BOGAZICI UNIVERSITY

CMPE321 - INTRODUCTION TO DATABASE SYSTEMS - SPRING 2019

Project 2 - Storage Manager Implementation

Halit Ozsoy - 2016400141

April 8, 2019

CONTENTS

1 Introduction

The objective of this project to design a storage manager for a database management system. The storage manager should support both DDL Operations (Create/Delete/List type) and DML Operations (Create/Delete/Search/List record). In this project, decisions relating to construction criteria (Page size, file size, etc.), assumptions about the database, definitions of the Structures and Operations of the database are made and documented.

2 Assumptions & Constraints

- There are 3 types of files: System Catalog File, Data-Table File, and Index File. Details of these files are explained in their own subsections.

2.1 System Catalog File

- There is only one system catalog file, named *SystemCatalog.txt*.
- Below assumptions within this subsection are about the *system catalog file*
- There's a 4 byte header in the file indicating the number of tables.
- There are multiple pages in that file.
- Each page corresponds to one table's meta data and there's one page for each table.
- Each table (type) has a unique name.

2.1.1 Page of a System Catalog File

  - Page size is 1KB = 1024 bytes.
  - Represents a table's meta data
  - Has a unique *table name*
  - That name is always 9 bytes (may be padded with empty strings to make it shorter).
  - That name is an ASCII string consisting of alphanumeric characters, underscore, padded with empty strings (byte, 00 in hex) in the end.
  - If the first byte of the name is 0, then, the table is deleted.
  - Page Header (1bytes in total) and the number of fields in the table (1 byte)
  - Each field within a table has a unique name.
  - There is at least one field in the table.
  - After the page header, there are N (number of fields, obtained from the page header) blocks of 9 bytes each representing a field name padded with 0s in the end.
  - Each field's size is 8 bytes (long long int).
  - Similarly to the table name, field names are also ASCII strings consisting of alphanumeric characters, underscore, padded with empty strings in the end. Also, field names are 9 byte each.
  - There can be at most 6 fields in a table.
  - The first field is the primary key field of the table.
  - There will not be an operation to change the fields of a table.

2.2 Data File

- There is one data file for each table (type).
- Below assumptions within this subsection are for each such data file.
- It is named, *<table-name>.table.txt*
- There are multiple pages in it.
- Each page corresponds to some portion of the records within that table.
- There is a table header in the beginning of the file. That header is always 4 bytes in total, where 4 bytes are the number of pages in that file.
- Each page within a file, has a unique page id starting from 1.
- Each record within a file, has a unique primary key.

2.2.1 Page of a Data File

- Page size is 1KB = 1024 bytes.
- Has a unique *page id*
- That id is a 4 byte unsigned number.
- Page header (1bytes in total) consists of the number of records in the page (1 byte).
- After that, there are N (number of records, obtained from the page header) fixed sized records of size M (summation of the field sizes of the table + 1, obtained from the system catalog).

2.2.2 Record of a Data File

- Each record starts with a 1 byte record header denoting whether or not it is deleted, then, the fixed sized primary key field of it (size is obtained from the system catalog). The rest of the record is the rest of the record's fields.
- Each record is at least of 9 bytes, (1 for record header, 8 byte for the only field, primary key of size 8).

2.3 Index File

- There is one index file for each table (type).
- Below assumptions within this subsection are for each such index file.
- It is named, *<table-name>.index.txt*
- There is an index header in the beginning of the file. That header is always 4 bytes in total, meaning the number of pages in the respective table file, call that S.
- Call the size of the primary key field for the respective table 8bytes (fixed)

- The rest of the file consists of S indexes each 12 bytes.
- Each such index starts with 8 bytes representing a primary key, and ends with 4 bytes representing the page number in which the record with that primary key is located.

2.4 USER INPUT & INTERACTION

- It is assumed that the user always enters valid input.
- Thus, there's no need to verify user input.
- Verification for whether a record already did exist, whether a type already did exist, will be skipped.
- However, it can easily be shown that such verification can be done via using calls to other operations: *List all type* & *Search by primary key*.

3 Storage Structures

3.1 System Catalog File

System Catalog File							
	System Catalog Pages						
File Header	System Catalog Record (64 bytes)						...
	System Catalog Record Header (10 bytes)			Fields (54 bytes)			
Page Count	Deletion Status	Table Name	Field Count	Field (40 bytes)		...	
				Field Size	Field Name		
4 bytes	TableName's first byte	9 bytes	1 bytes	0 (fixed size)	9 bytes		

3.1.1 System Catalog Page

System Catalog Record (64 Bytes)	
Header	Fields (each 9)
10 bytes	54 Bytes

3.1.2 System Catalog Page Header

System Catalog Record Header (48 Bytes)		
Deletion Status	Table Name	Field Count
First byte of table name	9 bytes	1 byte

3.1.3 System Catalog Page Fields

System Catalog Page Field (40 Bytes)	
Field Size	Field Name
1 byte	39 bytes

3.2 Data File

Data File										
Data File Header (4 bytes)			Data File Pages							
			Data File Page (1024 bytes)							...
Deletion Status	Table Name	Page Count	Page Header (1 bytes)			Records (1023 bytes)				
			Deletion Status	Page ID	Record Count	Record (9-49 bytes)			...	
						Record Header (1 byte)	Fields (8 bytes each)			
						Deletion Status	Primary Key Field	...		
removed	removed	4 bytes	removed	removed	1 byte	1 byte	8 bytes			

3.2.1 Data File Page

Data File Page (1024 Bytes)	
Page Header	Records
1 bytes	1023 bytes

3.2.2 Data File Page Header

Data File Page Header (1 Bytes)		
Deletion Status	Page ID	Record Count
removed	removed	1 bytes

3.2.3 Data File Page Record

Record (9-49 Bytes)				
Record Header	Fields (8-48 bytes)			
Deletion Status	Primary Key Field	2nd Field	3rd Field	... N^{th} Field
1 bytes	8 bytes	8 bytes	8 bytes	... 8 bytes

3.3 Index File

Index File			
Index File Header (4 bytes)	Indexes		
Index Count	Index (12 bytes)		...
	Primary Key	Page ID	
4 bytes	8 bytes	4 bytes	

4 Conclusions & Assessment

- If the number of records in one table grow very big, then, the index file for that table will start to grow big too. So, reading the index file at once would become a bottleneck for Search by Primary Key Operation.
- In order to prevent such situation, index files can be paginated too, using last 4 bytes of the hash values obtained by hashing the primary keys of records.
- By doing some real world testing and analysis, the limitations for chosen for field size & names, page sizes, record sizes, can be optimized further more.
- Removed all complex headers, flags, etc for implementation.
- Implemented all the functionality explained above, yet failed to make it run successfully with C++ binary file readers.