哈尔滨工业大学

**计算机科学与技术学院/国家示范性软件学院**

**2021年春季学期**

**《软件架构与中间件》课程**

**实验报告**

**Lab 5：分布式系统整合实验**

|  |  |  |
| --- | --- | --- |
| **姓名** | **学号** | **联系方式** |
| 梅智敏 | 1183710118 | 1044388658@qq.com |
| 王国庆 | 1183710131 | 1113292969@qq.com |

目 录

[1 实验概述 1](#_Toc40516715)

[1.1 实验目的 1](#_Toc40516716)

[1.2 实验要求 1](#_Toc40516717)

[2 实验内容与过程 1](#_Toc40516718)

[3 结对开发过程记录 21](#_Toc40516719)

[4 实验总结 22](#_Toc40516720)

[5 教师评语 24](#_Toc40516721)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **学号：** | **1183710118** | **姓名：** | **梅智敏** | |
| **学号：** | **1183710131** | **姓名** | **王国庆** | |
| 实验概述 | | | |  |
| 实验目的  1. 尝试综合运用计算层、数据层和表示层的架构技术 2. 实现满足高性能、高可用、高可靠等质量属性的复杂软件系统 3. 学会分析和构建高效的分布式软件系统  实验要求  1. 2人结对成组 2. 基于实验2-4在不同层面（计算、数据、表示）上的软件架构技术，对遗留进销存系统进行分布式改造。 3. 按照用户需求，商业业务逻辑，将各部分的内容进行逐步整合，形成一个功能完整，且具有较高可用性、可靠性等的分布式软件系统。 4. 分析改造前后系统的非功能指标。 5. 应该出设计过程和实现细节 | | | | |
| 实验内容与过程 | | | | |
| * 1. **对遗留系统从三个层面进行全面的分布式改造的设计过程和实现细节。**  2.1.1计算层 在计算层使用Hadoop进行map-reduce编程，对“进销存分布式系统” 进行计算层架构重构，以支持海量用户的在线高并发请求场景。   * **配置集群环境**   首先下载jdk-11.0.11\_linux-x64\_bin.tar.gz，解压到/home/meizhimin/jdk-11.0.11。    修改/etc/profile文件，配置JDK信息    然后执行source /etc/profile使得配置文件生效    然后为meizhimin用户添加权限：    修改 /etc/hostname 文件： master节点的主机设置为 master ，其他两个虚拟机分别设  置为slave1、 slave2  并且修改/etc/hosts文件如下：    安装ssh    创间.ssh目录并赋予当前用户权限    生成RSA密钥对以及集群内共享密钥：      配置集群内的3个节点可以通过ssh无密码访问：      现在测试无密码ssh访问      成功！   * 配置Hadoop集群计算环境，数据块副本数量为3   首先下载hadoop3.2.2并解压到用户目录下    然后给hadoop配置环境变量：    使用source命令使得配置文件生效：    修改Hadoop安装目录/etc/hadoop/目录下的Hadoop-env.sh文件，添加：    修改core-site.xml中主要内容：    hdfs-site.xml中主要内容：value字段设为3，表示数据块副本数为3    mapred-site.xml中主要内容(MapReduce的配置文件)    配置三台主机的Hadoop文件(就是创建masters和workers文件并编辑)      向 slave1 和 slave2 节点复制 hadoop-3.2.2 整个目录至相同的位置，使用如下命令（将sam换成自己的用户名，hadoop版本也换成自己的即可）    在master节点上执行hdfs namenode -format    再执行start-all.sh验证正确性    使用jps查看    故成功开启！   * **使用Hadoop集群环境计算遗留的“进销存”系统中海量用户的log日志中的访问统计。**   进入hadoop的安装目录    将存储有用户登录信息的文件上传到hdfs文件系统后，调用wordcount函数来对输入的文件进行处理，便可以利用三个数据块的计算资源，从而大大提高了系统的计算高性能。具体效果见2.2.1节。 2.1.2数据层 **使用Mycat实现数据库分库分表**   * **基本开发环境准备**   安装JDK1.8并完成环境配置    安装MySQL8.0并完成环境配置    安装MyCat-1.6.7.6    在Mycat的安装目录的lib子目录下，下载MySQL8.0版本的JDBC驱动，并替换默认的MySQL5.0版本的驱动     * **Mycat系统参数配置**   编辑mycat/conf/server.xml，在system标签下添加启动端口配置，端口设置为8066：    设置登陆Mycat的用户名、密码、逻辑库：     * **Mycat逻辑库、表分片配置**   遗留的进销存系统中存在Table：t\_goods，存储全部的商品价格信息，各个字段如下：    由于进销存系统中涉及到许多种货品，故该表内容极为庞大，所以我们需要对该表进行分片以提高后续在该表上的sql语句执行效率。  编辑mycat/conf/schema/xml，将t\_goods进行分片：dn1和dn2，分片规则为rule1    rule1按照goods\_id属性来进行分片，规则为func1    接下来配置物理库分片映射，Mycat作为数据库代理需要逻辑库、逻辑用户、表切分后需要配置分片，分片也就需要映射到真实的物理主机上。     * **Mycat表切分规则配置**   编辑mycat/conf/rule.xml，此配置文件用于编写表切分规则，它决定了数据切分后性能的好坏，因此是最重要的配置。  name为schema.xml 中 table 标签中对应的 rule="rule1" ,也就是配置表的分片规则；columns 是表的切分字段；algorithm 是规则对应的切分规则：映射到 function 的 name    前面的algorithm所指定的切分规则会映射到function，function配置了分片规则。  name 为切分规则的名称，名字任意取，但是需要与tableRule 中匹配；class 是切分规则对应的切分类，写死，需要哪种规则则配置哪种；property 标签是切分规则对应的不同属性，不同的切分规则配置不同。下面的配置就表示分成2片，若goods\_id大于512就会被分配到第二个子数据库中存储。     * **启动Mycat**   使用cmd运行bin目录下的startup\_nowrap.bat即可    可以发现启动成功，现在开始我们在对原数据库gwarbms中的goods\_id表执行SQL操作时，将会自动被映射成对dn2（gwarbms2）以及dn3（gwarbms3）的操作。具体效果见2.2.2节。 2.1.3表示层 **使用React重新开发进销存系统的交互界面**   * **安装React**   国内使用 npm 速度很慢，可以使用淘宝定制的 cnpm (gzip 压缩支持) 命令行工具代替默认的 npm  ***npm install -g cnpm --registry=https://registry.npm.taobao.org***  ***npm config set registry https://registry.npm.taobao.org***    使用 create-react-app 快速构建 React 开发环境  ***cnpm install -g create-react-app***    构建项目  ***create-react-app my-app***  ***cd my-app***  ***npm start***  根据提示进入 my-app 文件，运行项目。     * **利用React开发交互界面**   系统通知界面    商品编辑交互页面如下：    查看商品界面   2.2 每次改造的效果的具体体现（运行截图及文字等的论述）是什么？2.2.1计算层 **使用Hadoop集群环境计算遗留的“进销存”系统中海量用户的log日志中的访问统计**  在我们的分布式系统中会有一个专门的logInfo文件用以保存用户的登录信息，其内容如下所示：      将其上传到HDFS服务器并且调用wordcount函数来对输入的文件进行处理，统计出各个用户的访问次数，结果如下：    在我们采用Hadoop的map-reduce编程之后，系统将会充分利用我们设置的三个数据块的计算资源，从而更为快速地完成分布式统计计算任务，统计出每名用户的登录次数，大大提高我们系统的计算高性能。 2.2.2数据层 **使用Mycat实现分库分表**  进销存系统中存在Table：t\_goods，存储全部的商品价格信息，各个字段如下：    由于进销存系统中涉及到许多种货品，故该表内容极为庞大，所以我们需要对该表进行分片以提高后续在该表上的sql语句执行效率。  将t\_goods进行分片：dn1和dn2，分片规则为rule1    rule1按照goods\_id属性来进行分片，规则为func1    Func1规定单表最大长度512    下面进行插入数据验证，首先连接Mycat插入3条数据    进入真实物理数据库查看结果：      可以发现的确按照我们预先设置的分片规则func1来实现了数据库表的分片（goods\_id属性值低于512的被分到gwarbms2中，否则被分到gwarbms3中）。 2.2.3表示层 在我们遗留的进销存系统中，我们采用React重新开发了几个交互界面，使得整个系统对用户更加友好，更加易用。  **系统通知界面：**    **商品编辑交互页面：**    **查看商品界面：** | | | | |
| 结对开发过程记录 | | | |  |
| **（1）角色切换与任务分工** | | | | |
| 表1-1结对开发角色与任务分工 | | | | |
| | 日期 | 时间(HH:MM - HH:MM) | 驾驶员角色 | 领航员角色 | 本段时间的任务 | | --- | --- | --- | --- | --- | | 6.30 | 15:00——18:00 | 梅智敏 | 王国庆 | 使用Hadoop的map-reduce编程技术对遗留系统的计算层优化 | | 7.1 | 17:00——19:00 | 王国庆 | 梅智敏 | 使用Mycat和Redis完成对遗留系统的数据层优化 | | 7.2 | 15:00——16:00 | 梅智敏 | 王国庆 | 使用React完成对遗留系统的表示层优化 | | | | | |
| **（2）工作日志** | | | | |
| 由领航员负责记录，记录结对开发期间的遇到的问题、两人如何通过交流合作解决每个问题的。 | | | | |
| 表1-2 结对开发工作日志 | | | | |
| | 日期/时间 | 问题描述 | 最终解决方法 | 交流过程 | | --- | --- | --- | --- | | 6.30/16:00 | 计算遗留的“进销存”系统中海量用户的log日志中的访问统计时出现问题 | 通过检查整个流程发现是没有将用户登录信息文件上传到HDFS系统，上传之后再调用wordcount函数即可。 | 上网找处理流程和示例，进行自学，然后完成编码实现统计。 | | 7.1/18:00 | 开启Mycat失败，会闪退 | 使用cmd模式运行，从而可以显示报错信息，通过报错信息发现配置文件中缺少“/>”结束符号，补充之后即可。 | 我们双方交流并且合作百度。 | | 7.2/15:30 | create-react-app无法加载文件 | 在cmd模式中输入set-ExecutionPolicy RemoteSigned，并且把权限改权限为A | 我们交流之后觉得是权限不足导致的错误，结合百度到的报错信息成功解决问题。 | | | | | |
| **（3）结对开发工作现场照片** | | | | |
| 请其他同学帮助拍摄结对开发现场照片至少2张。 | | | | |
| E:\1113292969\FileRecv\MobileFile\20210513_162000.jpg E:\1113292969\FileRecv\MobileFile\20210513_161556.jpg  结对开发现场照片1 结对开发现场照片2 | | | | |
| 实验总结 | | | |  |
| * **收获与不足**   **理解了Hadoop的整体架构**：Hadoop 由许多元素构成。其最底部是 Hadoop Distributed File System（HDFS），它存储 Hadoop 集群中所有存储节点上的文件。HDFS（对于本文）的上一层是MapReduce 引擎，该引擎由 JobTrackers 和 TaskTrackers 组成。采用Hadoop 分布式计算，可以帮助我们轻松开发和运行处理大数据的应用程序。  **理解了Mycat的原理**：最重要的一个动词是“拦截”，它拦截了用户发送过来的 SQL 语句，首先对 SQL语句做了一些特定的分析：如分片分析、路由分析、读写分离分析、缓存分析等，然后将此 SQL 发往后端的真实数据库，最后收集和处理所有分片返回的结果数据，并输出到客户端。  **初步掌握React的知识：**我们了解了React，并与我们遗留的进销存系统进行结合，构建了一些新的交互界面，通过本次学习对数据表示层的相关框架知识有了更深的理解。  不足：我们对于Map-Reduce的使用还不够深入，目前只能利用它进行一些简单的计算，后续我们还需要更多的锻炼，多多熟悉Map-Reduce以最大限度优化我们分布式进销存系统的计算性能。   * **事件系统架构风格理解**   **构件：**对象或过程，并提供如下两种接⼝：过程或函数，充当事件源或事件处理器的⻆⾊；事件。  **连接器**：事件-过程绑定：事件处理器(事件的接收和处理⽅)的过程向特定事件注册，事件源构件发布事件，当某些事件被发布时，向其注册的过程被隐式调用，调用的次序是不确定的。  **特点**：事件的触发者并不知道哪些构件会被这些事件影响，相互保持独立。不能假定构件的处理顺序，甚⾄不知道哪些过程会被调用。各个构件之间彼此之间⽆连接关系，各自独立存在，通过对事件的发布和注册实现关联。   * **观测者模式理解**   指多个对象间存在一对多的依赖关系，当一个对象的状态发生改变时，所有依赖于它的对象都得到通知并被自动更新。这种模式有时又称作发布-订阅模式、模型-视图模式，它是对象行为型模式。  观察者模式是一种对象行为型模式，其主要优点如下：   * 降低了目标与观察者之间的耦合关系，两者之间是抽象耦合关系。符合依赖倒置原则。 * 目标与观察者之间建立了一套触发机制。   它的主要缺点如下：   * 目标与观察者之间的依赖关系并没有完全解除，而且有可能出现循环引用。 * 当观察者对象很多时，通知的发布会花费很多时间，影响程序的效率。 * **消息中间件理解**   消息中间件是利用高效可靠的消息传递机制进行异步的数据传输，并基于数据通信进行分布式系统的集成。通过提供消息队列模型和消息传递机制，可以在分布式环境下扩展进程间的通信。  **消息中间件一般有两种传递模式**：   * 点对点模式：消息生产者将消息发送到队列中，消息消费者从队列中接收消息。消息可以在队列中进行异步传输。 * 发布/订阅模式：发布订阅模式是通过一个内容节点来发布和订阅消息，这个内容节点称为主题（topic），消息发布者将消息发布到某个主题，消息订阅者订阅这个主题的消息，主题相当于一个中介。主题是的消息的发布与订阅相互独立，不需要进行基础即可保证消息的传递。   **消息中间件的优势**：   * 系统解耦   交互系统之间没有直接的调用关系，只是通过消息传输，故系统侵入性不强，耦合度低。   * 提高系统响应时间   例如原来的一套逻辑，完成支付可能涉及先修改订单状态、计算会员积分、通知物流配送几个逻辑才能完成；通过MQ架构设计，就可将紧急重要（需要立刻响应）的业务放到该调用方法中，响应要求不高的使用消息队列，放到MQ队列中，供消费者处理。   * 为大数据处理架构提供服务   通过消息作为整合，大数据的背景下，消息队列还与实时处理架构整合，为海量数据的快速处理提供技术支持。 | | | | |
| 教师评语 | | | |  |
|  | | | | |