2024-5-30

俞博

3210105512

个人报告

大规模信息系统构建技术导论

目录

[1 个人开发过程与主要贡献 1](#_Toc168000713)

[2 个人模块设计 2](#_Toc168000714)

[2.1 Master模块 2](#_Toc168000715)

[2.2 Region模块 4](#_Toc168000716)

[2.3 Client模块 6](#_Toc168000717)

[2.4 Util模块 6](#_Toc168000718)

[3 心得体会+小结 8](#_Toc168000719)

# 1 个人开发过程与主要贡献

在本次大规模分布式minisql的开发中，我主要负责部分分布式查询、部分副本管理、容错容灾与负载均衡，该部分也是本项目的设计亮点与相对难点所在。在本次开发过程中，我主要负责设计了主从对的副本管理，规定了必须保证投入使用的region服务器都为pair对，并根据主服务器断联有待用（Available）服务器、副本服务器断联有待用服务器、主服务器断联无待用服务器和副本服务器断联无待用服务器这四种情况设计了不同的容错容灾方法。有关负载均衡，我实现了基于hot code操作数的两种策略，一是在创建表时选择现有表数量最小且不处于繁忙状态的服务器，二是在不断的操作过程中，把繁忙服务器的一半的表按照Best Fit的算法调至最优的不繁忙服务器中，以实现整体系统的负载均衡。在完成自己的模块后，我还参与了项目后期的整合与debug，并最终为整体的系统进行了测试。

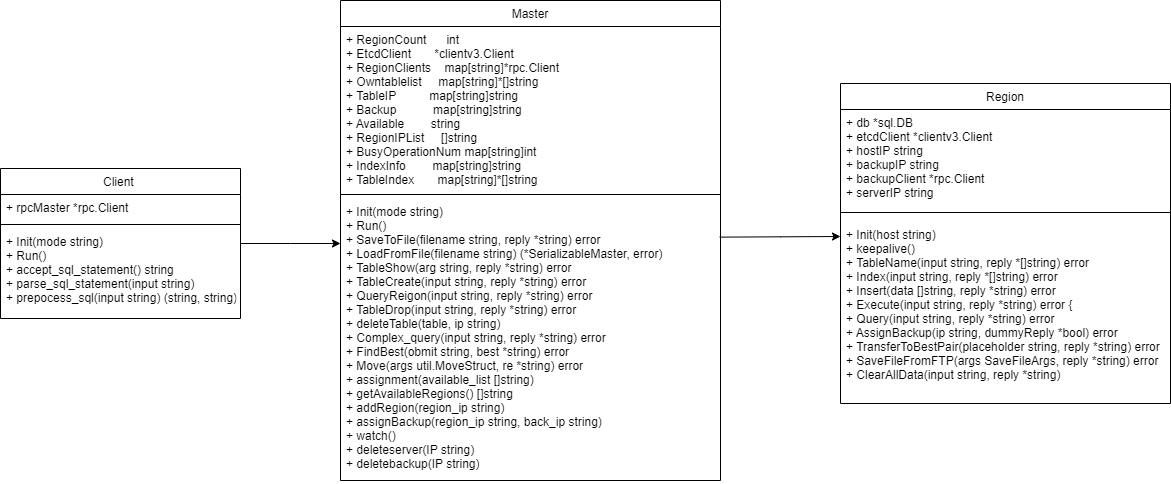
本系统主要由小组3位成员合作编写完成，使用go语言进行开发，分别使用vscode、golang作为主要开发工具，使用github进行版本管理以及团队协作。我的分工如下：

|  |  |  |
| --- | --- | --- |
| 成员姓名 | 学号 | 分工职责 |
| 俞博 | 3210105512 | 部分分布式查询，部分副本管理，容错容灾，负载均衡 |

# 2 个人模块设计

## 2.1 Master模块

Master的设计如下图所示:



在Master模块中，我主要实现了部分分布式查询、容错容灾与负载均衡的相关部分。具体的函数设计如下：

func (master \*Master) GetTableIP(table string, reply \*string) error：

该函数用于查询 table 和 region 的对应关系。通过调用这个函数，可以获取指定 table 所在的 region 的 IP 地址。（隶属工具函数）

func (master \*Master) TableDrop(input string, reply \*string) error：

该函数用于在数据库中删除 table。通过调用这个函数，可以从数据库中移除指定的 table。（隶属部分分布式查询）

func (master \*Master) check\_and\_reset\_Regions() error：

该函数用于检查和重置 region的hot code操作数。通过调用这个函数，可以确保所有 region 的状态正常，并在必要时进行重置。（隶属负载均衡）

func (master \*Master) deleteserver(IP string)：

该函数用于处理某主服务器下线。如果有待用 region，启动副本 region 作为新的主服务器，待用节点作为其副本服务器；否则把副本 region 中的内容都转存到某个 region 对中，将副本 region 变成待用 region。（隶属容错容灾）

func (master \*Master) deletebackup(IP string)：

该函数用于处理某副本服务器下线。如果有待用 region，将其设为副本 region；否则把主服务器 region 中的内容都转存到其他 region 对中，将主 region 转换为待用 region。（隶属容错容灾）

func (master \*Master) LoadBalance(placeholder string, re \*string) error

该函数用于检测处于繁忙状态的region服务器，并将其中一半的表挪到最优的region服务器中（表数量最小，且不处于繁忙状态），并重置繁忙状态的操作数busy\_operation\_num。（隶属负载均衡）

func (master \*Master) MoveHalf(args util.MoveStruct, re \*string) error

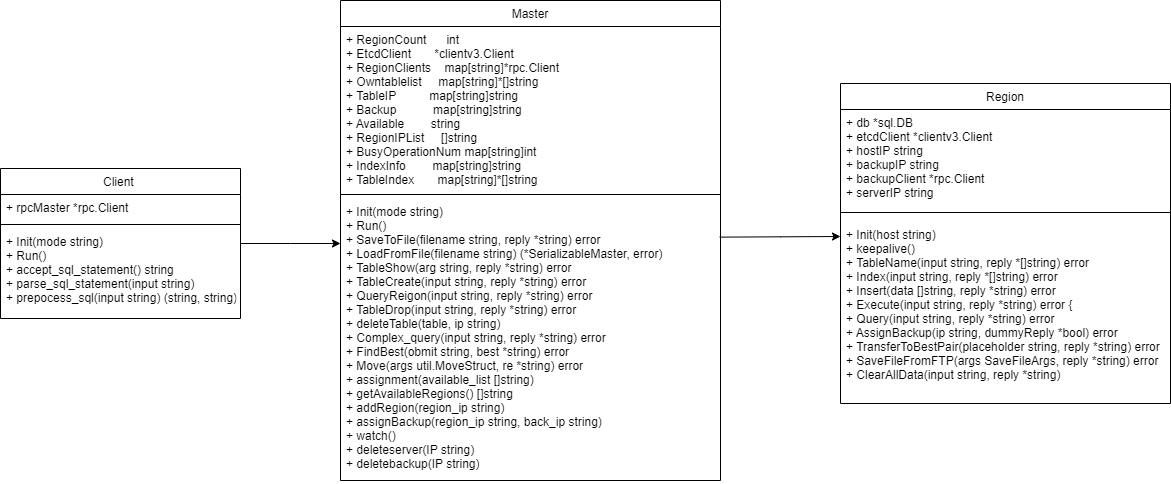
该函数作为LoadBalance的调用函数，实际实现了将指定的region服务器中的一半的表格挪到最优的服务器中（表数量最小，且不处于繁忙状态）。（隶属负载均衡）

func (master \*Master) Move(args util.MoveStruct, re \*string) error：

该函数用于将 table 移动到指定的 region。通过调用这个函数，可以将 table 从一个 region 移动到另一个 region，以实现负载均衡或其他管理需求。函数会更新相关的缓存表和映射关系，确保系统中的数据一致性。（隶属容错容灾与负载均衡）

## 2.2 Region模块

Region的类图如下：



在Region模块中，我主要实现了三个重要的功能性函数，用于规定单一服务器执行操作（非主从服务器对一起执行）、调用ftp来实现服务器的数据互传，具体的函数设计如下：

func (region \*Region) Exe(input string, reply \*string) error：

该函数用于处理非查询类的 SQL 语句。通过调用这个函数，可以执行插入、更新、删除等操作，并返回执行结果。此函数为指定单一服务器的非查询类的SQL操作。

func (region \*Region) TransferToBestPair(bestIp string, reply \*string) error

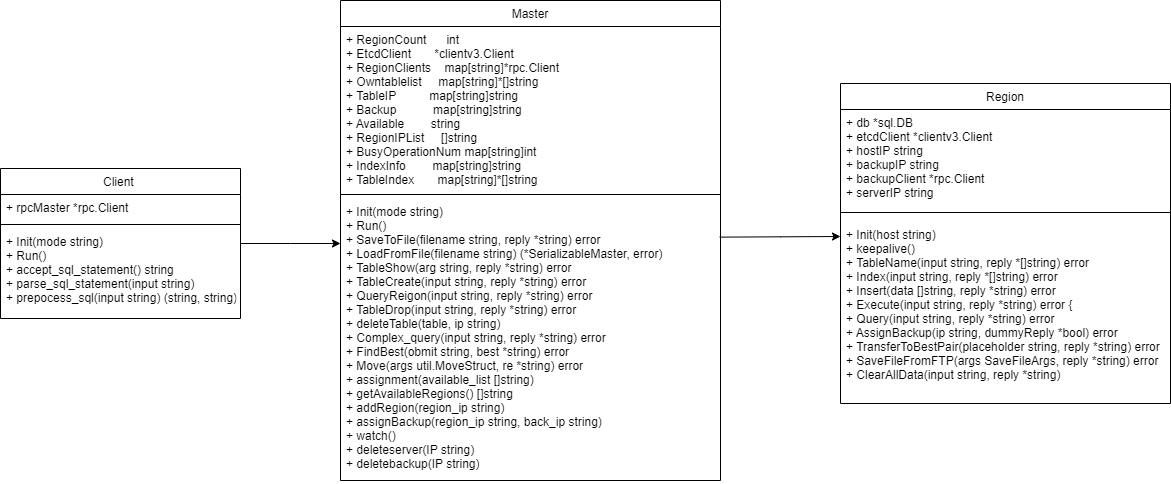
该函数用于将指定的region服务器中的所有内容转移到最优的主从服务器对中。实际的Master中的缓存表操作不交由此函数执行，此函数只进行实际的region服务器的数据转移。

func (region \*Region) SaveFileFromFTP(args SaveFileArgs, reply \*string) error

该函数用于将被传输到FTP服务器中的文件下载到指定的region服务器中，该函数可通过在args指定文件名称下载任意的文件或是默认下载region服务器的sqlite的.db文件。

## 2.3 Client模块

Client的类图如下：



我在Client模块中主要完善函数逻辑，保证在每次的sql操作（增删改查）都会增加region服务器的操作数，并在每次操作的结束保证调用检查、重置和负载均衡的函数。

## 2.4 Util模块

在util模块中，有几个关键函数用于实现模块的核心功能。这些函数涵盖了元素查找、RPC调用超时处理、字符串切片操作以及目录清理等操作。以下是每个函数的详细描述及其实现逻辑：

FindElement(pSlice \*[]string, str string) int

**功能描述：**

该函数在给定的字符串切片中查找指定的字符串，并返回其索引。如果找不到该字符串，则返回-1。

**实现逻辑：**

遍历字符串切片，比较每个元素与目标字符串。如果找到匹配的字符串，返回其索引。如果遍历结束仍未找到目标字符串，则返回-1。

该功能常用于在需要查找特定元素的位置时调用，比如在管理集群节点列表时。

DeleteFromSlice(pSlice \*[]string, str string) bool

**功能描述：**

该函数在字符串切片中删除指定的字符串。如果成功删除，返回true；否则返回false。

**实现逻辑：**

遍历字符串切片，找到与目标字符串匹配的元素。删除匹配的元素，并调整切片长度。返回删除操作的结果状态。

该功能常用于从集群列表或其他字符串集合中移除特定元素。

AddToSlice(ptr \*[]string, newString string)

**功能描述：**

该函数在字符串切片的末尾添加一个新的字符串。

**实现逻辑：**

将新的字符串附加到字符串切片的末尾。更新切片长度。确保新的字符串成功添加到切片中。

该功能用于动态扩展字符串集合，例如添加新的节点或配置项。

AddToSliceIndex(ptr \*[]string, newString string, index int)

**功能描述：**

该函数在字符串切片的指定位置插入一个新的字符串。

**实现逻辑：**

确定插入位置是否合法（在切片范围内）。在指定位置插入新的字符串，并调整切片长度。确保插入操作不会破坏现有数据。

该功能用于在特定位置插入新的元素，例如在有序列表中插入新项。

func DeleteValueFromMap(mp \*map[string]string, value string) error

**功能描述：**

该函数在字典中删除与传入变量相同的key-value对。如果成功删除，返回true；否则返回false。

**实现逻辑：**

遍历字典，找到与目标字符串匹配的value。删除匹配的key-value对。返回删除操作的结果状态。

该功能常用于从Master的缓存表中移除特定元素。

# 3 心得体会+小结

通过这个项目的开发，我收获了很多宝贵的经验和心得体会。

这个项目让我有机会将理论知识应用到实际开发中，通过这个项目这个项目把在课堂上学到的分布式系统和大规模信息系统的理论知识，充分地运用到了实践中。我们设计并实现了一个分布式大规模数据库系统，从系统架构到具体功能的实现，每一步都让我对这些理论有了更深刻的理解。

在项目开发过程中，我们三名成员紧密合作，各自负责不同的模块，并定期进行沟通和协调。通过这种方式，我们不仅提高了工作效率，还学会了如何在团队中高效协作，如何处理团队成员之间的分歧，以及如何在项目进度出现问题时共同解决。

我的项目管理能力也得到了锻炼。我们制定了详细的项目计划，明确了每个阶段的目标和任务，并定期进行进度汇报和调整。通过这种方式，我们学会了如何合理分配时间和资源，如何在项目进展中保持灵活性，以及如何在面对不确定性时进行有效的决策。

这个项目还让我对大规模信息系统的构建有了更深入的理解。我们学习了如何设计和实现一个分布式系统，如何进行数据的分布式存储和管理，以及如何确保系统的高可用性和容错性。这些知识和经验将对我未来的研究和工作产生深远的影响。

总的来说，这个项目不仅是一次重要的实践机会，更是一次宝贵的学习经历。通过这个项目，我在理论知识、团队协作和项目管理等方面都得到了全面的提升。我相信，这些经验和能力将在这个AI的盛行时代，为我未来的学术研究和职业发展打下坚实的基础。