BOĞAZİÇİ UNIVERSITY

CMPE 321 PROJECT 2

Spring 2019

STORAGE MANAGER SYSTEM

Yağmur Kahyaoğlu

2015400057

April 7, 2019

1 Introduction

In this project we were assigned to implement a Storage Manager System that can do basic DDL(create, delete, list) and DML(create, delete, list, search) operations. For the project I have designed a manager in which the types have thier own files. The document consists of assumptions and constraints regarding the design, explanations of the storage structures used and the pseudo code algorithms for the DDL and DML operations.

2 Assumptions & Constraints

- The system catalog consists of one file named "SystemCat.txt".
- Pages in system catalog contains list of record types.
- When a record is deleted in System Catalog the byte that keeps deletion information becomes 1.
- Pages in data files contain at most one type of record.
- A page is at most 1786 bytes.
- A data file is at most 17890 bytes.
- A data file can contain at most 10 pages. When the current file is full, a new file is created.
- A page in a data file contains at most 20 records.
- Records are inserted into a file according to their types.
- Number of fields is given when creating a record and it can not be more than 10. If it is less than 10, the space will be allocated for the remaining field areas but will be kept null.
- Every character is 1 byte.
- Names for the record fields are at most 12 characters.
- Values for the record fields are at most 8 characters.
- A record can not have fields with the same names.
- Record deletions are done not physically but logically. The byte that keeps deletion information of the record that is deleted becomes 1.

- Page ids are non-negative integers.
- When the all records of a page are deleted, the byte that keeps deletion information becomes 1.
- When the all pages of a file are deleted, the byte that keeps deletion information becomes 1.
- When a type is deleted, the files of that type are deleted physically.
- When a type is created, a file or that type with a file header named typename_1.txt is created.
- When a data file is full a new file for the type is created with the name typename_fileid+1.txt.
- Records of the same type cannot have the same primary key.

3 Storage Structures

3.1 System Catalog

System Catalog is a file named ”SystemCat.txt” that stores the metadata. The file stores list of record types. For every record 14 bytes that are reserved for the record header; 12 are reserved for the type name, 1 byte that keeps deletion information and 1 byte is for the number of fields of the type. 12 bytes are reserved for each 10 field names of that type. Hence a record size is 134 bytes.

TypeName1	isDeleted	NumberOfFields	FieldName1	...	FieldName10
TypeName2	isDeleted	NumberOfFields	FieldName1	...	FieldName10
...	...	...	...	...	...

Table 1: Example of records from the System Catalog

3.2 Data File

Data files are files where the actual data is stored. The files store pages that consists of a page header and records. The page header has 4 bytes reserved for page id , 1 byte that keeps deletion information and 1 byte for the number of records that page stores. Each record has 9 bytes reserved for the record header that contains 8 bytes for record id and 1 byte that stores deletion information and 8 bytes for every 10 fields. The total size of the page is 1786 bytes hence a page contains 20 records. The total size of the file is 17890 so the file contains at most 10 pages.

File ID	Is Deleted	Type Name	Name of Next File	
Page ID			Is Deleted	Number of Records
RecordID1	isDeleted	Field1	...	Field10
RecordID2	isDeleted	Field1	...	Field10
...	...	...	...	...
RecordID25	isDeleted	Field1	...	Field10

Table 2: Example page of a data file and the file header

4 Operations

4.1 DDL Operations

The system will allow us to make some operations regarding the System Catalog.

4.1.1 Create Type

Algorithm 1: Create Type

```
declare type
typeName ← input
numOfFields ← input
type.push(typeName, numOfFields)
for 0 to numOfFields do
    fieldName ← input
    type.push(fieldName)
end
for numOfFields to 10 do
    type.push(NULL)
end
file ← open("SystemCat.txt")
file.push(type)
openNewFile(type)
```

4.1.2 Delete Type

Algorithm 2: Delete Type

```
typeName ← input
file ← open("SystemCat.txt")
file.find(type).delete();
findFilesWithTypeName(type).delete();
```

4.1.3 List types

Algorithm 3: List Types

```
file ← open(”SystemCat.txt”)
foreach record in page do
    if record.typeName is not NULL then
        print(record.typeName)
    end
end
```

4.2 DML Operations

The system will allow us to make some operations regarding the data files.

4.2.1 Create Record

Algorithm 4: Create Record

```
declare record
record.typeName ← input
record.recordId ← input
numOfFields ← getNumOfFields(record.typeName,
  "SystemCat.txt")
for 0 to numOfFields do
    field ← input
    record.push(field)
end
if currentFile.isFull()) then
    currentFile ← newFile(record.typeName)
end
if currentPage.isFull() then
    page ← newPage() page.push(record)
    page.numOfRecords += 1
end
```

4.2.2 Delete Record

Algorithm 5: Delete Record

```
typeName ← input
recordId ← input
file ← getFileWithRecord(typeName, RecordId)
record ← file.getRecord(record.recordId)
record.isDeleted = 1
page.numOfRecords -= 1
if page.numOfRecords = 0 then
    page.delete
end
if currentFile.isEmpty then
    currentFile.delete
end
```

4.2.3 Search For Record

Algorithm 6: Search For Record

```
typeName ← input
recordId ← input
file ← getFileWithRecord(typeName, RecordId)
record ← file.getRecord(record.recordId)
return record
```

4.2.4 Update Record

Algorithm 7: Update Record

```
typeName ← input
recordId ← input
fields ← input
file ← getFileWithRecord(typeName, RecordId)
record ← file.getRecord(record.recordId)
record.update(fields)
return record
```

4.2.5 List All Records of Type

Algorithm 8: List All Records of Type

```
declare recordsWithType
typeName ← input
foreach page with typeName do
    foreach record in page do
        if record.isNotNull or record.recordId is not -1 then
            recordsWithType.push(record)
        end
    end
end
return records
```

5 Conclusion

In this project where the goal was to design a Storage Manager System I have implemented a system using Java in which the System Catalog stores types of records as a list and the data files stores the records using the paging system. The data files contains at most one type of record. The objective here is to keep the records with the same type together so it is easier and faster to search, delete or update any of the records.