

**系统能力综合培养实验报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 学 院： | 计算机科学与技术学院 | | |
| 专 业： | 计算机科学与技术 | | |
| 班 级： | 1904 | | |
| 学 号： |  | U201915050 |  |
| 姓 名： |  | 高博文 |  |
| 指导教师： | 左琼 | | |

|  |  |
| --- | --- |
| 分数 |  |
| 教师签名 |  |

2022年 11月 06日

目 录

[1 实验任务概述 1](#_Toc118922155)

[1.1 实验介绍 1](#_Toc118922156)

[1.2实验目的 2](#_Toc118922157)

[1.3实验要求 2](#_Toc118922158)

[1.4实验分工 3](#_Toc118922159)

[2 记录管理模块 4](#_Toc118922160)

[2.1 模块概述 4](#_Toc118922161)

[2.2 模块设计 4](#_Toc118922162)

[2.3 模块实现 5](#_Toc118922163)

[2.4 实验过程 8](#_Toc118922164)

[3 系统管理模块 9](#_Toc118922165)

[3.1 模块概述 9](#_Toc118922166)

[3.2 模块设计 10](#_Toc118922167)

[3.3 模块实现 11](#_Toc118922168)

[3.4 实验过程 12](#_Toc118922169)

[4 实验总结 13](#_Toc118922170)

[6.1 工作总结 13](#_Toc118922171)

[6.2 改进方案 13](#_Toc118922172)

[附录 14](#_Toc118922173)

# 实验任务概述

## 实验介绍

HustBase是一个用于实践教学的简化版数据库管理系统，其设计目标是支持简单的SQL语句，提供基本的数据定义、数据操纵、数据查询和查询优化功能。为了便于学生在为期四周的实验周期内掌握系统框架并完成实验内容，与实际的商用DBMS相比，HustBase做了大量的功能简化，舍弃了如多用户、事务、故障恢复等常见特性，支持的SQL语法也非常有限。

HustBase的体系架构如图3.1所示。系统由页面管理、记录管理、索引管理、系统管理、查询处理、语法分析和用户界面等模块构成。各模块功能如下：

(1) 页面管理：数据库中所有的数据、元数据和索引数据均以文件的形式存储在磁盘上。Windows操作系统支持的文件结构为流文件，而数据库中的数据是以记录为单位来进行存取的，为了实现对磁盘文件的高效I/O，需要将流文件转换为记录文件。这个过程分为两步：首先，将连续的流文件划分为由若干个固定大小的页面，将其抽象为分页文件；其次，将分页文件的页面划分为若干个固定大小的记录插槽或索引项插槽，将其抽象为记录文件或索引文件。本模块提供第一步所需功能，即将流文件抽象为分页文件，并提供以页面为单位的文件读写接口，具体包括：创建、销毁、打开和关闭分页文件；遍历指定文件中的所有页面；从指定文件中读取一个特定页面；在指定文件中添加、删除及修改页面等。

(2) 记录管理：系统表和数据表中的数据均以记录为单位进行存取。本模块在页面管理模块的基础上，将文件中每个页面又划分为若干个固定大小的记录插槽，实现记录的插入、删除、修改和查找，为上层的数据操纵及数据查询功能提供支持。具体包括：创建、销毁、打开和关闭记录文件；插入、删除、修改记录；查找符合指定条件的记录等。

(3) 索引管理：利用索引，系统可以为查询提供快速访问路径。本模块在页面管理模块的基础上，将文件中每个页面又划分为若干个固定大小的索引项插槽，提供对索引项的管理。索引文件与记录文件的不同之处在于，索引采用B+树结构来组织索引项，以便提高关键字查找的效率。具体包括：创建、销毁、打开和关闭索引文件；索引项的插入、删除、修改以及查找等。

(4) 系统管理：本模块提供对数据定义（CREATE/DROP TABLE，CREATE/DROP INDEX）和数据操纵（INSERT、DELETE、UPDATE）功能的支持。系统管理模块的实现依赖于记录管理模块和索引管理模块。

(5) 查询处理：本模块用于提供对数据查询（SELECT）功能的支持，并向客户端返回最终的查询结果集。查询处理模块的功能依赖于记录管理模块和索引管理模块。

(6) 语法分析：本模块提供对SQL命令的语法分析，并将分析结果以语法树的形式提供给系统管理模块和查询处理模块，以便其做进一步的分析和处理。

(7) 用户界面：本模块为用户提供操作界面，接收用户输入的SQL命令，并向用户输出SQL命令的执行结果。

在图3.1显示的各模块中，蓝色模块的功能代码已经提供，红色模块则需要学生自行设计实现。学生根据自己的实际能力，也可以对蓝色模块的代码进行一定程度的修改以扩展系统功能，但原则上不应修改本文档中规定的模块接口。

## 1.2实验目的

(1) 了解数据库、操作系统、编译、数据结构等专业知识在关系数据库管理系统实现中的综合运用；

(2) 了解关系数据库管理软件的体系结构和设计方法；

(3) 了解关系数据库中数据和元数据的组织方法；

(4) 掌握关系数据的底层存储实现技术；

(5) 掌握基本关系操作在关系数据上的实现算法；

(6) 掌握基本SQL语句的实现方法；

(7) 掌握基本的查询优化技术。

## 1.3实验要求

基于HustBase系统的总体设计架构，根据预先给定的系统框架、部分模块和接口要求，设计并实现系统中的其余模块功能，完成一个具有基本数据定义、数据操纵和数据查询功能的单用户关系数据库管理系统。

实验课时安排为四周，按周次划分为四个阶段。除去已预先提供的用户界面、语法分析和页面管理模块，要求学生自己设计实现记录管理、系统管理、查询处理和索引管理模块以及所有模块的联调，完成整个HustBase系统的开发。

HustBase系统的开发环境为Windows 操作系统及 Visual Studio 2010，开发语言为C/C++语言。为保证良好的模块独立性、便于共同开发及阶段性检查，每个模块必须按文档规定的格式对外提供调用接口，但是模块内部的实现方案不做统一规定，由学生自行设计。

具体要求：

1) 实验内容采取2-3人小组的形式合作完成，每个学生在小组中均应承担合理的工作量，基本原则为“共同设计，分工编码”。

2) 考核方式为现场演示及答辩和测试程序验证，最终成绩根据系统完成情况、演示答辩情况和实验报告质量综合评分，根据平时个人表现、互评情况，各人最终得分会有差异。

3) 课程结束后，以小组为单位提交系统的源码、编译后的程序及实验报告。

## 1.4实验分工

我主要负责系统管理模块和记录管理模块的任务，而查询处理模块和索引管理模块的任务由史宇辰主要负责。

# 记录管理模块

## 模块概述

数据表中的数据由若干条记录构成，最终存储在磁盘文件中。本模块在分页文件的基础上，进一步将页面文件改造为记录文件，每个页面的数据区被划分为若干个记录插槽，每个记录插槽用于存放一条记录，并对外提供以记录为单位的记录文件读写接口。记录管理模块依赖于下层的页面管理模块，页面管理模块以页面为单位进行文件读写，而记录管理模块则以记录为单位进行文件读写，每条记录以页面号+插槽号为唯一标识。为简化实现，规定如下：1. 记录文件中存储的记录为定长记录；2. 同一个文件中的记录格式相同，即一个数据表对应一个文件；3. 记录不能跨页，且单条记录的长度不超过页面数据区大小。

## 模块设计

将第1页作为本记录文件的信息控制页，存放改记录文件记录的整体信息以及后面页面是否存储记录已满。数据页在普通的页的基础上添加了该页存储记录的位图，表示该页在相应的位置上（记录插槽）是否存储了有效的记录，每一位对应着一个记录插槽。

在记录信息控制页面的数据区的开头存储的数据结构为

typedef struct {

int nRecords;

int recordSize;

int recordsPerPage;

int firstRecordOffset;

} RM\_FileSubHeader;

这个结构包含了该文件中存在的记录个数、每条记录的大小、每页最多存在的记录个数、每页数据的第一个记录在数据区中的开始偏移量大小。这个数据结构之后，就是一个全文件的位图，用来记录每个页是满页还是非满页，注意在创建文件的时候要把第0页和第1页标记为满页，因为第0页和第1页分别是页面信息控制页和记录信息控制页。满页用1代表，非满页用0代表。

这个模块主要实现的功能接口：

RMCreateFile创建记录文件：根据记录的大小创建一个记录文件。

RMOpenFile打开记录文件：根据文件名，获取一个记录文件的句柄。

RMCloseFile关闭记录文件：关闭文件，关闭该记录文件占用的资源。

InsertRec插入记录：插入一条新的记录到指定的记录文件中，并且返回记录位置。

插入记录时，首先在文件中找到一个非满页（若没有则需要申请一个新的页面），在该页中找到一个空插槽并插入记录。记录插入后，要更新控制页，即将文件包含的记录数加1，同时将插槽位图中的相应位置1。如果记录插入后，该页面已经没有空插槽，则将该页面标记为满页，即在记录文件的控制页的位图上将该页面所对应的位置1。

GetRec获取一条记录：从指定的记录文件中获取指定位置的记录。

DeleteRec删除一条记录：删除指定记录文件中指定位置的记录。删除记录时，对

于一个给定的记录ID，先检查该记录ID的有效性，然后找到相应的记录。记录删除并不需要将页面中记录所占用的区域清零，只需要将插槽位图中将相应的位置0，并把该页面标记为非满页。同时将控制页中的记录个数减1。

UpdateRec更新一条记录：对指定记录文件的指定一条记录进行更新。修改记录时，

对给定的记录ID，先检查该记录ID的有效性，然后找到相应的记录，用新的记录值替换原记录值。

GetNextScan记录扫描：根据给定的条件，返回符合条件的所有记录。此操作分两步完成，用户先初始化一个指定条件的文件扫描（打开扫描），然后逐个获取符合条件的记录。

## 模块实现

（1）NextRec（获取下一条记录）:参数：文件扫描RMFileScan,如果扫描即将处理的页面号和插槽号为-1返回失败。文件ID是扫描的记录文件句柄的关联的文件描述符。根据文件ID调用PF里的GetFileHandle获取对应的文件句柄。Page从要扫描的页号循环到该文件尾页的页号，每一个页在位图中的位置bitmap对应1左移（页号对8取余）位。判断该页是否为空：用该文件句柄 中的指向控制页位图的指针的(页号/8)位和bitmap相与，如果为0就直接continue。如果不是空，就调用GetThisPage获取该页的页句柄。插槽号在要处理的页号的情况下就是扫描的插槽号sn+1，如果不是第一个页号就是从0开始。循环找，记录号比每页最大记录数小的情况下循环，同时要求页的对应数据区的位置和位图中的位置想与为空，这样记录号一直++之后，如果找到了（最后记录号小于recordsperpage）就替换scan的信息方便下一次找，然后退出，没找到就把扫描初始化然后报RM-EOF。

（2）GetNextScan（获取一个符合条件的记录）:参数：rmFileScan, rec

pData=初始地址+第一条偏移+记录大小\*sn。初始地址是页的pData，偏移是记录信息控制页的offset，然后就是recordSize\*sn。循环用nextrec找到一条记录，getthispage获取页句柄。循环i+上rmFileScan中的条件数组指针conditions，每次取一个条件，match函数匹配pData和条件是否一样，但凡有一个不一样就break。如果全都一样，就将文件扫描中的页号、插槽号给到rec中对应的pagenum和slotnum，然后别的参数对应传，进程数减一，返回成功。

（3）Match（判断条件匹配），参数:pCon,pData

先用pCon里的运算符类型判断，然后看条件左右分别是属性1还是值0，如果是属性1就用pData+pCon中的偏移量取出，不然就pCon的value取出，然后直接返回比较结果。

（4）GetRec（获取指定文件中标识符为rid的记录到rec中），参数：RMfileHandle,rid,rec.

根据记录文件句柄获取文件ID属性，然后用它再获取一次文件句柄，有错误返回。然后根据rid中页号获取一次页句柄，错误就返回。再如果rid中的valid为false，或者页号太小太大，或插槽号太大，都错。再如果，根据文件句柄中的位图和页句柄中的位图判断页和记录是否为空，空就错误。现在可以赋值了，rec的rid就是rid,valid是true，偏移量用记录信息控制页的offset和recordsize算，位置偏移就是firstoffset+recordsize\*slot，最后把数据区的pData加上位置偏移的数据块给到rec的pData中，进程减一，返回。

（5）isFull（判断一个页有没有满），把页句柄弄下来，插槽数量弄下来，插槽起始号弄下来，用页中位图一个个判断插槽是不是空的。

（6）InsertRec（插入一个新的记录到指定文件中），参数：RMFIleHandle，pData,rid。先用记录文件句柄获取文件句柄，再获取页数量。页号从0开始循环找，用两个位图找到一个已分配但是没满的页。如果没找到，调用allocate函数再分配一个页，然后将pData拷贝到页中对应数据区，数据长度就是recordSize，地址为pData加上首次偏移offset。然后把该插槽位图对应0位置值置为1。如果每页只允许一个插槽，还得把记录信息控制页的位图中该页置为1表示已满。然后页句柄标脏，进程--，然后rid中的valid、pagenum、slotnum=0各自赋值。

如果不用分配页就获取页句柄，然后从0开始找到一个空插槽，用偏移量计算位置后拷贝data,该插槽位图置为1，如果添加后页变满用ifFull判断过后，位图中页置为1。

最后记录信息控制页中文件中的记录数nRecords++，指向控制页的指针的脏位为true表示被修改了。

（7）DeleteRec(指定文件中删除rid对应的记录)，参数：RMFileHandle.rid

先获取文件句柄，再根据rid中页号获取页句柄，用和插入相同办法判断是否合法。根据slot插槽号和firstoffset计算好位置后，每一页中位图中对应的记录先置为0，具体方法是用(~(1 << (slot % 8)))和位图相与，这样对应位就置为0了。同样方法，记录信息控制页中也修改位图中页，置为0，肯定不是满页了。然后nRecords--，页句柄标脏，控制页指针标脏，进程--，返回。

（8）UpdateRec（更新指定文件中的记录，rec指向的记录结构中的rid字段为要更新的记录的标识符，pData字段指向新的记录内容）

获取文件句柄、页句柄，同样判断合法性，计算位置，拷贝数据，标脏，进程--，返回。

（9）RMCreateFile（创建一个名为fileName的记录文件，该文件中每条记录的大小recordSize。）

调用CreateFile先创建一个分页文件，然后OpenFile打开文件，获取ID，然后根据文件ID调用Allocate分配一个页，根据页句柄获取数据区指针，根据数据区起始位置+记录信息控制页大小，计算文件位图位置，将第0和1页标为满页。然后记录控制页中的filesubheader的四个属性分别赋值，记录数目前是0，记录大小由参数给出，每页最大记录数PF\_PAGE\_SIZE / (recordSize + 1.0 / 8)，第一个记录的偏移

ceil((double)fileSubHeader->recordsPerPage / 8.0)。然后页句柄标脏，然后关闭文件，关闭成功返回。

（10）RMOpenFile（根据文件名打开指定的记录文件，返回其文件句柄指针。）

先调用OpenFile打开文件获取ID，然后用ID调用getthispage获取1页句柄，也就是记录信息控制页。然后给fileHandle各属性赋值：bopen句柄正在被打开为true，fileDesc为文件ID，pHdrFrame该文件头帧的指针就是页句柄的pFrame，pHdrPage指向该文件头页的指针就是页句柄中的frame指针中的页号，pBitmap指向位图的指针就是页句柄中的实际页的数据区加上RMFileSubHeader的大小。然后fileSubHeader就是页句柄中数据区的第一个位置的记录数据。

（11）RMCloseFile（关闭文件）。

如果文件句柄中的oepn为false说明已经关闭。不然，进程--，调用CloseFile,把文件ID传进去关闭，释放空间，然后bopen置为false。返回。

## 实验过程

本模块在设计的时候，由于有系统能力培养实验指导书，所以整个结构比较清晰，逻辑清楚，不用自己做太多额外设计，数据结构也没有什么创新，只有一个用到扫描记录的时候，发现过程有些复杂，所以单独封装了一个函数，用来获取下一个有效记录，这个函数主要是用来查询符合条件的函数里用的，可以循环获取所有的记录。 在与系统模块进行联调的过程中发现了问题，发现插入记录，修改记录无效，其中页不乏有编码的错误，有些编码没有考虑清除造成了逻辑上的错误。通过设计阶段理清思路后很容易编写正确的代码。最后通过与系统模块的联调，对数据表进行反复的插入，修改，删除记录后测试结果都是正确的，程序也没有出现其它的异常。

# 系统管理模块

## 模块概述

系统管理模块负责数据定义（表和索引）和数据操纵（插入、删除、修改记录），其功能依赖于记录管理模块和索引管理模块。

一个HustBase数据库由包含在指定目录中的若干个数据库文件构成。这些数据库文件包括2个系统文件、0到多个记录文件和0到多个索引文件。其中，每个记录文件与一个表相对应，每个索引文件与一个索引相对应。2个系统文件分别为系统表文件和系统列文件，用于保存数据库的元数据，记录当前数据库中有哪些表，每张表有哪些列（属性）等信息，HustBase系统必须通过这元数据来实现对数据库的管理。

HustBase数据库中的元数据也是以表的形式存在的，用于保存表信息的表称为SYSTABLES，该表中的数据保存在系统表文件中；用于保存列信息的表称为SYSCOLUMNS，其数据保存在系统列文件中。

系统表文件中记录的结构为一个tablename和attrcount。其中，表名（tablename）占21个字节，即表名为最大长度为20的字符串。表中属性的数量（attrcount）为int类型，占4个字节。

系统列文件中记录的结构为tablename、attrname、attrtype、attrlength、attroffset、ix\_flag、indexname。其中，表名（tablename）与属性名（attrname）各占21个字节，即表名与属性名的最大长度均为20个字节。属性的类型（attrtype）为int类型，占4个字节。属性的长度（attrlength）与属性在记录中的偏移量（attroffset）均为int类型，各占4个字节。该属性列上是否存在索引的标识(ix\_flag)占1个字节，’1’表示存在索引，’0’表示不存在索引。索引的名称(indexname)占21个字节，即索引名的最大长度为20个字节。

当系统执行建表等数据定义语句时，需要对SYSTABLES及SYSCOLUMNS表进行修改，在执行数据查询及数据操纵等语句时，则需要通过查询SYSTABLES及SYSCOLUMNS表来进行语义检查，并对记录数据进行正确的构造或解析。

## 模块设计

主要包含函数功能：

RC CreateDB (char \*dbPath ,char \*dbName) 创建一个名字固定的数据库，生成系统文件。

RC DropDB (char \*dbName) 删除一个数据库。

RC OpenDB (char \*dbName) 打开一个数据库。

RC CloseDB () 关闭当前数据库。

RC CreateTable (char \*relName, int attrCount, AttrInfo \*attributes) 创建一个表，表属性类都已给出。

RC DropTable (char \*relName) 销毁名为relName的表以及在该表上建立的所有索引。

RC CreateIndex (char \*indexName,char \*relName, char \*attrName) 该函数在关系relName的属性attrName上创建名为indexName的索引。函数首先检查在标记属性上是否已经存在一个索引，如果存在，则返回一个非零的错误码。否则，创建该索引。

RC DropIndex (char \*indexName) 删除名为indexName的索引。

RC Insert (char \*relName, int nValues, Value \*values) 在表中插入具有指定属性值的新元组，nValues为属性值个数，values为对应的属性值数组

RC Delete (char \*relName, int nConditions, Condition \*conditions)删除表中所有满足指定条件的元组以及该元组对应的索引项。

RC Update (char \*relName, char \*attrName, Value \*Value, int nConditions, Condition \*conditions);更新表中所有满足指定条件的元组，在每一个更新的元组中将属性的值设置为一个新的值。

RC execute(char \*sql);执行一条SQL语句，注意SELECT不用处理，如果执行成功，返回SUCCESS，否则返回错误码。

## 模块实现

RC CreateDB (char \*dbPath ,char \*dbName) 用字符串把路径名保存下来，具体是用strcpy把dpath和dbname用\\连接形成文件夹地址，然后用mkdir指令创建文件，调用create函数，在文件夹下创建相关的系统表和列，并且创建两个文件，一个用于保存元数据，一个用于保存实际数据。如果创建成功，返回成功。

RC DropDB (char \*dbName) 首先把需要删除的路径拼接上.SYSTABLE，先尝试删除这个文件，这是为了判断这个文件是不是我们创建的数据库文件，如果删除成功，就可以再次调用rmdir指令删除整个文件夹。否则不应该删除。

RC OpenDB (char \*dbName) 调用DataBase文件打开一个数据库，这里打开的方式主要是fopen函数，Database中的open函数也是这样设计的，直接调用。

RC CloseDB () 关闭当前数据库，调用DataBase文件中的close函数关闭一个数据库，关闭元数据文件，关闭数据文件，直接调用函数就行，调用成功返回。

RC CreateTable (char \*relName, int attrCount, AttrInfo \*attributes) 创建一个表，表属性类都已给出。首先调用表创建函数create，已经在table中实现，所以调用成功时，调用open\_table打开这个表，调用update\_table\_metadata更新元数据文件，返回成功。

RC DropTable (char \*relName) 销毁名为relName的表以及在该表上建立的所有索引。调用database中实现的drop\_table函数，首先根据表名路径打开表，根据元数据列结构中查询有没有索引，如果关联索引，调用drop\_index函数删除索引，之后再调用close函数关闭表，更新元数据文件中对应表信息后，调用remove\_file函数删除表名为table\_name的表。删除成功返回。

RC CreateIndex (char \*indexName,char \*relName, char \*attrName) 该函数在关系relName的属性attrName上创建名为indexName的索引。函数首先检查在标记属性上是否已经存在一个索引，如果存在，则返回一个非零的错误码。否则，创建该索引。直接调用database中的add\_index函数，如果返回值OK就返回成功，否则返回创建错误。

RC DropIndex (char \*indexName) 删除名为indexName的索引。直接调用database中的drop\_index函数，返回值OK返回成功。

RC Insert (char \*relName, int nValues, Value \*values) 在表中插入具有指定属性值的新元组，nValues为属性值个数，values为对应的属性值数组。调用database中的insert函数插入元组，打开表之后如果有索引先open\_index打开，然后调用insert\_entry加索引，最后释放缓冲区空间。

RC Delete (char \*relName, int nConditions, Condition \*conditions)删除表中所有满足指定条件的元组以及该元组对应的索引项。调用delete函数，删除之前首先有一个条件转变函数，用于把输入的条件数组保存到文件中使用的conditions数组中，这个新条件数组作为参数可以用到条件比较函数中。删除前，先打开扫描，然后next\_scan查找下一条符合条件的记录，找到之后，首先同样的过程删除索引，然后根据之前扫描返回的记录id，调用remove\_by\_rid删除数据文件中对应内容，如果没删除成功，会直接释放cons，关闭扫描，直接返回错误。如果删除成功，则会循环删下一个，直到循环结束，释放空间，关闭扫描。

RC Update (char \*relName, char \*attrName, Value \*Value, int nConditions, Condition \*conditions);更新表中所有满足指定条件的元组，在每一个更新的元组中将属性的值设置为一个新的值。更新的前半部分过程和插入删除类似，找到符合条件的之后也是先更新索引，然后调用UpdateRec更新记录，这个更新函数在RM中实现，之后一直循环scan\_next找下一个符合条件的记录，找到就更新，最后释放cons空间，关闭索引，返回成功。

RC execute(char \*sql);执行一条SQL语句，注意SELECT不用处理switch case函数中每一个分支写的语句很简单，就是调用本模块中相对应的insert、update、delete、createtable、droptable等函数，直接将参数传进去，break即可。

## 实验过程

系统模块主要工作量就是在实现外部接口，以及部分函数的用户接口和内部模块接口，但是具体需要的各种底层实现函数各模块都已经写好。创建和删除数据库都是用系统调用完成的，比较省事，不过不是最优解，系统调用的坏处是耗费资源多，并且在桌面创建时优先级太高，会把所有应用图标移动位置。所有的外部用户接口在sys文件中写好后，与各个模块对应的接口在database中完成，

这样不用需要太多的系统代码量，整个过程层层封装。

# 实验总结

## 工作总结

（1）首先完成了记录管理部分。在页面管理模块的基础上，将文件中每个页面又划分为若干个固定大小的记录插槽，实现记录的插入、删除、修改和查找，为上层的数据操纵及数据查询功能提供支持。具体包括：创建、销毁、打开和关闭记录文件；插入、删除、修改记录；查找符合指定条件的记录等。

（2）完成系统关系部分，提供对数据定义和数据操纵功能的支持。系统管理模块的实现依赖于记录管理模块和索引管理模块。

虽然此次课设是以数据库为背景，但是也学习到了其它学科的基础知识，以及提高了基本编程能力。在页面到记录管理部分对页面缓存和利用下层抽象为上层提供记录管理的处理有了更加深刻的认识，对文件处理认识加深。数据库部分，理解了关系型数据库底层处理和检索数据的基本原理，对关系型数据库有了进一步的理解。这次课设是对数据库等一些理论知识进行了一次实践，不仅学习到了编程层面的知识也加深了认识。在合作中分模块和联调也收益蛮多，总之本次实验是一次比较成功和收获很多。但是由于时间比较紧迫，大部分时间需要用在考研上，这次模块的所有实现都是以尽量简单的方式完成，不管是数据查询的多表链接方式，还是索引管理，都是不容易实现的就直接放弃，尽量拿到一个完整数据库的基本分，实现方式也没有采用很多理论上的优化方案。

## 改进方案

数据库创建可以不用系统调用实现，本身可以重新调用用户级的库函数创建文件或者删除文件。另外查询方案也可以优化，链接和投影的方式至少不用如此基础的直接合并然后一条条查询方式，有很多理论上的简化方案，比如先投影后链接。还有索引管理后面的功能很多没有实现，有时间可以更新项目，完成相关部分。

# 附录