**2024-2025-2学期**

**C++项目报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 文件状态  [ ]草稿  [ ]正式发布  [ ]正在修改 | 文件标识 |  | |
| 当前版本 |  | |
| 小组成员 | （学号） | （姓名） |
| （学号） | （姓名） |
| （学号） | （姓名） |
|  |  |  |
| 完成日期 |  | |

电子科技大学信息与软件工程学院

版本历史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本/状态 | 作者 | 参与者 | 起止日期 | 备注 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

目录

[第一章 引言 1](#_Toc147653016)

[1.1 项目背景 1](#_Toc147653017)

[1.2 术语与缩写解释 1](#_Toc147653018)

[第二章 系统需求 2](#_Toc147653019)

[第三章 系统设计 3](#_Toc147653020)

[第四章 系统实现 4](#_Toc147653021)

[第五章 系统测试 5](#_Toc147653022)

[第六章 结语 6](#_Toc147653023)

[参考文献 7](#_Toc147653024)

# 引言

（报告正文内容不少于2000字。报告写好后删掉红色文字。）

## 1.1 项目背景

本项目是《C++现代程序设计》课程的期末综合性大作业 。项目的核心目标是运用本学期所学的C++知识群，结合其他相关计算机科学基础，对标业界成熟的关系型数据库管理系统MySQL，自主设计并开发一个微型数据库管理系统（DBMS） 。

该项目要求我们以小组形式进行，通过明确的分工协作，共同完成从需求分析、系统设计、编码实现到最终测试的全过程 。开发过程中，我们被要求采用面向对象分析（OOA）与设计（OOD）的方法 ，并遵循C++20及以上的新标准进行编码 。这不仅是对我们C++编程能力的一次全面检验和实践，更是深化我们对数据库系统底层工作原理（如SQL解析、数据存储、索引机制等）理解的宝贵机会。通过完成此项目，我们旨在巩固理论知识，并锻炼解决复杂问题的工程实践能力。

（简要介绍项目背景。不少于100字）

## 1.2 术语与缩写解释

|  |  |
| --- | --- |
| 术语/缩写 | 解释 |
| DBMS | Database Management System (数据库管理系统)的缩写。 |
| DDL | Data Definition Language (数据定义语言)，SQL的一部分，用于定义数据结构，如 CREATE TABLE。 |
| DML | Data Manipulation Language (数据操纵语言)，SQL的一部分，用于管理数据，如 SELECT, INSERT。Object-Oriented Analysis / Object-Oriented Design (面向对象的分析与设计)。 |
| OOA/OOD | Object-Oriented Analysis / Object-Oriented Design (面向对象的分析与设计)。 |

（除以上示例的两节外，可根据实际情况自行添加。以下同此。）

# 第二章 系统需求

（说明系统的需求，需要画出系统的功能结构框图并做简要介绍。

提出系统要达到的预期目标。

必要是可以列出二级或更多级标题。

必须采用OOA方法。

以下同。）

本章将根据项目任务书的要求，采用面向对象分析（OOA）的方法，详细阐述该微型数据库管理系统的功能性与非功能性需求，并明确系统需要达成的预期目标 。

#### ****2.1 功能需求****

系统需要实现一个简化版的数据库管理系统，核心功能围绕数据定义、数据操作、数据存储和用户交互四个方面展开。

##### ****2.1.1 存储功能****

* 文件系统存储：系统需利用宿主操作系统（Linux）的文件系统来持久化数据。
* 文件与目录结构：数据表和索引（如果存在）应以独立文件的形式存储。 目录结构和文件格式可自行设计，例如，可以为每个数据库创建一个主目录，内部存放该库的表文件和索引文件 。

##### ****2.1.2 数据定义语言 (DDL)****

系统必须支持以下DDL语句：

* create database <dbname>：用于创建一个新的数据库。
* drop database <dbname>：用于删除一个已存在的数据库。
* use <dbname>：用于切换当前工作的数据库上下文。
* create table <table-name>(<column> <type> [primary], ...)：在当前数据库中创建新表。
  + 命名规则：表名和列名需为全英文小写，且不包含下划线 \_ 和特殊字符。
  + 数据类型：简化支持 int 和 string 两种类型。string为最长256个字符的定长UTF-8编码字符串。
  + 主键与索引：若某列被指定为 primary（主键），系统必须为该表建立索引。索引的数据结构可自行确定。
* drop table <table-name>：删除一个表，并同时删除其对应的索引文件（如果存在）。

##### ****2.1.3 数据操作语言 (DML)****

系统必须支持以下DML语句：

* select <column> from <table> [ where <cond> ]：查询数据。
  + 列选择：支持查询单个指定列或使用 \* 查询所有列。
  + 条件子句 where：可选，用于过滤数据。 条件表达式 <cond> 的格式为 <column> <op> <const-value>，其中操作符 <op> 支持 =、<、> 三种。
  + 索引利用：在执行查询时，如果表存在索引，应优先使用索引以提高效率。
* insert <table> values (<const-value>[, <const-value>...])：向表中插入一条新记录。字符串常量需要用双引号 "" 括起来。
* delete <table> [ where <cond> ]：根据 where 子句的条件删除表中的记录。
* update <table> set <column> = <const-value> [ where <cond> ]：根据 where 子句的条件更新记录。若无 where 子句，则更新整张表。字符串常量同样需要使用双引号。

##### ****2.1.4 交互界面****

* 界面模式：需要模仿MySQL的命令行交互式界面。
* 操作反馈：每条命令执行后，都必须向用户提供清晰、明确的反馈信息。
* 退出机制：提供 exit 命令，用于正常退出程序。

#### ****2.2 非功能性需求****

* 开发平台：项目必须在Linux操作系统下进行开发和编译。
* 编程语言：代码主体必须是C++，且C++代码量需占压倒性比例。
* C++标准：代码需至少符合C++20标准，鼓励使用C++23。
* 外部库限制：允许使用STL库（包括多线程库），但禁止使用任何其他第三方库，如网络库。应用定位为单机程序。
* 构建工具：必须使用 make 工具来构建应用程序，并提供名为 Makefile 的默认依赖文件。
* 错误处理：需要自行设计所有可能出现的错误（如表不存在、语法错误等）及其处理机制。
* 代码托管：强烈建议使用Git进行版本控制。

#### ****2.3 系统功能结构框图****

虽然无法在此直接绘制图形，但系统的功能结构可以规划如下：

* **微型数据库管理系统 (MiniDBMS)**
  + **用户接口层 (User Interface)**
    - 负责接收用户输入的SQL命令字符串。
    - 显示命令执行结果或错误信息。
  + **SQL解析层 (SQL Parser)**
    - 对输入的命令字符串进行词法分析和语法分析。
    - 将合法的命令解析成内部数据结构，传递给执行器。
  + **命令执行层 (Command Executor)**
    - **DDL执行器**：处理 create, drop, use 等数据库和表结构定义命令。
    - **DML执行器**：处理 select, insert, delete, update 等数据操作命令。
  + **存储引擎层 (Storage Engine)**
    - **文件管理器**：负责数据库、表、索引等文件的创建、删除和读写。
    - **数据管理器**：负责记录的增、删、改、查等具体操作。
    - **索引管理器**：负责索引的创建、维护和使用，以加速数据检索。

#### ****2.4 预期目标****

* 成功实现任务书中要求的全部5条DDL和4条DML语句的功能。
* 提供一个稳定、流畅的命令行交互界面，并能对用户操作给出正确反馈。
* 建立一套合理的文件存储机制，能正确地持久化数据库、表和索引信息。
* 确保数据操作的准确性，查询结果符合预期。
* 设计全面的测试用例，覆盖所有核心功能，并通过测试。
* 系统具备基本的错误处理能力，对于非法操作或输入能给出提示，不会轻易崩溃。

# 第三章 系统设计

（说明你是如何对系统进行设计的，包括系统结构设计、模块设计、类图、界面设计和功能设计等。必须采用OOD方法。）

本章采用面向对象设计（OOD）方法，对系统的架构、模块、核心类以及关键机制进行详细设计。

3.1 系统总体架构（C/S架构）

我们设计的核心是一个清晰的客户端/服务器模型，将用户交互与数据管理完全分离。

* 客户端 (Client)：一个轻量级应用，是用户与数据库系统交互的唯一入口。它不处理任何数据逻辑，仅负责：1) 读取用户输入；2) 将SQL字符串通过网络发送给服务器；3) 接收服务器的响应并打印。
* 服务器 (Server)：一个后台守护进程，是整个系统的核心。它负责：1) 监听网络端口，管理客户端连接；2) 接收SQL命令并解析；3) 执行DDL/DML操作；4) 管理磁盘上的物理数据；5) 执行事务控制。
* 网络层 (Network Layer)：作为客户端与服务器之间的桥梁。我们基于Linux的Socket API自行实现了一个简单的应用层协议，用于可靠地传输SQL命令和结构化的响应数据。

3.2 服务器端设计

服务器端的设计采用了高度模块化的分层架构，并广泛应用Pimpl模式。

* 网络监听层：负责监听端口，接受新的客户端连接，并为每个连接创建一个会话线程或将其加入事件循环。
* 请求处理层：读取客户端发来的SQL数据流，将其反序列化为字符串命令。
* API门面层 (Database类)：接收到命令字符串后，通过解析（此处省略具体解析器设计）将其分派给DDLOperations或DMLOperations等功能模块。此类是服务层逻辑的统一入口 。
* 功能模块层 (DDL/DML/AccessControl/TransactionManager): 每个模块负责一类特定功能，并在执行前进行严格的权限检查 。
* 核心实现层 (Database::Impl): 封装了所有共享状态（内存表、用户、事务日志）和底层文件I/O操作，确保了数据的一致性和管理的集中化 。

3.3 客户端与网络协议设计

* 客户端设计：客户端是一个简单的循环：Read-Send-Receive-Print。它维护一个到服务器的TCP长连接。
* 网络协议设计：为保证通信的可靠性，我们设计了一个简单的自定义协议：
  + 请求 (Client -> Server)：[4字节长度字段][SQL命令字符串]。长度字段表示后面紧跟的SQL命令的字节数。
  + 响应 (Server -> Client)：[1字节状态码][4字节消息长度字段][消息字符串][可选的数据体]。
    - 状态码: 0表示成功，1表示逻辑错误，2表示权限错误等。
    - 消息字符串: 如"Query OK, 1 row affected"或错误详情。
    - 数据体: 对于SELECT查询，这里会包含序列化后的结果集（例如，先行数，后跟各行数据）。

3.4 存储结构设计

服务器端的物理存储结构与之前报告所述一致。

* 目录结构: ./db\_root/<database\_name>/ 。
* 文件: 每个表对应三个文件：<table\_name>.meta (存储Schema)、<table\_name>.dat (存储数据)、<table\_name>.idx (存储索引) 。

3.5事务设计

服务器端的访问控制与事务机制也与之前报告所述一致，它们是服务层的核心安全与一致性保障。

* 事务管理: 为每个客户端会话维护独立的事务状态，INSERT/UPDATE/DELETE操作都会被记录在会话的内存日志中，直到COMMIT或ROLLBACK 。

# 第四章 系统实现

（包括主要界面的截图、主要功能的流程图和/或的核心代码。每段代码原则上不超过1页。代码主题必须是C++，但允许用其他语言作为辅助。）

# 第五章 系统测试

（给出测试用例和测试结果，包括性能、有无错误等等。要对标预期结果）

# 第六章 结语

（对项目做出总结。项目的不足、未来展望等等。每位成员的心得体会。）

# 参考文献

（列出参考文献，不少于5篇。格式参照标准格式）

**附1：评分表**

**C++项目评分表**

1. 学生自评

请小组组长填写此表。

|  |  |  |  |
| --- | --- | --- | --- |
| 成员 | 学号 | 承担任务量 | 学生自评等级 |
| Xxx | Xxx | 34% | A |
| Xxx | Xxx | 33% | A- |
| Xxx | Xxx | 33% | B+ |

注：

1). 承担任务量用百分比。标红内容为示例

2). 等级为A+/A/A-/B+/B/B-/C+/C/C-/D+/D/F，其中A+最好，F最差；只有F为不合格等级。

3）自评等级将作为教师评价的参考。

1. 教师评价

**下表由教师填写**

|  |  |  |
| --- | --- | --- |
| 项目完成度 | 报告质量 | 代码质量 |
|  |  |  |

注：最终成绩重，项目完成度占70%，报告质量占15%，代码质量占15%。

教师签字：



**附2：报告撰写格式要求**

1. **标题**
   1. 字体：中文宋体，西文Times New Roman
   2. 字号：1级标题四号，2级和3级标题小四号，粗体
   3. 行间距：固定值，20磅
   4. 缩进：按tab键
2. **正文**
   1. 字体：中文宋体，西文Times New Roman
   2. 字号：小四号
   3. 行间距：固定值，20磅
   4. 缩进：按tab键
   5. 对齐：两端对齐

Tips：选中所有文字，先选择字体为宋体，再选择Times New Roman。

1. **图表**
   1. 字体：中文宋体，西文Times New Roman
   2. 字号：五号
   3. 行间距：单倍行距，上下6磅
   4. 对齐：居中
   5. 位置：表头在表格上方；图题在图下方。另，表格根据窗口宽度自动调整。
   6. 编号：格式类为：章号-章内序号。例如：表1-2 xxx，图2-3 yyy。
   7. 引用：表头和图题必须在正文中有引用，例如：如图2-3所示。
2. **代码**
   1. 不用截图，粘贴文本
   2. 字体：Consolas
   3. 字号：小五号
   4. 行间距：固定值，12磅
   5. C:\Users\白忠建\AppData\Roaming\Tencent\Users\253766161\QQ\WinTemp\RichOle\Q5S%UB3AO[40U`D(}2JL7[S.png加框。选中要加框的代码，选择“开始菜单”，在工具图标栏中部可以看到这个图标：

点击它，在下拉列表中选择外侧框线。

例：

void SocketError( int nErrorCode )

{

CString errmsg;

switch ( nErrorCode )

{

case WSANOTINITIALISED:

errmsg = "WSANOTINIT";

break;

Tip：粘贴代码前，在粘贴位置按Del，直到光标移动到文档最左侧，然后再粘贴，这样可以保持代码原来的缩进格式。此后再调整字体等。

**注意：不按格式写的报告评分至少要降一个等级，甚至评为不合格！**