



哈爾濱工業大學(深圳)

HARBIN INSTITUTE OF TECHNOLOGY, SHENZHEN

## 实验四 数据库系统功能实现

2024秋



# 本学期实验总体安排

本学期**实验**课程共 **16** 个学时， **3**个实验项目， 总成绩为 **30** 分。

实验项目	MySQL及SQL的 使用		数据库设计	数据库系统功能 实现	
学时	2	2	4	4	4
分数	8		8	14	



# 目录

---

1

实验目的

2

实验环境

3

实验内容

4

实验步骤



## 实验目的

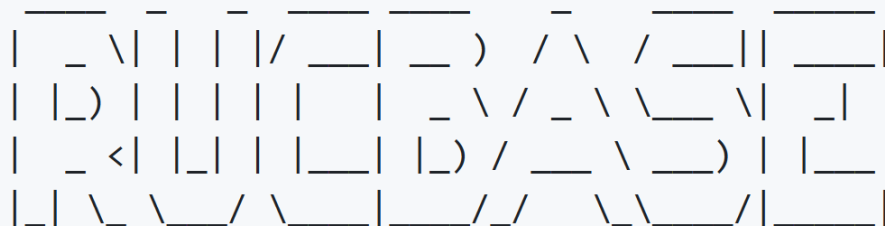
---

- 掌握缓冲池管理中的页面调度策略LRU算法。
- 学习并实现磁盘存储管理器、缓冲池替换策略，以及缓冲池管理器。
- 理解记录管理器的工作原理及其在数据库中的作用。
- 掌握定长记录的组织形式和记录操作的实现。



## 实验环境

- 操作系统：Ubuntu 18.04 及以上(64位)
- 编译器：GCC
- 编程语言：C++17
- 管理工具：cmake
- 推荐编辑器：VScode



[链接: GitHub - ruc-deke/rucbase-lab: RUC Educational Database Project open lab](#)



# 实验内容

		需编码的文件	测试文件	分值
任务一：缓冲池管理器	任务1.1 磁盘存储管理器	src/storage/disk_manager.cpp	src/test/storage/disk_manager_test.cpp	10
	任务1.2 缓冲池替换策略	src/replacer/lru_replacer.cpp	src/test/storage/lru_replacer_test.cpp	20
	任务1.3 缓冲池管理器	src/storage/buffer_pool_manager.cpp	src/test/storage/buffer_pool_manager_test.cpp	40
任务二：记录管理器	任务2.1 记录操作	src/record/rm_file_handle.cpp	src/test/storage/record_manager_test.cpp	30
	任务2.2 记录迭代器	src/record/rm_scan.cpp		

[rucbase-lab/docs/Rucbase-Lab1\[存储管理实验文档\].md](#)



## 实验步骤

---

### 环境准备 (3选1)

- 1、可参考官方文档进行安装配置：Rucbase环境配置文档及Rucbase使用文档
- 2、可使用实验室的机器进行开发测试
- 3、可使用老师提供的虚拟机进行开发测试（推荐）

# RMDB项目结构详解



## 实验步骤

### 了解框架代码

[Rucbase项目结构.pdf](#)

#### RMDB项目结构详解

代码框架

存储管理(Storage Management)

相关知识点

项目结构

文件存储组织模块: src/storage

Page

DiskManager

BufferPoolManager

缓冲区管理: src/replacer

记录存储组织模块: src/record

元数据存储组织: src/system

索引(Index)

相关知识点

项目结构

并发控制(Concurrency control)

相关知识点

项目结构

故障恢复(Failure recovery)

相关知识点

项目结构

查询处理与执行(Query processing and execution)

相关知识点

项目结构

语法解析

语法树结构

错误与异常处理





## 实验步骤

### 了解本次实验任务

[Rucbase-Lab1\[存储管理实验文档\]](#)

---

- [存储管理实验文档](#)
  - [任务一 缓冲池管理器](#)
    - [任务1.1 磁盘存储管理器](#)
    - [任务1.2 缓冲池替换策略](#)
    - [任务1.3 缓冲池管理器](#)
  - [任务二 记录管理器](#)
    - [任务2.1 记录操作](#)
    - [任务2.2 记录迭代器](#)
  - [实验计分](#)

本实验要求依次完成三个子任务，实现DiskManager、Replacer、BufferPoolManager类的接口。



## 实验步骤

# 开发

### 任务1.1 磁盘存储管理器

本任务要求补全DiskManager类，其负责读写指定页面、分配页面编号，以及对文件进行操作。

DiskManager 类的接口如下：

```
class DiskManager {
public:
    explicit DiskManager();           // Constructor
    ~DiskManager() = default;         // Destructor
    // 读写页面
    void write_page(int fd, page_id_t page_no, const char *offset, int num_bytes);
    void read_page(int fd, page_id_t page_no, char *offset, int num_bytes);
    // 对指定文件分配页面编号
    page_id_t allocate_page(int fd);
    // 文件操作
    bool is_file(const std::string &path);
    void create_file(const std::string &path);
    int open_file(const std::string &path);
    void close_file(int fd);
    void destroy_file(const std::string &path);
}
```

这些接口的内部实现调用了Linux操作系统下 `/usr/include/unistd.h` 提供的 `read()`、`write()`、`open()`、`close()`、`unlink()` 等函数。

#### (1) 读写页面

- `void write_page(int fd, page_id_t page_no, const char *offset, int num_bytes);`
- `void read_page(int fd, page_id_t page_no, char *offset, int num_bytes);`

提示：可以调用 `read()` 或 `write()` 函数。通过 `(fd, page_no)` 可以定位指定页面及其在磁盘文件中的偏移量。注意：这里支持读写的字节长度 `num_bytes` 与调用此函数读写页面时，其值 等于页面大小 `PAGE_SIZE`，但有时也可以小于 `PAGE_SIZE`，比如只读写部分数据。



## 实验步骤

# 开发

在框架代码中找到DiskManager类：

```
src > storage > disk_manager.cpp
1  /* Copyright (c) 2023 Renmin University of China
2  RMDB is licensed under Mulan PSL v2.
3  You can use this software according to the terms and conditions of the
4  You may obtain a copy of Mulan PSL v2 at:
5  |   http://license.coscl.org.cn/MulanPSL2
6  THIS SOFTWARE IS PROVIDED ON AN "AS IS" BASIS, WITHOUT WARRANTIES OR
7  EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO
8  MERCHANTABILITY OR FIT FOR A PARTICULAR PURPOSE.
9  See the Mulan PSL v2 for more details. */
10
11 #include "storage/disk_manager.h"
12
13 #include <assert.h>    // for assert
14 #include <string.h>    // for memset
15 #include <sys/stat.h>  // for stat
16 #include <unistd.h>    // for lseek
17
18 #include "defs.h"
19
20 DiskManager::DiskManager() { memset(fd2pageno_, 0, MAX_FD2PAGE_NO); }
21
22 /**
23  * @description: 将数据写入文件的指定磁盘页面中
24  * @param {int} fd 磁盘文件的文件句柄
25  * @param {page_id_t} page_no 写入目标页面的page_id
26  * @param {char} *offset 要写入磁盘的数据
27  * @param {int} num_bytes 要写入磁盘的数据大小
```



## 实验步骤

### 开发

找到需要补充的部分：

```
disk_manager.cpp X
src > storage > disk_manager.cpp
14 #include <string.h> // for memset
15 #include <sys/stat.h> // for stat
16 #include <unistd.h> // for lseek
17
18 #include "defs.h"
19
20 DiskManager::DiskManager() { memset(fd2pageno_, 0, MAX_FD * (sizeof(std::atomic<page_id_t>) / sizeof(char))); }
21
22 /**
23  * @description: 将数据写入文件的指定磁盘页面中
24  * @param {int} fd 磁盘文件的文件句柄
25  * @param {page_id_t} page_no 写入目标页面的page_id
26  * @param {char} *offset 要写入磁盘的数据
27  * @param {int} num_bytes 要写入磁盘的数据大小
28  */
29 void DiskManager::write_page(int fd, page_id_t page_no, const char *offset, int num_bytes) {
30     // Todo:
31     // 1.lseek()定位到文件头, 通过(fd,page_no)可以定位指定页面及其在磁盘文件中的偏移量
32     // 2.调用write()函数
33     // 注意write返回值与num_bytes不等时 throw InternalError("DiskManager::write_page Error");
34 }
35
36
37 /**
38  * @description: 读取文件中指定编号的页面中的部分数据到内存中
39  * @param {int} fd 磁盘文件的文件句柄
40  * @param {page_id_t} page_no 指定的页面编号
41  * @param {char} *offset 读取的内容写入到offset中
42  * @param {int} num_bytes 读取的数据量大小
43  */
44 void DiskManager::read_page(int fd, page_id_t page_no, char *offset, int num_bytes) {
45     // Todo:
46     // 1.lseek()定位到文件头, 通过(fd,page_no)可以定位指定页面及其在磁盘文件中的偏移量
47     // 2.调用read()函数
```







## 实验步骤

### 测试

- 在build目录下执行make disk\_manager\_test:

```
问题  输出  调试控制台  终端  端口
• [09/25/24]seed@VM:~/.../rucbase-lab$ cd build
• [09/25/24]seed@VM:~/.../build$ pwd
/home/seed/temp/20198001/rucbase-lab/build 在此目录下执行
• [09/25/24]seed@VM:~/.../build$ make disk_manager_test
Scanning dependencies of target gtest
[ 10%] Building CXX object deps/googletest/googletest/CMakeFiles/gtest.dir/src/gtest-all.cc.o
[ 20%] Linking CXX static library ../../lib/libgtest.a
[ 20%] Built target gtest
[ 60%] Built target storage
Scanning dependencies of target gtest_main
[ 70%] Building CXX object deps/googletest/googletest/CMakeFiles/gtest_main.dir/src/gtest_main.cc.o
[ 80%] Linking CXX static library ../../lib/libgtest_main.a
[ 80%] Built target gtest_main
Scanning dependencies of target disk_manager_test
[ 90%] Building CXX object src/test/CMakeFiles/disk_manager_test.dir/storage/disk_manager_test.cpp.o
[100%] Linking CXX executable ../../bin/disk_manager_test
[100%] Built target disk_manager_test
```



## 实验步骤

### 测试

- 执行测试:

```
问题 输出 调试控制台 终端 端口

[ 90%] Building CXX object src/test/CMakeFiles/disk_manager_test.dir/storage/disk_manager_test.cpp.o
[100%] Linking CXX executable ../../bin/disk_manager_test
[100%] Built target disk_manager_test
• [09/25/24]seed@VM:~/.../build$ ./bin/disk_manager_test
Running main() from /home/seed/temp/20198001/rucbase-lab/deps/googletest/googletest/src/gtest_main.cc
[=====] Running 2 tests from 1 test suite.
[-----] Global test environment set-up.
[-----] 2 tests from DiskManagerTest
[ RUN      ] DiskManagerTest.FileOperation
[         OK ] DiskManagerTest.FileOperation (5 ms)
[ RUN      ] DiskManagerTest.PageOperation
[         OK ] DiskManagerTest.PageOperation (7 ms)
[-----] 2 tests from DiskManagerTest (13 ms total)

[-----] Global test environment tear-down
[=====] 2 tests from 1 test suite ran. (13 ms total)
[ PASSED  ] 2 tests.
```





# 实验步骤

## 测试

•本实验涉及的测试：

### 实验计分

在本实验中，每个任务对应一个单元测试文件，每个测试文件中包含若干测试点。通过测试点即可得分，满分为100分。

测试文件及测试点如下：

任务点	测试文件	分值
任务1.1 磁盘存储管理器	src/test/storage/disk_manager_test.cpp	10
任务1.2 缓冲池替换策略	src/test/storage/lru_replacer_test.cpp	20
任务1.3 缓冲池管理器	src/test/storage/buffer_pool_manager_test.cpp	40
任务2 记录管理器	src/test/storage/record_manager_test.cpp	30

编译生成可执行文件进行测试：

```
cd build

make disk_manager_test
./bin/disk_manager_test

make lru_replacer_test
./bin/lru_replacer_test

make buffer_pool_manager_test
./bin/buffer_pool_manager_test

make record_manager_test
./bin/record_manager_test
```

## 实验步骤

### 测试

- 可以自己新增或修改测试案例：

```
src > test > storage > disk_manager_test.cpp > DiskManagerTest > SetUp()
1  #include "storage/disk_manager.h"
2
3  #include <cassert>
4  #include <cstring>
5  #include <unordered_map>
6  #include <vector>
7
8  #include "gtest/gtest.h"
9
10 constexpr int MAX_FILES = 32;
11 constexpr int MAX_PAGES = 128;
12 const std::string TEST_DB_NAME = "DiskManagerTest_db"; // 以TEST_DB_NAME作为存放测试文件的根目录名
13
14 // Add by jiawen
15 class DiskManagerTest : public ::testing::Test {
16     public:
17         std::unique_ptr<DiskManager> disk_manager_;
18
19     public:
20         // This function is called before every test.
21         void SetUp() override {
22             ::testing::Test::SetUp();
23             // 对于每个测试点，创建一个disk manager
24             disk_manager_ = std::make_unique<DiskManager>();
25             // 如果测试目录不存在，则先创建测试目录
26             if (!disk_manager_->is_dir(TEST_DB_NAME)) {
27                 disk_manager_->create_dir(TEST_DB_NAME);
28             }
29             assert(disk_manager_->is_dir(TEST_DB_NAME)); // 检查是否创建目录成功
30             // 进入测试目录
31             if (chdir(TEST_DB_NAME.c_str()) < 0) {
32                 throw UnixError();
33             }
34         }
35 }
```



## 实验步骤

### 测试

- 可以自己新增或修改测试案例：

```
/**
 * @brief 测试文件操作 create/open/close/destroy file
 * @note lab1 计分: 5 points
 */
> TEST_F(DiskManagerTest, FileOperation) { ...

/**
 * @brief 测试读写页面, 分配页面编号 read/write page, allocate page_no
 * @note lab1 计分: 5 points
 */
> TEST_F(DiskManagerTest, PageOperation) { ...
```



## 参考资料

# 全国大学生计算机系统能力大赛

[首页](#)[成绩查询](#)[我的报名](#)[个人信息](#)[通知|新闻](#)

### 通知、新闻

[查看更多》](#)

2024年数据库管理系统设计赛全国总决赛获奖名单

[查看详情](#)

2024年数据库管理系统赛全国总决赛重要事项提醒函

[查看详情](#)

## 数据库管理系统设计赛

[大赛时刻](#)[章程与技术方案](#)[比赛内容与提交方式](#)[技术报告](#)[优秀作品](#)[技术支持](#)[常见问题](#)

2024年全国大学生计算机系统能力大赛-数据库管理系统设计赛全国总决赛获奖名单



## 参考资料

# 全国大学生计算机系统能力大赛

### 2024全国大学生系统能力大赛数据库管理系统设计赛第二场技术培训会

- 1. 直播时间：2024年5月20日19:00-20:30
- 2. 培训日程与直播地址：[点击查看](#)
- 3. 直播回放：[点击查看](#)
- 4. 直播下载：[下载](#)

题目	讲者	报告文件
主持人	彭煜玮	
分布式数据库发展简介	马晓宇	<a href="#">下载</a>
数据库管理系统-存储管理	刘斌	<a href="#">下载</a>
数据库管理系统设计赛参赛经验分享	江梓雯	<a href="#">下载</a>



## 提交方式

---

### 课后提交:

- 将实验报告、工程文件打成zip包，提交至作业提交平台（截止日期参考平台发布）

作业平台入口: <http://grader.tery.top:8000/#/login>



---

**同学们  
请开始实验吧!**