Storage Management System Design

# Introduction

The purpose of the project is designing a basic storage management system. The system owns DDL and DML operations to manage and manipulate the data. It assures users to be in contact with the database with appropriately handled functions and ordinately stored data files and pages in it. But in this project, we are not defining a disk manager that will retrieves the pages we stored the data in it. We are doing our operations through a disk manager already defined and served us. There are types and instances of them to store data. Every single type is stored in one different file and files have multiple pages containing records of instances. They all together constructs the database.

To accomplish the project purpose, we have main and basic structures. In later chapters of this document they will be analyzed deeply enough to clarify what they do and how they do but we need to briefly explain them in here to give a main idea of structures and operations to the reader.

Firstly, we have System Catalogue named SytemCatalogue.txt in the program that holds the main parameters and types that defined. DDL and DML operations firstly need to visit the System Catalogue.

Afterwards, we have Page Header and Record Header to describe properties of the pages and records respectively. These structures allow us to reach to the correct datum. And operations use them as a guide.

Lastly, program have operations gathered in two main headers as DDL and DML. These operations will serve to user to contact with the data.

#### DDL:

* Create a type
* Delete a type
* List all types

#### DML:

* Create a record
* Delete a record
* Search for a record (by primary key)
* List all record of a type

# Assumptions & Constraints

To handle the operations we mentioned earlier we need to make some assumptions and constraints to ease the program service burdens in the basic storage management system(SMS) project. Also these approaches will clarify our main goals and processes. Apart from these, some assumptions and constrains must be determined to design a SMS.

#### Assumptions

* User always enters a valid input to every function with appropriate order.
* All fields are integers and all type and field names are string.
* Every type is represented with only one key and the key is the first field of the type.
* A disk manager already exists and serves us.

#### Constraints

* The data organized in pages and pages contain records.
* Every single file contains one type. And every file contains multiple pages.
* The program reads page by page the files to don’t load whole file to RAM.
* Page size is 2KB.
* Max number of fields a type can have is 5.
* Max length of a type name is 10
* Max length of a field name is 10

# Storage Structures

#### System Catalogue

System Catalogue is used to store the schemas at all levels of the Database. In this basic Storage Management System, we only have type schemas and some necessary parameters. System Catalogue stored in SystemCatalogue.txt. First line of SystemCatalogue.txt has four values. First one stores the number of Types we declared earlier. It is for searching the catalogue without any problem. Second value stores the max length of type name which is assigned to 10.Third value stores the max length of a field name which is assigned to 10 again. Fourth value stands for the max number of field that a type can store. It is assigned to 5. Last three assignment have done under our constraints we examined earlier. But first value changes with the created types. This process is handled by DDL operations.

After the first line, our catalogue has a Type value in every line. Properties of types are stored in such a way: First value stores the name of the Type which is uniquely declared. Second value stores the address of first page of a File that stores the records of this type. This value allows us to go and search, add, or delete a record. Third value is for learning whether this type is declared but deleted afterwards or still in usage. It is a boolean value. Fourth value is the key name of the Type. It is stored to give a user the fields and values while highlighting the key value with the key name. Fifth value is an array of strings that stores the field names. It’s length can be less then maxFields which is five. When a Type is added, the program stores a new line of appropriate values to access it later. But deletion operation doesn’t erase a line. Only changes the values of the line.

Graphical representation of the System Catalogue shown below, in figure 1.

A screenshot of a cell phone

Description automatically generatedfigure 1

#### Page Design

Every page stores the records of a specific Type. It has Page Header part and a part that stores the records. First line is reserved for the Page Header values. After that line, every line represents a record. Page design represented under the Page Header topic, in figure 2.

#### Page Header

Page Header is a line with values on it in the first line of every page. First value is a pointer and points to the next page of belonged file. With this pointer we are making a linked list between the pages of a same file. Type stores the first page of a file, and every page stores the next page address. With these pointers, while we are searching a value, for example, we get rid of the necessity of loading whole file to the RAM and be able to read a page at a time. Second value represents whether this page is full or not. The program uses this value when adding a new record operation. If the page is full already, the program doesn’t traverse the page and goes to the next page using the first value of the Page Header line. Third value stands for the number of records this page has. It is also used in searching operations. Fourth value is the size of page. When a new record is declared, this value increases with the amount of the record size. After this, the program looks for the size of page and decides whether the page is full thereafter or not. And this decision may change the is full parameter’s value.

Representation of a Page Design and Page header shown below, in figure 2.

A screenshot of a cell phone

Descriptiasdason automatically generated

figure 2

#### Record Header

Every record, in a page, has record header which contains two values. After these values every record stores field data. First value of a Record Header is a Boolean value to show whether this record is deleted or not. It is used searching a record like before in the searching a type. Second value is the size of the record. It is calculated when the record declared and used to increase the belonged page’s size.

Representation of the Record Header shown below, in figure 3.

A screenshot of a cell phone

Description automatically generated

figure 3

# Operations

#### DDL Operations

DDL is the Data Definition Language and it is for specifying the Database. Our basic Database have types and nothing more. So, our DDL is for only declaring and organizing the types. These operations is listed and clarified below.

* Create a type

This operation handled with CreateType function. The function uses three parameters named typeName(string), fieldNames(array of strings), keyName(string). Firstly, it checks the correctness of the typeName using System Catalogue. After that, searchs a type named typeName to controls if it is declared earlier. If it is declared but deleted after, the program should override the values given to the old ones. If it is not declared earlier we need to create a new line with appropriate values and increment the number of types in the System Catalogue. Pseudocode of the algorithm is given below. Searching algorithm didn’t defined below because it is a simple for loop.

A screenshot of a social media post

Description automatically generated

* Delete a type

This operation handled with DeleteType function. The function uses only one parameter named typeName(string). Firstly, it checks the correctness of the typeName using System Catalogue. After that, searchs a type named typeName to controls if it is declared earlier. If it is declared but deleted after or it’s not declared the algorithm gives an error message. When it finds the type, it changes the values in the System Catalogue. Pseudocode of the algorithm is given below. Searching algorithm didn’t defined below because it is a simple for loop.

A screenshot of a social media post

Description automatically generated

* List all types

This operation handled with ListAllTypes function. The function doesn’t use any parameters. The algorithm gets the total type number from System Catalogue to search correct value of lines in System Catalogue. Every time looks for a line and gets the values from there. If the Type declared earlier isn’t deleted now, it will be listed. So, the algorithm gets the values from the line and sends it to the user. Pseudocode of the algorithm is given below.

A screenshot of a cell phone

Description automatically generated

#### DML Operations

DML is the Data Manipulation Language. Our data stored as records which are instances of Types. In our basic Storage Management System, DML stands for manipulating the records. Every record is a line in page. Every page stores records belong to one type as mentioned earlier. Needed operations listed below.

* Create a record

This operation handled with CreateRecord function. The function uses two parameters named typeName(string), and fieldValues(array of ints). First value of the fieldValues is the key value. The algorithm firstly searchs for the Type named typeName. If there is such a type isn’t deleted, the program gets the address of the file that stores these instances of the Type using System Catalogue. It searches not full page to add a record. Firstly looks to the first page and if is full goes to the next page using address pointer of the Page Header. When it founds a appropriate space it adds a Record with appropriate values. Also changes the Page Header’s size with the calculation of record size. If the page fills up, it creates a new page and connects to the current page. Pseudocode given below.

A screenshot of a social media post

Description automatically generated

A screenshot of a cell phone

Description automatically generated

* Delete a record

This operation handled with DeleteRecord function. The function uses two parameters named typeName(string), and keyValue(int). The algorithm firstly searchs for the Type named typeName. If there is such a type isn’t deleted, the program gets the address of the file that stores these instances of the Type using System Catalogue. It traverses pages one by one until finding record with key value same with the keyValue. Because we are assuming user always gives valid input, we also assume the program will find a record with the keyValue. After finding the record, the program changes its values. Pseudocode given below.

A screenshot of a social media post

Description automatically generated

* Search a record

This operation handled with SearchRecord function. The function uses two parameters named typeName(string), and keyValue(int). The algorithm firstly searchs for the Type named typeName. If there is such a type isn’t deleted, the program gets the address of the file that stores these instances of the Type using System Catalogue. It traverses pages one by one until finding record with key value same with the keyValue. Because we are assuming user always gives valid input, we also assume the program will find a record with the keyValue. The algorithm very similar with we used before in DeleteRecord function. This algorithm only differs in not changing the values of the record but sending them to user with I/O System. Pseudocode given below.

A screenshot of a social media post

Description automatically generated

* List all records of a type

This operation handled with ListAllRecordsofaType function. The function uses one parameter named typeName(string). The algorithm firstly searchs for the Type named typeName. If there is such a type isn’t deleted, the program gets the address of the file that stores these instances of the Type using System Catalogue. After that, looks one by one to the every record. Again checks pages one by one using pointers on Page Headers. Retrieves lines as record objects and looks the isDeleted value on them. If the record isn’t deleted the algorithm sends them to the user with I/O System. Pseudocode given below.

A screenshot of a social media post

Description automatically generated

# Readme

To run the code given, first of all your system needs to have python. After that, be sure that you have project.py and SystemCatalogue.txt in your system.

You can run the code with two different options. First you can open the code with some IDE and edit the configurations as follows:

*operationName argumants(in ordered and space between them)*

Example usage:

listAllRecords Student

createType Student id age department gpa

Secondly you can use terminal. Go to the directory that stores the project.py and SystemCatalogue.txt. Then type:

py project.py *operationName argumants(in ordered and space between them)*

Example usage:

py project.py listAllRecords Student

py project.py createType Student id age department gpa

# Conclusion & Assessment

Designed system looks fine. It uses the RAM tidily, Stores data even if it is deleted. In some points, we may use this knowledge. Disk Manager used clearly. But in some points search algorithms may be enhanced and may be used in other algorithms as a part of them although it seems appropriate for the basic Storage Management System.