浙江大学

大规模信息系统构建技术导论

分布式 MiniSQL 系统设计报告

2021 学年　第一学期（上）

组员信息

学号	姓名
3190106087	裴睿韬
3190103179	赵文琛
3190105239	张柏涵

2022 年 4 月 12 日

目　录

1 引言

本次的项目开发为一个分布式 MiniSQL 系统，根据《数据库系统》课程所学的基本知识以及本课程所学的大规模信息技术系统构建的相关知识来进行项目的开发，通过 Client，Master，Region Server 以及 Zookeeper 集群等模块的设计，实现一个在多主机上共享数据资源访问的分布式数据库系统。

1.1 系统目标

扫雷游戏是 Windows 附件中的一个很有趣的游戏。本文使用 Java 语言编写一个与其类似的扫雷游戏。具体功能如下：

此次的实验项目为一个分布式 MiniSQL 系统，项目整体采用 Java 语言进行编写，需要实现的具体功能如下：

(1) 数据库查询：支持基本的 SQL 语句查询，包括数据库数据的插入、查询、修改、删除等数据库的基本功能的实现

(2) 副本管理：采用主从复制策略，选择其中的一个表作为主副本，负责副本的复制操作。

(3) 负载均衡：Master Server 可以对其他 Region Server 进行管理，当某一 Region Server 出现繁忙时，Master Server 可以将某些 Region 重新分配到其他的 Region Server 上；同时各 Region Server 可以实现数据传输，等待负载均衡之后修改 Table 的定位信息。

(4) 容错容灾：当某个 Region Server 失效后，Master Server 可以实现将其中的 Region 重新分配到其他的 Region Server 上。

1.2 设计说明

本程序采用 Java 程序设计语言，使用 Maven 作为包管理工具，在 IDEA 开发环境下进行项目的编写、调试与测试等，使用 Zookeeper 作为集群管理工具，支持在不同的操作系统与语言环境下跨平台使用。

表 1 各成员分工表

成员姓名	学号	分工
裴睿韬	3190106087	主节点（Master）开发，Interpreter 模块开发，Index 模块开发，API 模块开发
赵文琛	3190103179	从节点(Region Server)开发，Record 模块开发，Buffer 模块开发
张柏涵	3190105239	客户端开发，Catalog 模块开发，Buffer 模块开发

2 总体设计

2.1 总体架构设计

本系统为分布式 MiniSQL 系统，主要架构特点如下：

(1) 该分布式 MiniSQL 系统主要由 Client，Master，Region Server 与 Zookeeper 集群模块组成，其中 Client，Master，Region Server 分别对应客户端，主节点，节点，通过 Zookeeper 的帮助对数据表信息进行管理。

(2) MiniSQL 为 Region Server 层提供技术服务。

(3) Client 层作为应用端实现用户交互。

系统总体架构图如下所示：

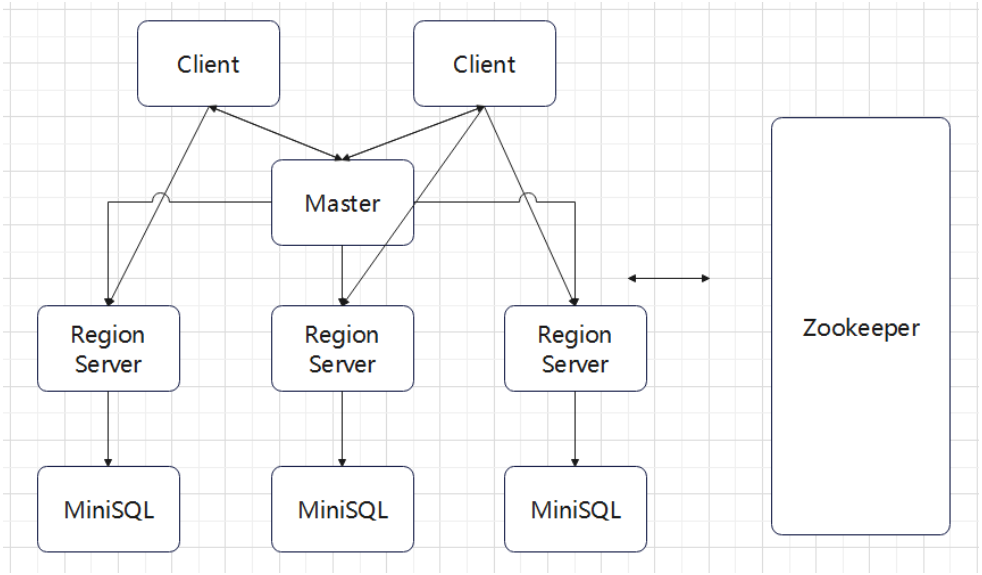

图 1 系统总体架构图

2.1.1 Client 层

在该项目中将 Client 的功能实现交由三个子类负责：

(1) 子类 1 负责客户端缓存，采用 HashMap 管理并为其设计增删改查的接口。

(2) 子类 2 负责 Client 与 Region Server 的通信，通过 sql 语句实现 Socket 建立释放等功能，并应该有多线程支持。

(3) 子类 3 负责 Client 与 Master 通信，在单独线程内完成。需要注意的是，Master 并不直接储存数据，而是储存子节点数据表。

整体工作流程图如下：

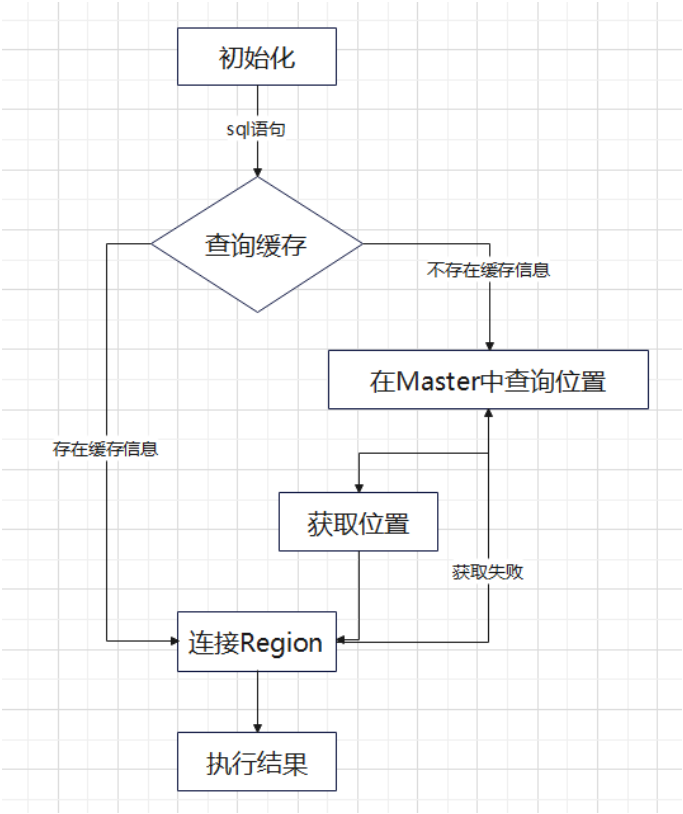

图 2 Client 整体工作流程图

2.1.2 Master 层

该项目中 Master 层的功能由三个子类实现：

(1) 建立与 Zookeeper 的连接，实现数据集群管理，对节点的状态改变进行监听。基于此可以实现容错容灾等功能。

(2) 储存维护主节点的重要信息。例如 Region Server 列表以及 ip 的相关信息等。基于此可实现容错容灾，副本管理等功能。

(3) 建立与 Client 及 Region Server 的通信，实现对 Region 的分配以及增删改查等功能。

整体工作流程图如下：

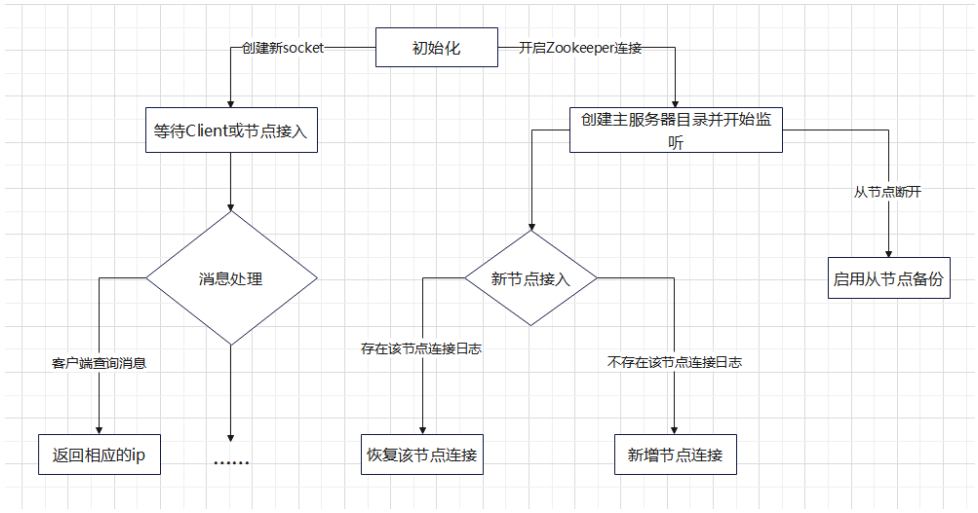

图 3 Master 整体工作流程图

2.1.3 Region Server 层

实现每一个从服务器的类由四个子类负责实现其功能：

(1) 向 Zookeeper 注册从节点信息。

(2) 向主节点发送表的信息。

(3) 建立线程与 Client 建立并保持通信。执行客户端的数据库执行语句并将备份上传至服务器。

(4) 监听主节点并与主节点保持通信。

整体工作流程图如下：

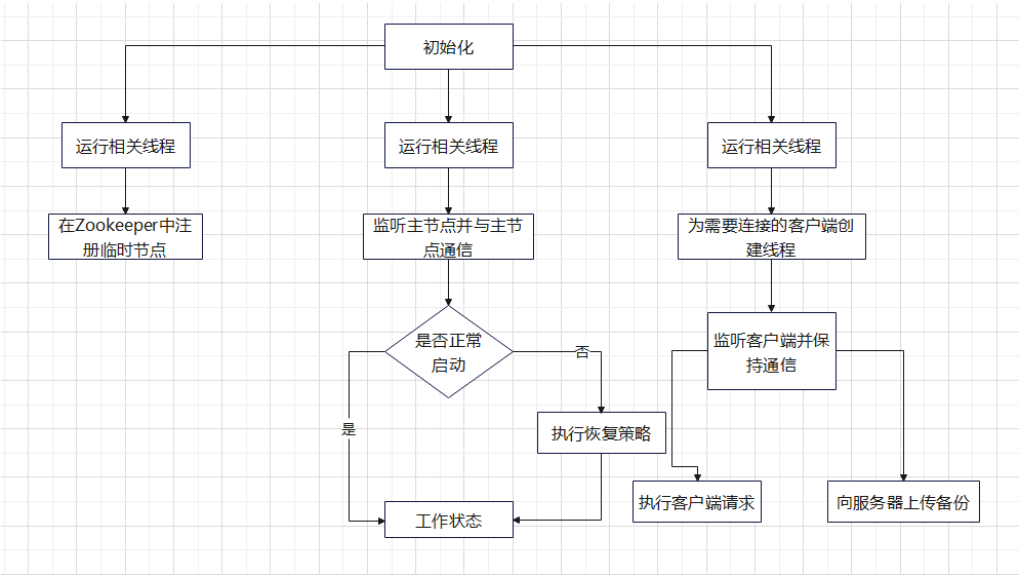

图 4 Region Server 整体工作流程图

2.1.4 Zookeeper 集群

在该项目中我们采用 Zookeeper 进行集群管理，其主要功能是：

(1) Master 与 Region Server 监控 Zookeeper 中的目录。

(2) 维持 Region 服务器列表

(3) 当 Region Server 中从服务器崩溃时，将反馈信息送达 Master 作出回应。

(4) 采取小数据储存，例如 HBase 中储存表格信息。

Zookeeper 集群的主要工作原理图如下所示：

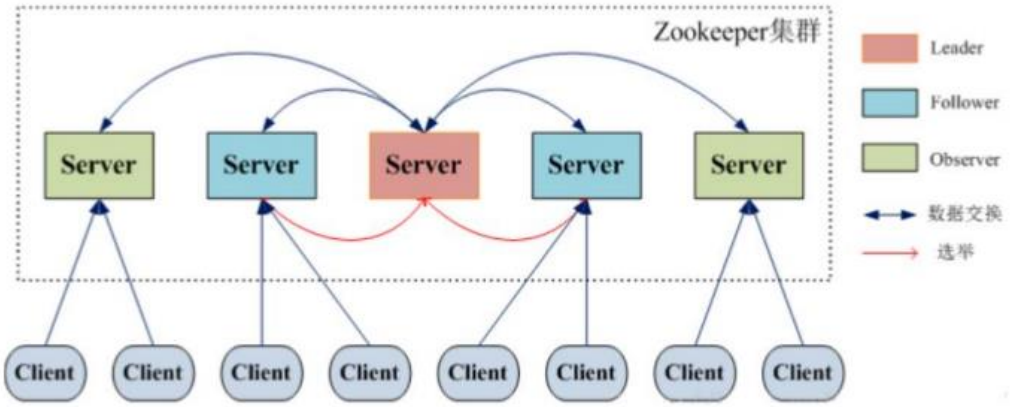

图 5 Zookeeper 集群工作原理图

2.2 MiniSQL 架构设计

MiniSQL 作为独立单元在 Region 中运行，该项目借鉴了数据库课程中 MiniSQL 的结构搭建，架构图如下所示：

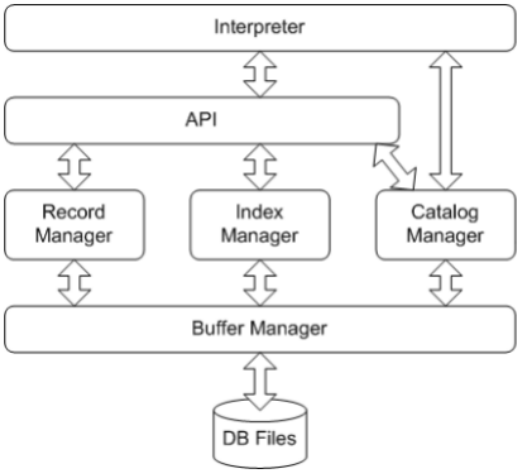

图 6 MiniSQL 架构图

3 详细设计

3.1 Master Server

主服务器 Master Server 主要是负责管理和维护表的分区信息等元数据信息。Master Server 需要实现以下功能：

(1) 维护 Region Server 列表，管理 Region Server 的元信息。

(2) 能够分配 Region，将 Region 分配到不同的 Region Server 上。

(3) 实现不同 Region Server 之间的负载均衡。

(4) 管理用户的基本数据库操作，如数据库表的增、删、改、查等。

(5) 对发生故障的 Region Server 上的 Region 进行重新迁移分配，具有一定的容错容灾能力。

3.2 Region Server

从服务器 Region Server 是 MiniSQL 的主要设计部分，需要实现存储和维护分配给自己的 Region，处理来自客户端的请求。每个独立的 Region Server 负责 MiniSQL 的来自客户端的请求，利用 MiniSQL 来管理自己的 Region，一个 Region 对应一个 Table，无需进行切分或合并；负责 MiniSQL 的启动和管理以及不同 Client 之间的通信。Region Server 还可以实现缓存机制，保存某些表的对应的定位信息，做到可以直接访问对应的 Region，减少访问负担。

3.3 系统模块设计

分布式 MiniSQL 主要由以下几个模块组成：Interpreter，API，Catalog Manager，Record Manager，Index Manager，Buffer Manager，DB Files。各模块功能如下：

3.3.1 Interpreter

Interpreter 模块直接与用户交互，接收并解释用户输入的命令，生成命令的内部数据结构表示，同时检查命令的语法正确性和语义正确性，对正确的命令调用 API 层提供的函数执行并显示执行结果，对不正确的命令显示错误信息。

Interpreter 部分可处理的语句包括：insert、select、create table、create index、drop table、drop index、delete、show、exit。

3.3.2 API

API 模块是整个系统的核心，其主要功能为提供执行 SQL 语句的接口，供 Interpreter 层调用。在 interpreter 接受、解释并执行 SQL 语句时会通过 API 来调用 Catalog manager、buffer manager 、index manager 和 record manager 几个模块来完成 SQL 命令。

3.3.3 Catalog Manager

Catalog Manager 负责管理数据库的所有模式信息，包括：

(1) 数据库中所有表的定义信息，包括表的名称、表中字段（列）数、主键、定义在该表上的索引。

(2) 表中每个字段的定义信息，包括字段类型、是否唯一等。

(3) 数据库中所有索引的定义，包括所属表、索引建立在哪个字段上等。

3.3.4 Record Manager

Record Manager 负责管理记录表中数据的数据文件。主要功能为实现数据文件的创建与删除（由表的定义与删除引起）、记录的插入、删除与查找操作，并对外提供相应的接口。其中记录的查找操作要求能够支持不带条件的查找和带一个条件的查找（包括等值查找、不等值查找和区间查找）。

3.3.5 Index Manager

Index Manager 负责 B+树索引的实现，实现 B+树的创建和删除（由索引的定

义与删除引起)、等值查找、插入键值、删除键值等操作，并对外提供相应的接口。

3.3.6 Buffer Manager

Buffer Manager 负责缓冲区的管理，主要功能有：

(1) 根据需要，读取指定的数据到系统缓冲区或将缓冲区中的数据写出到文件。

(2) 实现缓冲区的替换算法，当缓冲区满时选择合适的页进行替换。

(3) 记录缓冲区中各页的状态，如是否被修改过等。

(4) 提供缓冲区页的 pin 功能，及锁定缓冲区的页，不允许替换出去(实现过程中并没有 SQL 语句执行该操作。

3.3.7 DB Files

DB Files 指构成数据库的所有数据文件，主要由记录数据文件、索引数据文件和 Catalog 数据文件组成。

3.4 客户端

客户端指的是用户与系统进行直接交互的界面，并能够与服务器建立稳定的网络连接。客户端获取 Region 的位置信息后，直接从目标 Region Server 的对应 Region 上读取数据，并不通过 Master Server 来进行数据读取。使用 Zookeeper 进行集群管理，启动后将等待客户端连接；客户端将连接到 Zookeeper 集群中的一个 Region 节点，该节点向客户端分配特定的会话 ID 并发送确认，客户端未收到确认信息则会尝试连接其他的节点。连接建立后将会持续发送确认信息保证连接的稳定。

客户端的通信基于 RPC 通信协议。RPC 协议是一种网络通信协议，采用 C/S 架构，项目的客户端与服务器之间的通信将基于 RPC 协议。

客户端将采用缓存机制，在内存中维护一个记录 Region 的 HashMap，每次访问之前会在 HashMap 中查找相关信息，提高客户端的工作效率。

4 测试与运行

4.1 系统测试用例设计

基础的用例设计，最终成果的设计将会更为复杂。

用例编号	01
描述	Int 等值查询
输入信息	Select * from xxx where xx = 12
假设	不存在异常
输出	正常查询结果
备注	测试 int 为 0、-1、+1、+极大、+极小等

用例编号	02
描述	Int 不等值查询
输入信息	Select * from xxx where xx <> 12
假设	不存在异常
输出	正常查询结果
备注	测试 int 为 0、-1、+1、+极大、+极小等

用例编号	03
描述	Int >、< 查询
输入信息	Select * from xxx where xx > 12
假设	不存在异常
输出	正常查询结果
备注	测试 int 为 0、-1、+1、+极大、+极小等

用例编号	04
描述	测试 and 逻辑
输入信息	Select * from xxx where xx > 12 and xx <25
假设	不存在异常
输出	正常查询结果

备注	测试 and 条件不符合（查询不到值的情况）

用例编号	05
描述	测试 or 逻辑
输入信息	Select * from xxx where xx > 12 or xx <25
假设	不存在异常
输出	正常查询结果
备注	测试 or 条件不符合（查询不到值的情况）

用例编号	06
描述	测试 char 的等值查询
输入信息	Select * from xxx where xx = ‘dsds’
假设	不存在异常
输出	正常查询结果
备注	输入如 ‘\’ ’ /’ 等特殊字符

用例编号	07
描述	测试 char 的不等值查询
输入信息	Select * from xxx where xx <> ‘dsds’
假设	不存在异常
输出	正常查询结果
备注	输入如 ‘\’ ’ /’ 等特殊字符

用例编号	08
描述	测试 Double/Float 的各项功能
输入信息	Select * from xxx where xx = 3.123
假设	不存在异常
输出	正常查询结果
备注	输入小数点后多位的 Double/Float 测试

用例编号	09
描述	测试索引的创建
输入信息	Create index xx on xxx
假设	不存在异常
输出	正常查询结果
备注	尝试在某个键上创立多个索引等

用例编号	10
描述	测试索引查询
输入信息	测试使用索引和不使用索引的速度
假设	不存在异常
输出	正常查询结果
备注	使用多组数据进行测试

用例编号	11
描述	测试 delete 功能
输入信息	Delete from xxx where xx = ‘dsds’
假设	不存在异常
输出	正常查询结果
备注	测试删除不存在的表或者不符合的条件

用例编号	12
描述	测试 create 功能
输入信息	Create table xxx(int xx, char x)
假设	不存在异常
输出	正常查询结果
备注	测试各种创建情况

用例编号	13
描述	测试 insert 功能
输入信息	Insert into xxx value (xxx, ppp)
假设	不存在异常
输出	正常查询结果
备注	测试各种插入情况

用例编号	14
描述	测试 drop index 功能
输入信息	Drop index xxx
假设	不存在异常
输出	正常查询结果
备注	尝试 drop 不存在的 index

用例编号	15
描述	测试 drop table 功能

输入信息	Drop table xxx
假设	不存在异常
输出	正常查询结果
备注	尝试 drop 不存在的 table

4.2 项目规划

表 2 项目规划表

计划时间	**完成阶段**
4 月中旬-4 月下旬	系统设计、职能分配、环境搭建
4 月下旬-5 月上旬	代码开发
5 月上旬-5 月中旬	测试
5 月中旬-5 月下旬	报告以及文档书写、以及验收

5 总结

我们的项目由三位同学在一个月内完成。从时间上还是比较紧凑。开发流程准备使用底层语言如 C/C++、Java 进行后端的开发，前端的命令行界面可能会使用比较方便的脚本语言进行开发。在比较紧张的情况，我们会加紧进度，尽量完成一个符合标准的完善系统。并在此过程中掌握 sql 语言和大规模数据的开发。并对分布式系统有进一步的了解。