实验二：存储管理与查询优化算法的设计

**本次实验为二选一，选择本文档实验或project2之一完成并提交。**

1. **关系连接算法的实现**

**1. 实验目的**

掌握关系连接操作的实现算法，理解算法的I/O复杂性，使用高级语言实现重要的关系连接操作算法。

**2. 实验环境**

Windows操作系统、MinGW编译器或Microsoft Visual C++编译器。C、C++，JAVA等语言均可。(可以用其他操作系统和编程环境，数据库尽量选择MySQL)

**3. 实验内容**

**3.1 实验任务**

关系R具有两个属性A和B，其中A和B的属性值均为int型（4个字节），A的值域为[1, 40]，B的值域为[1, 1000]。

关系S具有两个属性C和D，其中C和D的属性值均为int型（4个字节）。C的值域为[20, 60]，D的值域为[1, 1000]。

1）实现关系选择算法：基于ExtMem程序库，使用高级语言实现关系选择算法，选出R.A=40或S.C=60的元组，并将结果存放在磁盘上。

2）实现关系投影算法：基于ExtMem程序库，使用高级语言实现关系投影算法，对关系R上的A属性进行投影，并将结果存放在磁盘上。

3）实现Nested-Loop Join (NLJ)、hash-join和 sort-merge-join算法：基于ExtMem程序库，使用高级语言实现以上三种join算法，对关系R和S计算R.A连接S.C，并将结果存放在磁盘上。

**注意：**

**（1）需要在课前准备好相应的数据，详见3.3节。**

**（2）需要完成全部上述全部算法，根据完成情况给分。**

**（3）本次实验需要撰写实验报告。**

**3.2 ExtMem程序库介绍**

ExtMem程序库是一个专门为本课程编写的模拟外存磁盘块存储和存取的程序库，由C语言开发。ExtMem程序库的功能包括内存缓冲区管理、磁盘块读/写，它提供了1个数据结构和7个API函数。

ExtMem程序库定义了Buffer数据类型，包含如下6个域：

* numIO：外存I/O次数；
* bufSize：缓冲区大小（单位：字节）；
* blkSize：块的大小（单位：字节）；
* numAllBlk：缓冲区内可存放的最多块数；
* numFreeBlk：缓冲区内可用的块数；
* data：缓冲区内存区域。

缓冲区内每个块的大小为blkSize个字节，其最后4个字节用来存放其后继磁盘块的地址（在ExtMem库中，我们4个字节来记录磁盘块地址，地址在程序中为unsigned int类型。若无后继磁盘块，则置为0），其余(blkSize – 4)个字节用于存放块内的记录。

ExtMem库提供了如下API函数：

* Buffer \*initBuffer(size\_t bufSize, size\_t blkSize, Buffer \*buf);

初始化缓冲区，其输入参数bufSize为缓冲区大小（单位：字节），blkSize为块的大小（单位：字节），buf为指向待初始化的缓冲区的指针。若缓冲区初始化成功，则该函数返回指向该缓冲区的地址；否则，返回NULL。

* void freeBuffer(Buffer \*buf);

释放缓冲区buf占用的内存空间。

* unsigned char \*getNewBlockInBuffer(Buffer \*buf);

在缓冲区buf中申请一个新的块。若申请成功，则返回该块的起始地址；否则，返回NULL。

* void freeBlockInBuffer(unsigned char \*blk, Buffer \*buf);

解除块blk对缓冲区内存的占用，即将blk占据的内存区域标记为可用。

* int dropBlockOnDisk(unsigned int addr);

从磁盘上删除地址为addr的磁盘块内的数据。若删除成功，则返回0；否则，返回-1。

* unsigned char \*readBlockFromDisk(unsigned int addr, Buffer \*buf);

将磁盘上地址为addr的磁盘块读入缓冲区buf。若读取成功，则返回缓冲区内该块的地址；否则，返回NULL。同时，缓冲区buf的I/O次数加1。

* int writeBlockToDisk(unsigned char \*blkPtr, unsigned int addr, Buffer \*buf);

将缓冲区buf内的块blk写入磁盘上地址为addr的磁盘块。若写入成功，则返回0；否则，返回-1。同时，缓冲区buf的I/O次数加1。

文件test.c中给出了ExtMem库使用方法的一个具体示例。

**声明：**ExtMem库是为本课程专门开发的模拟外存磁盘块存储和存取的程序库，不保证其能够真正实现对磁盘块的存取操作，同时也不保证其排除一切软件错误。本课程及ExtMem开发者不会对使用该程序库所导致的一切错误负责。

**3.3 数据准备**

使用ExtMem程序库建立两个关系R和S的物理存储。关系的物理存储形式为磁盘块序列B1, B2, …, Bn，其中Bi的最后4个字节存放Bi+1的地址。

即R和S的每个元组的大小均为8个字节。

块的大小设置为64个字节，缓冲区大小设置为512+8=520个字节。这样，每块可存放7个元组和1个后继磁盘块地址，缓冲区内可最多存放8个块。

编写程序，随机生成关系R和S，使得R中包含16 \* 7 = 112个元组，S中包含32 \* 7 = 224个元组。

**二、查询优化算法的设计**

**1. 实验目的**

掌握关系数据库中查询优化的原理，理解查询优化算法在执行过程中的时间开销和空间开销，使用高级语言实现重要的查询优化算法。

**2. 实验环境**

Windows操作系统、MinGW编译器或Microsoft Visual C++编译器。C、C++，JAVA等语言均可。(可以用其他操作系统和编程环境，数据库尽量选择MySQL)

**3. 实验内容**

**3.1 实验任务**

任选下列查询语句中的三条，将其转化为对应的查询执行树，并且根据设计的查询优化算法，对生成的查询执行树进行优化。

SELECT [ ENAME = ’Mary’ & DNAME = ’Research’ ] ( EMPLOYEE JOIN DEPARTMENT )

PROJECTION [ BDATE ] ( SELECT [ ENAME = ’John’ & DNAME = ’ Research’ ] ( EMPLOYEE JOIN DEPARTMENT) )

SELECT [ ESSN = ’01’ ] ( PROJECTION [ ESSN, PNAME ] ( WORKS\_ON JOIN PROJECT ) )

PROJECTION [ ENAME ] ( SELECT [ SALARY < 3000 ] ( EMPLOYEE JOIN SELECT [ PNO = ’P1’ ] ( WORKS\_ON JOIN PROJECT ) )

PROJECTION [ DNAME, SALARY ] ( AVG [ SALARY ] ( SELECT [ DNAME = ’ Research’ ] ( EMPLOYEE JOIN DEPART MENT) )

**注意：**

**（1）在上述语句中，为了便于设计语法分析器，不同符号与关键词之间均以空格分隔。**

**（2）每人只用完成上述查询语句中的三条，根据完成情况给分。**

**（3）上述语句并非标准sql语句，而是本实验第一部分关系链接操作算法的函数抽象**

**3.2 实验说明**

1）需要设计一个语法分析器，能够识别上述关系代数语句，并且对其进行解析，生成对应的查询执行树，在检查时需要将树的结构显示出来，格式不限。

2）根据本课程所学的查询优化技术，对生成的查询执行树进行优化，并且将最后优化后的查询执行树输出至屏幕，格式不限。

**三、实验提交**

代码打包成压缩包“**实验2-学号-姓名.rar/zip**”，通过文件收集链接提交。