface EmailInterface {

public void sendEmail(ICustomer iCustomer);
public String getFromEmail();
public void setFromEmail(String fromEmail) ;
public String getPassword();
public void setPassword(String password) ;
public String getSmtpHost() ;
public void setSmtpHost(String smtpHost);
public String getUserId() ;
public void setUserId(String userId);
```

10. 软件工程与设计模式

111、UML 方面

标准建模语言 UML。用例图,静态图(包括类图、对象图和包图),行为图,交互图(顺序图,合作图),实现图。

112. 软件开发的

92、j2ee 常用的设计模式？说明工厂模式。

总共 23 种，分为三大类：创建型，结构型，行为型

我只记得其中常用的 6、7 种，分别是：

创建型（工厂、工厂方法、抽象工厂、单例）

结构型（包装、适配器，组合，代理）

行为（观察者，模版，策略）

然后再针对你熟悉的模式谈谈你的理解即可。

Java 中的 23 种设计模式：

Factory（工厂模式），Builder（建造模式），Factory Method（工厂方法模式），Prototype（原始模型模式），Singleton（单例模式），Facade（门面模式），Adapter（适配器模式），Bridge（桥梁模式），Composite（合成模式），Decorator（装饰模式），Flyweight（享元模式），Proxy（代理模式），Command（命令模式），Interpreter（解释器模式），Visitor（访问者模式），Iterator（迭代子模式），Mediator（调停者模式），Memento（备忘录模式），Observer（观察者模式），State（状态模式），Strategy（策略模式），Template Method（模板方法模式），Chain Of Responsibleity（责任链模式）

工厂模式：工厂模式是一种经常被使用到的模式，根据工厂模式实现的类可以根据提供的数据生成一组类中某一个类的实例，通常这一组类有一个公共的抽象父类并且实现了相同的方法，但是这些方法针对不同的数据进行了不同的操作。首先需要定义一个基类，该类的子类通过不同的方法实现了基类中的方法。然后需要定义一个工厂类，工厂类可以根据条件生成不同的子类实例。当得到子类的实例后，开发人员可以调用基类中的方法而不必考虑到底返回的是哪一个子类的实例。

113、开发中都用到了那些设计模式?用在什么场合?

每个模式都描述了一个在我们的环境中不断出现的问题，然后描述了该问题的解决方案的核心。通过这种方式，你可以无数次地使用那些已有的解决方案，无需在重复相同的工作。主要用到了 MVC 的设计模式。用来开发 JSP/Servlet 或者 J2EE 的相关应用。简单工厂模式等。

11. j2ee 部分

117、BS 与 CS 的联系与区别。

C/S 是 Client/Server 的缩写。服务器通常采用高性能的 PC、工作站或小型机，并采用大型数据库系统，如 Oracle、Sybase、InFORMix 或 SQL Server。客户端需要安装专用的客户端软件。

B/S 是 Brower/Server 的缩写，客户机上只要安装一个浏览器（Browser），如 Netscape Navigator 或 Internet Explorer，服务器安装 Oracle、Sybase、InFORMix 或 SQL Server 等数据库。在这种结构下，用户界面完全通过 WWW 浏览器实现，一部分事务逻辑在前端实现，但是主要事务逻辑在服务器端实现。浏览器通过 Web Server 同数据库进行数据交互。

C/S 与 B/S 区别：

1．硬件环境不同:

C/S 一般建立在专用的网络上, 小范围里的网络环境, 局域网之间再通过专门服务器提供连接和数据交换服务.

B/S 建立在广域网之上的, 不必是专门的网络硬件环境,例与电话上网, 租用设备. 信息自己管理. 有比 C/S 更强的适应范围, 一般只要有操作系统和浏览器就行

２．对安全要求不同

C/S 一般面向相对固定的用户群, 对信息安全的控制能力很强. 一般高度机密的信息系统采用 C/S 结构适宜. 可以通过 B/S 发布部分可公开信息.

B/S 建立在广域网之上, 对安全的控制能力相对弱, 可能面向不可知的用户。

３．对程序架构不同

C/S 程序可以更加注重流程, 可以对权限多层次校验, 对系统运行速度可以较少考虑.

B/S 对安全以及访问速度的多重的考虑, 建立在需要更加优化的基础之上. 比 C/S 有更高的要求 B/S 结构的程序架构是发展的趋势, 从 MS 的.Net 系列的 BizTalk 2000 Exchange 2000 等, 全面支持网络的构件搭建的系统. SUN 和 IBM 推的 JavaBean 构件技术等,使 B/S 更加成熟.

４．软件重用不同

C/S 程序可以不可避免的整体性考虑, 构件的重用性不如在 B/S 要求下的构件的重用性好.

B/S 对的多重结构,要求构件相对独立的功能. 能够相对较好的重用.就入买来的餐桌可以再利用,而不是做在墙上的石头桌子

５．系统维护不同

C/S 程序由于整体性, 必须整体考察, 处理出现的问题以及系统升级. 升级难. 可能是再做一个全新的系统

B/S 构件组成,方面构件个别的更换,实现系统的无缝升级. 系统维护开销减到最小.用户从网上自己下载安装就可以实现升级.

６．处理问题不同

C/S 程序可以处理用户面固定, 并且在相同区域, 安全要求高需求, 与操作系统相关. 应该都是相同的系统

B/S 建立在广域网上, 面向不同的用户群, 分散地域, 这是 C/S 无法作到的. 与操作系统平台关系最小.

７．用户接口不同

C/S 多是建立的 Window 平台上,表现方法有限,对程序员普遍要求较高

B/S 建立在浏览器上, 有更加丰富和生动的表现方式与用户交流. 并且大部分难度减低, 减低开发成本.

８．信息流不同

C/S 程序一般是典型的中央集权的机械式处理, 交互性相对低

B/S 信息流向可变化, B-B B-C B-G 等信息、流向的变化, 更像交易中心。

2、应用服务器与 WEB SERVER 的区别？

应用服务器：Weblogic、Tomcat、Jboss

WEB SERVER：IIS、 Apache

32、应用服务器有那些？

BEA WebLogic Server，IBM WebSphere Application Server，Oracle9i Application Server，jBoss，Tomcat

3、J2EE 是什么？

答:Je22 是 Sun 公司提出的多层(multi-diered),分布式(distributed),基于组件(component-base)的企业级应用模型(enterpriese application model).在这样的一个应用系统中，可按照功能划分为不同的组件，这些组件又可在不同计算机上，并且处于相应的层次(tier)中。所属层次包括客户层(clietn tier)组件,web 层和组件,Business 层和组件,企业信息系统(EIS)层。

一个另类的回答：j2ee 就是增删改查。

67、J2EE 是技术还是平台还是框架？ 什么是 J2EE

J2EE 本身是一个标准，一个为企业分布式应用的开发提供的标准平台。

J2EE 也是一个框架，包括 JDBC、JNDI、RMI、JMS、EJB、JTA 等技术。

95、请对以下在 J2EE 中常用的名词进行解释(或简单描述)

web 容器：给处于其中的应用程序组件（JSP，SERVLET）提供一个环境，使 JSP,SERVLET 直接更容器中的环境变量接口交互，不必关注其它系统问题。主要有 WEB 服务器来实现。例如：TOMCAT,WEBLOGIC,WEBSPHERE 等。该容器提供的接口严格遵守 J2EE 规范中的 WEB APPLICATION 标准。我们把遵守以上标准的 WEB 服务器就叫做 J2EE 中的 WEB 容器。

EJB 容器：Enterprise java bean 容器。更具有行业领域特色。他提供给运行在其中的组件 EJB 各种管理功能。只要满足 J2EE 规范的 EJB 放入该容器，马上就会被容器进行高效率的管理。并且可以通过现成的接口来获得系统级别的服务。例如邮件服务、事务管理。

JNDI：（Java Naming & Directory Interface）JAVA 命名目录服务。主要提供的功能是：提供一个目录系统，让其它各地的应用程序在其上面留下自己的索引，从而满足快速查找和定位分布式应用程序的功能。

JMS：（Java Message Service）JAVA 消息服务。主要实现各个应用程序之间的通讯。包括点对点和广播。

JTA：（Java Transaction API）JAVA 事务服务。提供各种分布式事务服务。应用程序只需调用其提供的接口即可。

JAF：（Java Action FrameWork）JAVA 安全认证框架。提供一些安全控制方面的框架。让开发者通过各种部署和自定义实现自己的个性安全控制策略。

RMI/IIOP:（Remote Method Invocation /internet 对象请求中介协议）他们主要用于通过远程调用服务。例如，远程有一台计算机上运行一个程序，它提供股票分析服务，我们可以在本地计算机上实现对其直接调用。当然这是要通过一定的规范才能在异构的系统之间进行通信。RMI 是 JAVA 特有的。