CMPE 321 : Introduction to Database Systems

Project 2

2017 - Spring

Database Management System Implementation

Oğuzhan Göller
2013400228

Contents

1 Introduction

This project is about implementing a simple Database Management System which we designed before. The DBMS is required to organize data in a convenient and efficient way and allow users to modify and delete current data in database. This project aims to teach fundamentals of creating a database.

2 Assumption & Constraints

- Page size should be 1024 Bytes.
- User input will always be valid according to criteria listed below.
- Each record will have a unique id and this id will be used as key.
- Record id will be automatically assigned by Database System.(new rule)
- Type name length can't exceed 20 characters.
- Each type will have a fixed number of 10 fields.
- Each type will have at least 1 field.
- Each field will have a fixed size of 10 bytes. (Changed from 12 bytes to 10 bytes)
- Field name length can't exceed 9 characters. The last free byte will be used as an exterminator while reading. (Changed from 12 char. to 9 char)
- User will use record id number to update or delete a record.
- User won't be able to do any changes on types except deleting it.
- Users can't update record's unique id.
- Type name will be changed to "NULL" if a user deletes it.
- Type name "NULL" coming from user will be invalid.

- Record header will have a size of 8 bytes for "class.txt" files.(new rule)
- Record header for "class.txt" files will contain the following elements(Order of elements are changed) :
  - an integer flag to indicate whether that record is Full - - (isFull) = by default '0' (flag is changed from bool to integer)
  - it's unique record id - - (recordId)
- Record header for "Syscat.txt" files will contain the following elements(new rule) :
  - an integer flag to indicate whether that record is Full - - (isFull) = by default '0' (flag is changed from bool to integer)
- Page header will have a fixed size of 24 bytes.
- Page header for will contain the following(Order is changed and maxNumberOfRecords removed) :
  - an integer flag to indicate whether this page is the last page - - (isLast) = by default '1' indicating it is the last page(4 bytes(changed from bool to int))
  - an integer address pointer to next page - - (next) = by defaul 'NULL' (4bytes)
  - free space left in that page —- (freeSpace) = by default '1012' —- integer=4 bytes
  - it's unique page id - - (pageId) integer-¿4 bytes
  - current number of records in that page(integer 4 bytes) - - (currentNoRecords) = by default '0'
  - fixed size of each record(integer-4 bytes) - - (recordSize) = by default '125 byte' for Syscat and '48 byte' for class.txt files

3 Data Structures

A page has a size of 1KB which is 1024 Bytes. It has a page header of size 24(changed from 12 bytes to 24) plus 8 rows(previously 7 rows) holding

metadata. A row consists of a record header of size 4 bytes(former - 2 bytes), type name info of size 21 bytes(former 20 bytes) and 10 columns of field info of size 12 bytes(former 10 bytes each). So in total each row has 10*10 + 21 + 4 = 125 bytes. It has 8 rows plus page header 125*8+24 = 1024 bytes. So 1024(former 1006) of 1024 bytes is used.
Page Design("Sys.cat") (1024 Bytes)

Page Header(24 Byte)					
Record Header1(4 Byte)	Type Name1(21 Byte)	Type1 Field 1(10 Byte)	Type1 Field 2(10 Byte)	...	Type1 Field 10(10 Byte)
Record Header2(4 Byte)	Type Name2(21 Byte)	Type2 Field 1(10 Byte)	Type2 Field 2(10 Byte)	...	Type2 Field 10(12 Byte)
Record Header3(4 Byte)	Type Name3(21 Byte)	Type3 Field 1(10 Byte)	Type3 Field 2(10 Byte)	...	Type3 Field 10(12 Byte)
...	...	...	...	...	...
Record Header7(4 Byte)	Type Name7(21 Byte)	Type7 Field 1(10 Byte)	Type7 Field 2(10 Byte)	...	Type7 Field 10(10 Byte)

Page Design(".dat Files")(1024 Byte)

Each page has 1024 bytes size. Each row has a header of size 8 bytes(former 2 bytes) and 10 columns of field info of size 4 byte(former 1 bytes) making a total of 48 bytes. And there is a page header at the beginning of each page of size 24 bytes(former 12 bytes). So (1024-24)/48 = 20. 20(former 84) is the number of records each page is able to have. There are 40 bytes of unused space for each page

Page Header(12 Byte)				
Record Header1(8 Byte)	Record1 Field 1(4 Byte)	Record1 Field 2(4 Byte)	...	Record1 Field 10(4 Byte)
Record Header2(8 Byte)	Record2 Field 1(4 Byte)	Record2 Field 2(4 Byte)	...	Record2 Field 10(4 Byte)
Record Header3(8 Byte)	Record3 Field 1(4 Byte)	Record3 Field 2(4 Byte)	...	Record3 Field 10(4 Byte)
...	...	...	...	...
Record Header84(8 Byte)	Record84 Field 1(4 Byte)	Record84 Field 2(4 Byte)	...	Record84 Field 10

Page Header(12 Byte)(Former)

Page Id(1 Byte)	Number Of Records(1 Byte)	isLast Flag(1 Byte)	Address of Next Page(4 Byte)	Free Space(1 Byte)	Record Size(4 Byte)

Page Header(24 Byte)(New Version)

isLast Flag(4 Byte)	Address of Next Page(4 Byte)	Free Space(4 Byte)	Page Id(4 Byte)	Number Of Records(4 Byte)	Record Size(4 Byte)

Record Header(2 Byte)(Former)

Record Id(1 Byte)	isEmpty Flag(1 Byte)

Record Header for Syscat(4 Byte)(New Version)

isFull Flag(4 Byte)

Record Header for Class.txt files(8 Byte)(New Version)

isFull Flag(4 Byte)	recordId(4 Byte)

4 Operations

There are two types of operation : DDL(type) and DML(record).

4.1 Type Operations

4.1.1 Create Type

```
open "Sys.cat";
page = first Page of "Sys.cat";
while page.isLast != true do
    page = page.next ;
end
get typeName, numberOfFields, fieldNames, from user;
create file "typeName.dat" ;
if page.freeSpace <recordSize then
    newPage ← create new Page ;
    add newType(keyId,typeName,numberOfFields,fieldNames) to
    newPage;
    newPage.isLast = true ;
    newPage.freeSpace = newPage.freeSpace - recordSize ;
    newPage.currentNoRecords++ ;
    page.isLast = false ;
end
else
    add newType(keyId,typeName,numberOfFields,fieldNames) to
    page;
    page.currentNoRecords++;
    page.freeSpace = page.freeSpace - recordSize ;
end
```

4.1.2 Delete Type

```
get typeName to be deleted from user;
open "Sys.cat" ;
found = search typeName among all type names in "Sys.cat" ;
if found then
    change typeName of that row to "NULL";
    page.freeSpace = page.freeSpace + recordSize ;
    page.currentNoRecords- -;
    delete file "typeName.dat" ;
end
else
    print error("Type name not found!)
end
```

4.1.3 List All Types

```
open "Sys.cat" ;
page ← First page of "Sys.cat" ;
whilepage.next != NULL i = page.CurrentNoRecords ;
while i >0 do
    if page[i].typeName != "NULL" then
        print(page[i].typeName) ;
    end
    i- -;
end
page = page.next ;
```

4.2 Record Operations

4.2.1 Create Record

```
open related ".dat" file of this type ;
get record fields from user ;
create newRecord(fields) with it's unique id;
page ← first page of ".dat" file ;
while record.isEmpty != true do
    record = page.nextRecord ;
    if record = lastRecordOfPage then
        page = page.next ;
    end
end
record.insert(newRecord);
record.isEmpty(false);
```

4.2.2 Delete Record

```
get typeName and recordId from user ;
open "typeName.dat" file ;
if "typeName.dat" contains recordId then
    record.isEmpty = true ;
end
else
    print error("Record not found") ;
end
```

4.2.3 Update Record

```
get typeName and id of that Record from user ;
open file "typeName.dat" ;
search for 'typeName' in file "typeName.dat" ;
get field numbers to be changed from user and their new values ;
update corresponding values in table ;
```

4.2.4 Search Record

```
get id(primary key) and typeName of record from user ;
found ← search for typeName in "typeName.dat" file ;
if found then
    address ← get address of that key ;
    go to address ;
    print("Related fields") ;
    for f : address.fields do
        print(f) ;
    end
end
else
    print("Record not found!")
end
```

4.2.5 List All Records

```
open file "Sys.cat" ;
foreach page in "Sys.cat" do
    get typeName from Page;
    if typeName == "NULL" then
        continue ;
    end
    open file "typeName.dat" ;
    foreach record in page do
        if record.isEmpty then
            continue ;
        end
        print(record.Info) ;
    end
end
```

5 Conclusions & Assessment

At the new version of my project, the page size is the same as my design, which is 1024 bytes. However, i made a lot of changes concerning sizes of page and record headers, its components and record size. I decided to make page header size equal to 24 bytes to make my job easier and my code more understandable. This header consists of all integers, so that it can hold a lot bigger numbers compared to my previous design, where there were bytes for some components.

I enlarged record size from 12 to 125. This great change is due to changing record fields from bytes to integers. Because a byte can hold values from -127 to +127 where the integer range is about 2 billion. So in order to enable users greater flexibility of records, i made this change, which in return dropped record size per page from 84 to 20, causing system to slow down.

In my previous design, i used only one record header for both "Syscat" and class files. At this new version, i used two different record headers; because record header was necessary for using as a key for class.txt files. But for Sys.cat, i didn't need an id since it wasn't necessary.

Because of the reason that my record size was 48 bytes, i had 48*20+24 = 984 of total space occupied in each class.txt file page. So there is 40 bytes of unused space in each class.txt file page, which is a disadvantage for my system.

I also assumed that there are constant number of 10 fields for each type, which is also a disadvantage of memory usage, since user may define less than 10 fields and the rest will be unused. But it is advantageous in case that we know exact locations of each field in memory, we can directly and easily access it.