Database Programming

SAIF KHATATBEH_18110004

Table of Contents

I. Introduction:

I have been recently assigned by my company to re-visit the database that was previously built for Miles Hypermarkets and significantly modifying it by creating a Database Management System (DBMS). Where me and my team will be creating and updating the DBMS components, manage the data exchange between them, and optimize their functionality using available state-of-the-art methods.

This report will talk about the process that went in when designing and implementing the DBMS in Miles Hypermarket. Where it will talk about the architecture of the DBMS used and the system and user requirements of Miles Hypermarket to build the system, then the development of the program along with its SQL queries, then an implementation and explanation of the principles of concurrency control, and finally using modern methods to wrap the developed DBMS components along with a test plan to test the system functionality.

II. Part 1: DBMS Components and their Functions:

This section of the report will talk about the DBMS architecture that is going to be used in Miles Hypermarket, then a deeper dive on the main problem that caused to build the system along with Miles requirements to build the system (user requirements) and the requirements needed to build what Miles had required (system requirements), then diagrams to further understand how the system works and how the data is moving inside of the system, and finally and evaluation of how the system was effective depending on Miles requirements.

A. DBMS Requested Architecture:

Many companies create their own DBMS architecture to fit their needs and requirements. For that, the team leader requested the build of this architecture (Figure 1) in which it would fit the needs and requirements of Miles Hypermarket.

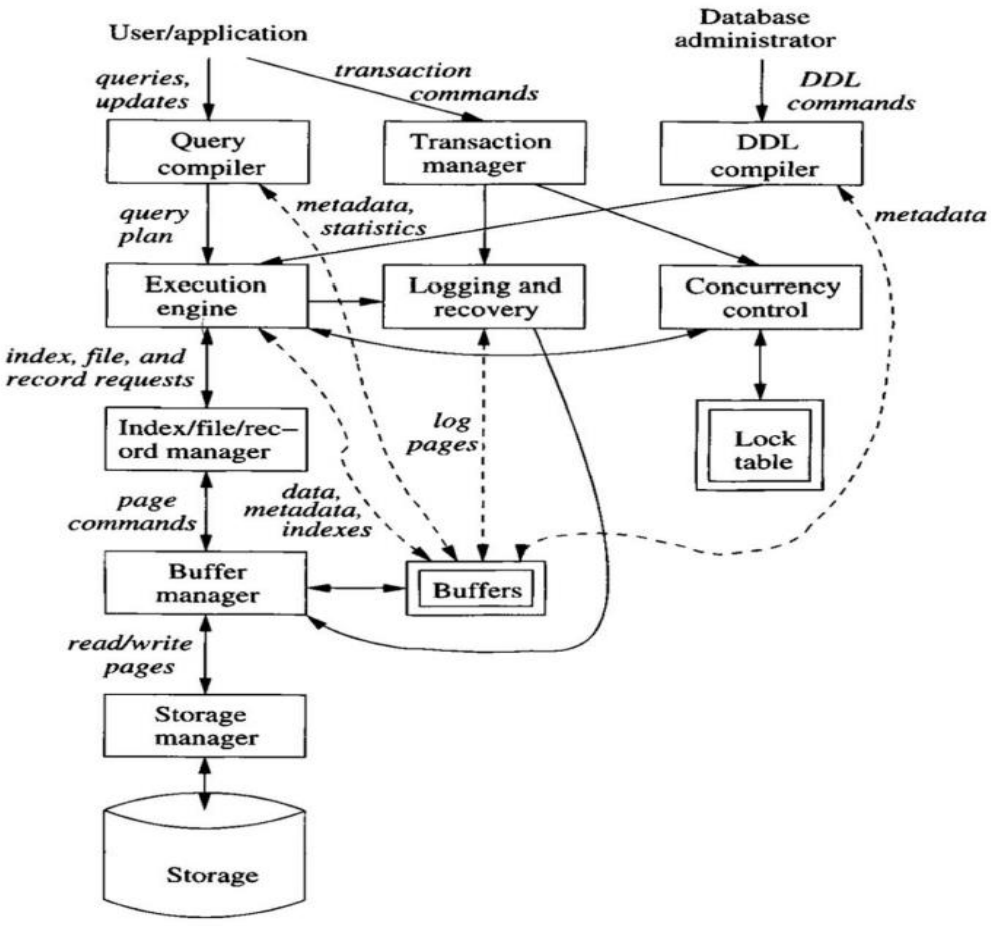


Figure 1: DBMS Architecture

This architecture would be perfect to implement for Miles DBMS because it shows how the user would interact with the system and how the database administrator would interact as well. This architecture uses any subcomponent, and these subcomponents interact with each other. The subcomponents that will be used to implement the Miles DBMS using this architecture are:

1. Storage:

The Storage subcomponent in this architecture is a structure that is used for holding data. They are usually data structures that are designed specifically to hold data that then can be further be manipulated by the other subcomponents in the architecture.

There are multiple of way to implement a storage in the DBMS, but the one that will be used in the Miles program is the Circular Linked List data structure. Linked Lists are a data structure that stores data in nodes and then by using pointer these data can be accessed. Usually Linked lists are linear, meaning the data after the end of the node is NULL, but with the circular linked list all the nodes are connected to each other forming a circle, meaning there is no NULL after the end node (geeksforgeeks, 2021).

2. Storage Manager:

The storage manager subcomponent in this architecture is the subcomponent that manages the data that are held in the storage subcomponent. Managing the storage means handling the data in the storage by performing CRUD (Create, Read, Update, Delete) operations, which means that the storage manager can insert data into the storage, delete data from the storage, update data from the storage, and finally can retrieve data from the storage.

There are multiple of way to implement a storage manager, but for this program, the storage manager will be implemented using C programming language due to its simplicity and its ability to implement CRUD operations using functions that can be created using C language.

3. Buffer Manager:

Buffer manager in this architecture is a component that is used to create and allocate memory blocks with a certain size in the memory temporarily that the buffer can then use.

There are multiple of way to implement a buffer manager, but the one that will be used in this system is by using malloc in C language, which is responsible to create memory blocks using the size of the data structure that contains the data in the storage.

4. Buffer:

Buffer, in general, is a temporary storage area in the memory where data can be passed into it and then these data are put on hold in the buffer until its freed or being using in the program (Priya, 2021). For this architecture the buffer will hold the data that has been allocated in the memory by the buffer manager then these data can be used to perform multiple operations such as prompting all the data to the user or updating these data or deleting this data without affecting the main storage component. If the data is no longer useful it can be freed from the memory to allow more data to be saved in the buffer

There are multiple of way to implement a buffer, but for this system, the buffer will be implanted using the malloc (memory allocation) in C language, where malloc is memory allocation technique used to save data temporary in the memory and can be freed using the free command.

5. Index/File/Record Manager:

Index/File/Record Manager in this architecture is a component used to search for data in the memory by using unique keys.

This component will be implemented in this system using the Linked List data structure and by defining unique keys in the node of the linked list, where the program will search for the key and then return the index (pointer) of the certain data that contains that key.

6. Execution Engine:

The Execution Engine in this architecture is a component that will take queries then process it and then translate it in an environment where it can be executed.

The programming technique that will be used to implement execution engine is the C compiler (GCC compiler) since it will take compile the queries and then translate them into an action that will be prompted to the user.

7. Query Compiler:

Query Compiler in this architecture is a component that translates queries that user want to use into executable actions, these queries can be storing, deleting, modifying, reading, and retrieving data.

The programming techniques that will be used to implement query compiler are the functions creating in C language, where each function would be called depending on the user's query.

8. Concurrency Control:

Concurrency Control in this architecture is a component that helps in the execution of multiple of process without conflicting each other (Priya, 2021).

The programming techniques that will be used to implement this component in the system are going to be the use of threads in C by taking advantage of the threading library called pthread.

9. Logging and Recovery:

The logging and recovery component in this architecture is a component that creates backups from the data that is stored in the storage component and then can be recovered by importing the backup back into the storage component.

There are multiple of ways to implement this component, one of the most common is from the C programming language, where there is a programming technique to export the data in a binary file from C using fwrite command. where then it can be recovered by opening the file using the fopen command and then reading the binary using fread command to read the binary file and save it to the buffer then back to the main storage component.

B. Miles Problem Statement and DBMS Requirements and Design:

Miles Hypermarket was a previous client of our company where years ago we were asked to create a database for their stores using MySQL RDBMS (Relational Database Management System). The problem is that they now have a lot of suppliers that aren't being recorded in the database, and as they are now a big hypermarket store, they want to branch from MySQL and want to have their own database management system for recording and handling their supplier chains.

1. Miles Requirements for the DBMS (user requirements):

Here are the user requirements for Miles DBMS:

The first requirement is that their must be a menu where the users can see all the operations that can be done on the DBMS, which are inserting products, updating the products information, deleting the product from the system, and finally displaying the products in the system.

The second requirement is that the users must have the ability to enter the products that are coming from the supplier chain into the system, where the data that are required to be entered are the ID of the product, the name of the product, the data of the arrival of the product, the category of the product (for example if the product was milk then its category is Dairy), the name of the supplier of the product, the country for which the product came from, and finally the quantity provided by the supplier of this product.

The third requirement is that the user must have the ability to search for the products based on their ID's so that they can get its information or can perform updates on the information or can delete the product from the system

The fourth requirement is that the user can modify the products information that are in the system, in the case of the misinput of the information. All the information can be modified except for the ID as it would create confusion when searching for the product and the ID that would be modified could violate their naming standards.

The fifth requirement is that the user must have the ability to delete any product in the system because this product may not of use to them anymore and would be a wasted space for other important products or products that have ran out of stock.

As Miles want to have a better understanding of their products and supplier performance and demand to create correct business decisions, this where the sixth requirement comes as the user must have the ability to search and display the products information, where the searches can vary from searching and displaying a specific product to searching and displaying all the products based on the information entered in the system. By searching for products based on the information entered means that there must be searches to display all products with the same name or products that have arrived on the same date, or products of the same category, or products that are brought by the same supplier, or products that are from the same country, or products that have the same quantity.

2. Miles DBMS System Requirements and Design:

System Requirements are requirements needed to build the system on what the user needs (user requirements). Based on the architecture proposed and the user requirements, the DBMS that

will be built will be a simple one to handle CRUD and Concurrent operation. So, the fitting programming framework will be C and since the user did not give any mention on the user interface other than the menu, the user can run the system on the black screen (terminal). Here are the system requirements and the design for Miles DBMS.

Menus:

There shall be a menu that is going to be developed in C and run through the terminal, where the user must input their choice based on the number of the operation. Then the choice that the user inputs will be mapped to the operation in the menu (insert products, delete products, update products, read products). The operations on the menu are queries that the user would do using the CRUD operations. There shall also be another menu specifically designed for the read operation where it will have two operations, displaying a specific product (specific search) or displaying products based on their name, date of arrival, supplier name, supplier country, quantity available (broad search), where the broad search would have another menu to get products based on the number of the query.

This requirement will be handled by the query complier and execution engine, since it will translate the queries that are written on the menu to an action that the C program can execute.

Adding new products into the system:

The user must have the ability to add the products received from the supplier to the system by using a circular linked list to store the data. This would be done by allocating a memory block from the size of the linked list then getting from the user the ID of the product along with the name of the product, data of arrival, supplier name, supplier country, and quantity available to store them in a buffer. There must be a check before storing the data in the linked list where if the linked list doesn't exist then create a new one and store the data directly, but if the linked list already exists then check if the ID exists in the linked list, if not then store it in the linked. However, if the ID exist prompt the user with an error message telling them that the ID is already present and free the data from the memory. Once the data is stored in the linked list, the last node inserted will become the head node since it's a circular linked list. The circular linked list was used to utilize the ability to point to the previous and next nodes, which would significantly help in searching for the products.

This requirement will be handled by multiple of components in the DBMS architecture, where the allocation of a memory block will be handled by the buffer manager, the temporary data saved in the buffer will be handled by the buffer component, the search of the ID will be handled by the storage manager where data from the storage will be retrieved by the storage manager from the storage and then compared with the data input in the buffer to check if the ID exists which this operation would be handled in the index/file/record manager. Then the creation and storage to the linked list will be handled by the storage component.

Searching for a specific product:

The user must have the ability to search for a specific product using the product ID. This happens when the user enters the ID of the product, they want to search then the products IDs in the linked list will be compared with the ID that the user entered, and if it found return the current position (current pointer) of the ID that has the product information. The search starts with pointing to the next node in the linked list as the head is the last node, because it's a circular

linked list, then the program will keep pointing to the next node until it reaches the head node (last node) to return a response whether the ID was found or not. The search process is essential since it will be also used in the update and delete requirements.

This requirement was handled by the index/record/file manager component as it uses the linked list pointers to find the location (index) of the specific product and then return it back to the main process to be further used or displayed to the user.

Updating Product Information:

The user must have the ability to update any product information that is present in the system. This is done by first searching for the ID of the product they want to modify, if the ID wasn't the user will be prompted with an error message indicating that the ID entered doesn't exist. However, if it was found, then display the product and prompt the user with the menu that has all the product information that the user wants to modify.

This process is handled by index/record/file manager component to find the product and the storage manager to modify the information in the linked list (Storage).

Deleting Product from the System:

The user must have the ability to delete any product information that is present in the system. This is done by first searching for the ID of the product they want to delete, if the ID wasn't the user will be prompted with an error message indicating that the ID entered doesn't exist. However, if it was found, then create a thread for this delete process and keep creating threads if the user wants to continue to delete data from the linked list. Once the user is done, then join the threads created with the main process and display to the user the remaining data.

This process is handled by index/record/file manager component to find the product and the storage manager to delete the information in the linked list (Storage) and finally concurrent component to create multiple of threads for each deleting process.

Reading Products in the System:

The user must have the ability to read the products that are stored in the system based on the ID or the information of the product. Meaning the user must have the ability to search for a specific product or to search for multiple of products within a certain query. This done by prompting the user with a menu to choose whether to display a specific product or to search for multiple of products. If the user chose to search for multiple of products, another menu would be prompted that have all the available search queries that the user can make and when the user selects an option, for example to search for all the products with the same category, then the user will be prompted to enter the name of the category that they want to search, then input from the user will shall be compared using string comparing methods to see if the input matches with the data in the linked list, if it doesn't, an error will be prompted. However, if it exists, then display the products to the user.

This process is handled by Query Compiler and Execution engine to translate the query to action then by index/record/file manager component to find the products and return the indexes of the products to the storage manager to display the data from the storage component.

C. Evaluating the Data Flow Process between Different DBMS Components with Flow Chart of the Design:

After specifying the user and system requirements along with creating the initial design of Miles DBMS, now comes the evaluation of each process that would be implemented to Miles DBMS and how the process would look like when a user use the DBMS.

1. Inserting Process:

Here is the data movement when inserting products into the system:

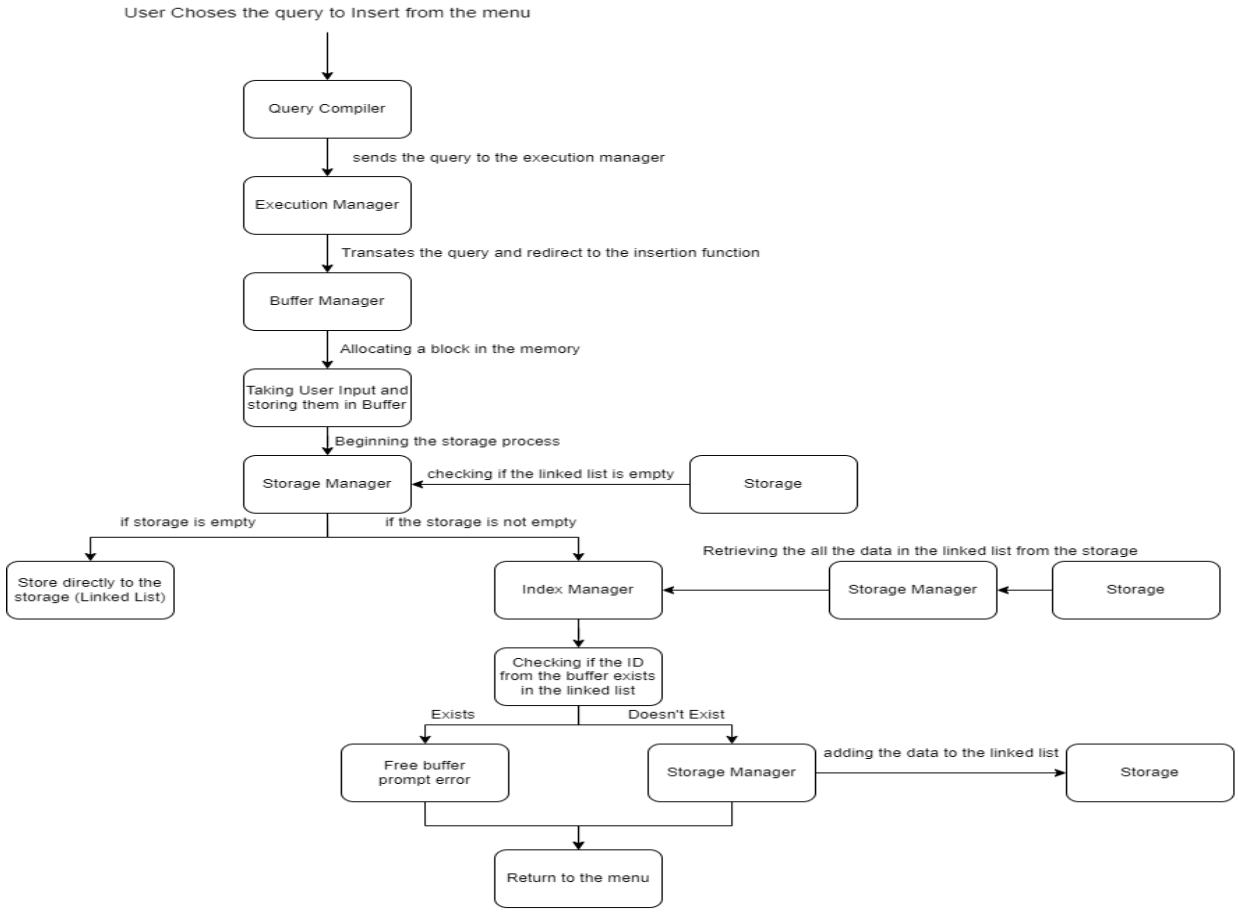


Figure 2: Inserting Process

As seen in figure 2, the process starts when the user requests the insert option from the menu, where the Query Compiler would receive the request and sends it to the execution manager to be translated into an action, which is executing the insert function. Then once in the Insert function, a memory block is allocated in the memory using the buffer manager. Then the system would require the user to input the products information, where they would be temporarly saved in the buffer. After the data has been saved in the buffer, the storage manager would check if the storage is empty (linked list is empty), if it is, then store the information directly to the linked list, and if its not empty then check if the ID is present in the linked list and if it was then free the buffer and prompt the user with an error then return to the menu, and if wasn't present then add the data to the linked list and then return to the menu.

2. Updating Process:

Here is the data movement when updating a product in the DBMS:

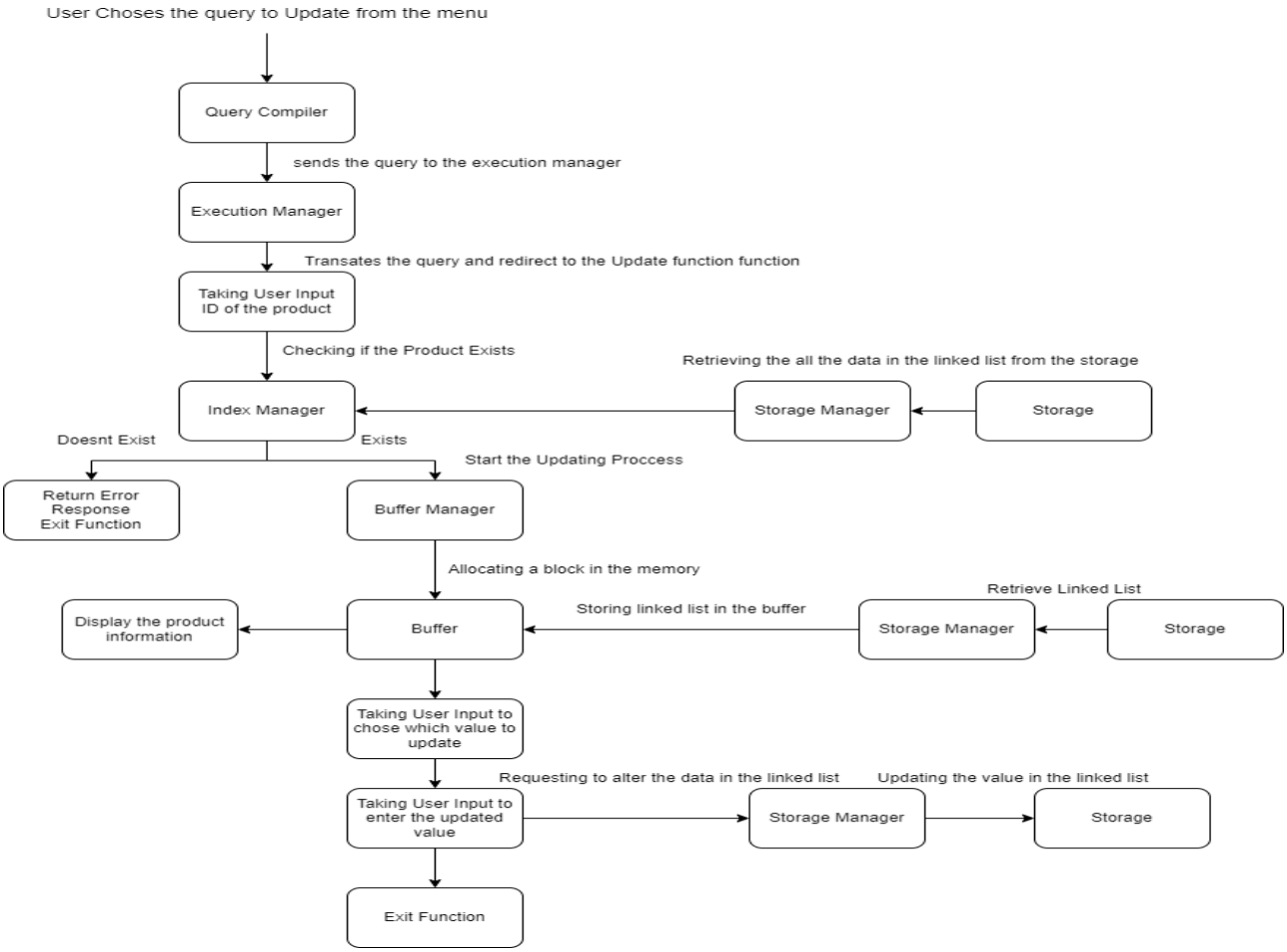


Figure 3: Update Process

As seen in figure 3, the process starts similarly to the insertion process, however when the user is redirected to the update function they are required to enter the ID of the product they want to update where the ID is then ran through the index manager where the index manager would retrieve the linked list from the storage using the storage manager to check if the ID exists, and if it does then return the current position of the product in the linked list and save it in the buffer after the allocating a block in the memory through the buffer manager and then display the retrieved node from the linked list. Then the user is prompted with all the product information of the product they searched for and required to chose which value they want to update and once selected they are prompted to input the new value for which it would be directly saved in the storage after requesting to alter the node from the storage manage and once the update process is done then the user returns to the menu.

3. Deleting Process:

Here is the data movement for deleting a product from the DBMS:

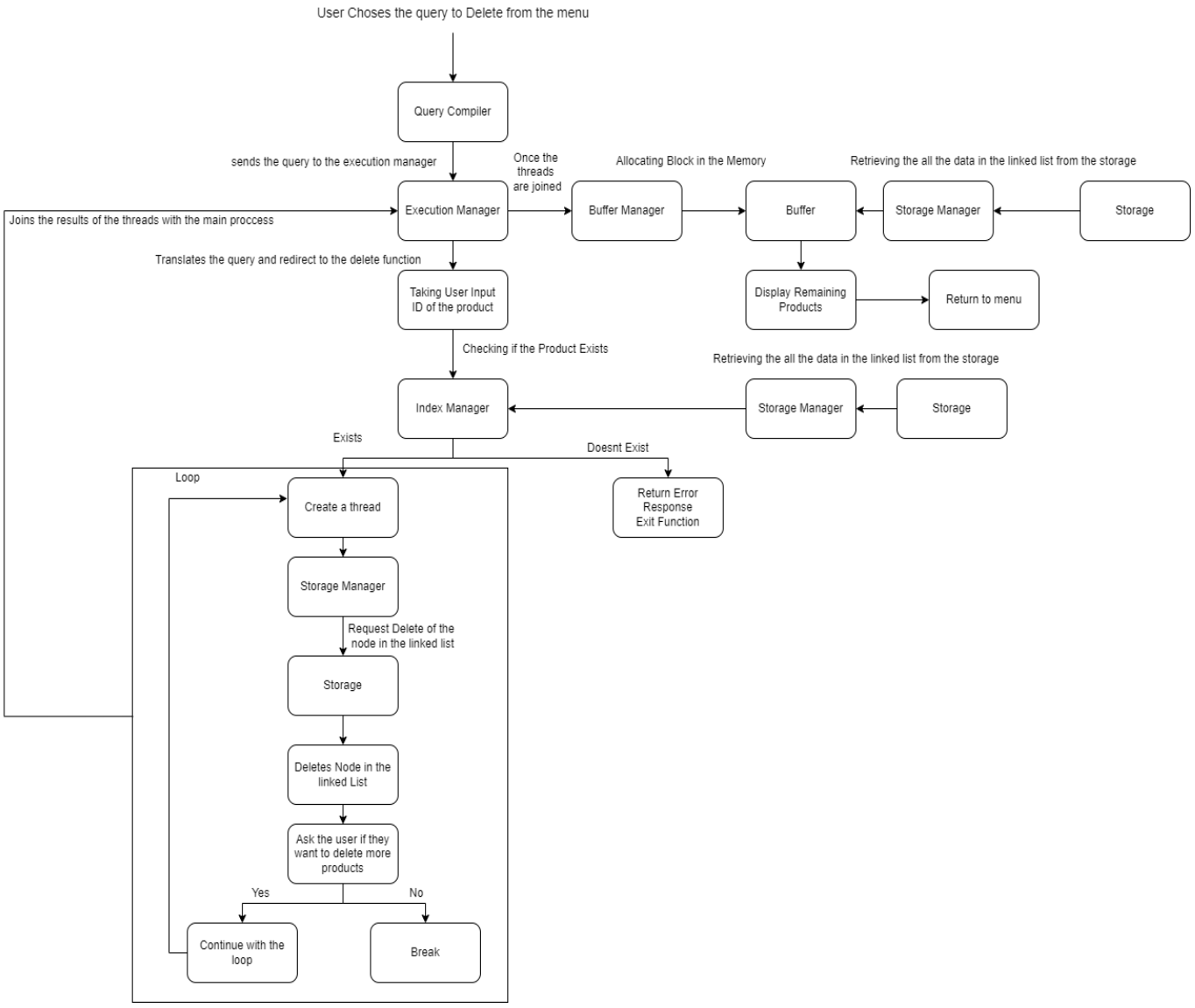


Figure 4: Delete Process

As seen in figure 4, the process starts similarly to the update until the product is found by the index manager, where when the product is found the concurrent process stars by creating a thread. In this thread the delete process starts with requesting to delete the node from the storage (linked list) through the storage manager. When the product is deleted, the user is prompted with the choice to either delete another product or to stop, when the user wants to delete more products, the loop continues, and more threads are created to accommodate each delete request. When the user stops the deleting process, the threads that have been created along with the results are then joined with the main process in the execution manager. After the threads have been joined, the remaining threads are saved in the buffer, after a memory block was allocated by the buffer manager, and then displayed to user then the user returns to the menu.

4. Read Process:

Here is the data movement when reading products in the DBMS:

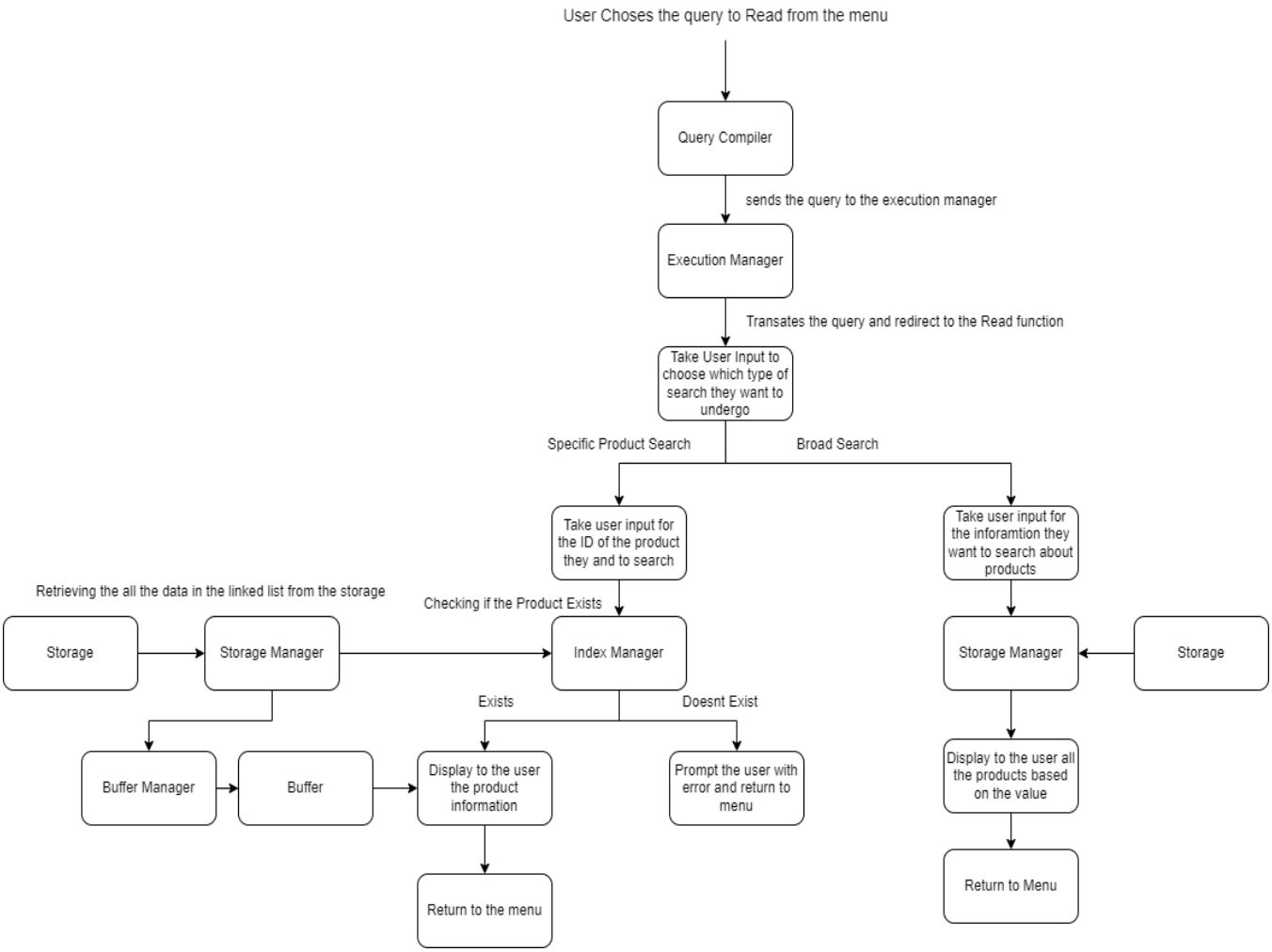


Figure 5: Read Process

As seen in figure 5, the read process starts similarly to the other processes until when the user is redirected to the read function, where the user will be prompted with a submenu asking them the type of search they want to perform, if the user selects specific search, the process would be similar to the update and delete when searching for the product using the ID, however instead of deleting or altering the product, it would be displayed to the user then the user returns to the main menu. As for the Broad Search, the user will be prompted with a second menu showing all the read operations that the user can chose, and once the user chose a read operation, the user is prompted to input the value they want to search where the storage manager would retrieve all the products that are within the search category and then all the product within the search category would be displayed to the user then the user returns back to the main menu.

5. Flowchart showing the whole process:

Here is a flowchart that shows the whole user journey when using Miles DBMS.

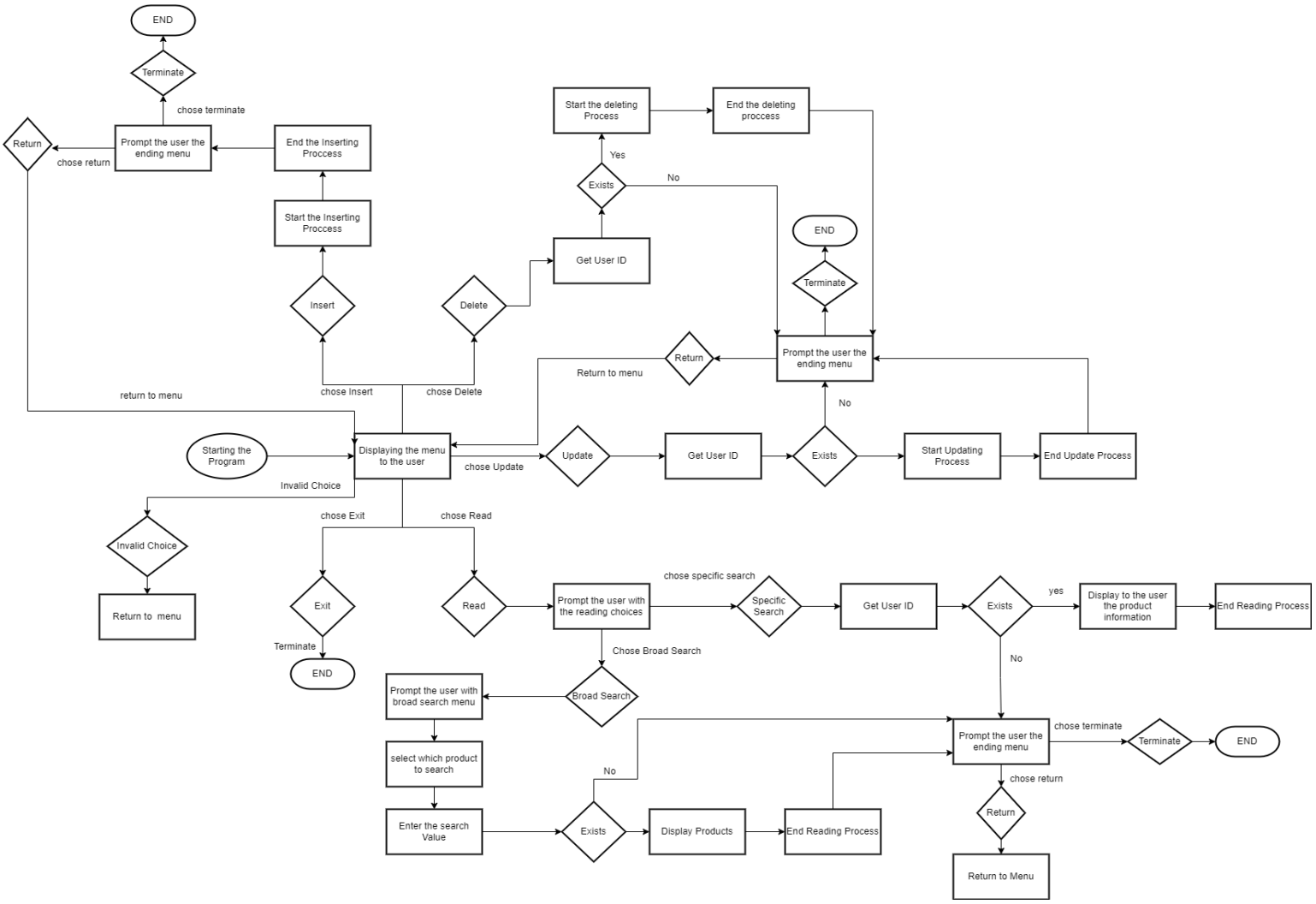


Figure 6: Flowchart of the whole user journey when using Miles DBMS

As seen in figure 6, the user journey starts with the main menu where it has 5 operations. The user can chose each operation to execute the actions, which are insert, delete, update, read, and exit, and if the user didn't chose any of the operations in the menu, an error would occur, and the user is returned to the menu. Also, one the user finish with an operation, they are prompted with a menu to either exit the program or return to the menu whether the operation was successful or not.

D. Evaluating the Effectiveness of the Design in terms of System and User Requirements:

After implementing the design of Miles DBMS and evaluating using diagrams, we had a better understanding of the how the design can be operated by the user and how the data can move between DBMS components. The design in terms of the user requirements was as the user intended where there are all the operations that the needed to be implanted from adding, deleting, updating, and reading products. Also, the design included Miles most important requirement, which is the ability to search on a broader scale, not just for a specific product but for values that some products share, which would help them to make better business decisions. This was implemented in the form of a broad search in the design as seen in the diagrams.

However, with the system requirements comes a different case, although we have used and implemented each the main components from the proposed DBMS architecture, which are storage and storage manager, buffer and Buffer manager, query compiler, execution manager, concurrent control, and index manager, onto the design. There were some components that could've been designed better to provide better scalability and DBMS performance for Miles DBMS. The components where the storage and the index manager. For the storage, the designed data structure used for the storage was the double (Circular) linked list, which is great for a simple DBMS since there aren't that many data going to be stored in it. However, as Miles grow as so the data storage where the linked list would reach a point that it couldn't handle all that data and the performance would drop significantly. The solution that me and my team came up with is the use of trees or graphs data structures, because unlike the linked list as the data grows, the trees and graphs would grow as well to help accommodate the data increase.

As for the Index manager component, it was implemented using while loops to search for a unique ID in the linked list and then return the current position (index) of the node. It works now since the DBMS is small, but as it grows along with the data storage, this process wouldn't be feasible anymore as its very slow because it would require more time to search for products since the time complexity of this approach is o(n) in its best case which would be much worse in its worst case. That's why me and me came up with another solution for the Index manger by implementing a hash map, where we can map the data we want to search for with a key, where using this key the data can be retrieved in an extraordinary faster time than the previous approach with a time complexity of O(1) in its best case and O(n) in its worst, being a major improvement over the previous approach.

III. Part 2: Developing DBMS Main Functionalities:

This section of the report will talk about the implementation of the dynamic data structures at the storage and buffer manager level, then the SQL queries that have been used and mapped to C functions as CRUD functions, then the techniques that have been used to optimize these SQL queries and how have they been implemented in the C functions, and finally a critique on the process of mapping SQL queries to the C functions.

A. Developing the Data Structures:

The first step of creating Miles DBMS was to create the dynamic data structures in C. the data structure that is going to be implemented is the circular linked list (double) because of its versatile by allowing the nodes to be pointed to the next and previous nodes. By dynamic data structures, means that there must be a memory block allocated in the memory from the size of the linked list where the data must be stored first in the buffer then its either going to be inserted to the linked list or displayed to the user.

*Note: The functions have been implemented in the C program attached.

B. SQL Queries and CRUD Functions:

The norm of database is using queries to manage the data in the database. These queries are usually written in SQL (Structured Query Language) where these queries relate to the normal English language, for example to get all the data in a database, the SQL query is SELECT * from [TABLE NAME]. The queries that Miles DBMS will are:

1. SQL Queries:

Here are all the queries written in SQL format:

Creating the Products Table:

The first query is for creating the table that will hold the data. The query is:

```
CREATE TABLE Products (
ID int NOT NULL PRIMARY KEY,
Name varchar(500) NOT NULL,
Category varchar(500) NOT NULL,
Dateofarrival DATE NOT NULL,
SupplierName varchar(500) NOT NULL,
supplierCountry varchar(500) NOT NULL,
quantity int NOT NULL);
```

Inserting into the Products Table:

This is the second query to insert the data into the products table:

INSERT INTO Products (ID, Name, Category, Dateofarrival, SupplierName, supplierCountry, quantity) VALUES ([ID VALUE], [NAME VALUE], [CATEGORY VALUE], [Dateofarrival VALUE], [supplierCountry VALUE], [quantity VALUE]);

Update a product information in the Products table:

This is the third query to update a certain field within a column for a specific product inside the Products table:

UPDATE Products SET [COLUMN NAME ex. Product Name] = [NEW VALUE]

WHERE ID = [ID of the product to modify];

Delete a product in the Products Table:

This is the fourth query to delete a specific product within the Products Table:

DELETE FROM Products WHERE ID = [ID of the product to delete];

Searching for a specific product:

This is the fifth query to search for a specific product within the Products Table:

SELECT * FROM Products WHERE ID = [ID of the product to search];

Where the "*" means to get all the product information.

Searching for all products:

This is the sixth query to search for all the products within the Products Table:

SELECT * FROM Products;

Searching for products based on a value:

This the seventh and final query to search for products based on a certain value, for example to search for all the products that have the same category or name.

SELECT * FROM Products WHERE [NAME OF THE VALUE ex Category] = [NAME OF THE VALUE TO SEARCH];

2. Mapping The SQL queries to C format to perform CRUD operations:

Here are all the rewritten SQL statements in C format:

Creating the Products Table:

This is done by implementing a structure (struct) that has all the products information then within the insert function, a condition occurs to see if the table exist or not, if not then the new table will be created in the form of a linked list.

Inserting into the Products Table:

This done by implementing a function in C to insert data to the created linked list where the data will be temporary saved in a buffer allocated in a memory block that has the size of the linked list and using the pointers in the linked list to add the information then move to the next node to add the other information.

Update a product information in the Products table:

This is done by using a function to search within the linked list using a while loop then if the id input by the user found the product information will be prompted to the user to indicate which value to update by simply replacing the value in the node with a new value.

Delete a product in the Products Table:

This is done by using a function to search within the linked list using a while then if the id input by the user was found then the product will be deleted within the linked list by making the next node the current one, moving the next node of the deleted node backwards

Searching for a specific product:

This is done by using a function to search within the linked list using a while then if the id input by the user was found then the product will be save in the buffer then displayed to the user using print statements.

Searching for all products:

This is done by using a function to search within the linked list using a while for all the data in the linked list then these data will be save in the buffer then displayed to the user using print statements.

Searching for products based on a value:

This is done by using a function to search within the linked list using a while and string commands to compare between different string values or normal integer comparisons using if statements. Then when the products are found they will be displayed to the user through print statements.

*Note: the detailed mapping for these functions can be found in the C program attached.

C. Assessing the Techniques used to Optimize SQL queries:

Using the SQL queries by themselves and then mapping them directly into C is not enough. These queries must be optimized first before implementing them on C.

1. SQL Query Optimization and mapping them to C:

Here are the techniques used to optimize each SQL query:

Using Primary Keys and NOT NULL operators

When creating the products table, the product ID is identified as a primary key, which is a unique id used to identify a certain row, so that there will be no incorrect outcomes when searching or deleting or updating the product. Also, the use of NOT NULL operator in SQL to ensure that

there are no null values in when inserting into the table. This was done using C by performing a search inside the insert function where it compares the id the user input with the ones presents in the linked list, if the ID was found then free the data saved in the buffer and prompt the user with an error message indicating that the ID exists. As for the NOT NULL, the use of scanf in C ensures that the user doesn't insert empty data (NULL) in the linked list.

The Use of WHERE Statements:

When wanting to search for a product to either display, modify, or delete it, the WHERE statements are very important as they compare the data inside the table with the a certain value whether it was the id to search for a specific product or other values to search for a certain number of products. This was done in C by creating a function that searches for the id and any other value within the menu, these functions are called idSearch() and it takes the id as a parameter, and broadSearch() where it takes the other values as parameters. Where the idSearch() function operates by going through the whole linked list using a while loop where it increments by pointing to the next node, then the id input is then compared within the ID's in the linked to using an if statement, and when the ID is found, the function would return a confirmation and make the current node the node that the id was found. The same process goes for the boradSearch() function, however, since some variable are strings, the use of the string library which has a function that helps in comparing strings called strcmp() is used to compare the string input with the strings (varchar) inside the linked list.

Identifying the type of the data to be inserted into the Products Table:

When wanting to add data to the table, a type must be defined, there cannot be strings where the value requires an integer. This helps to avoid any unnecessary inputs to a field that may require an integer which could cause problem when searching based on this value. This is done in C by defining a structure that hold all the column with their type, where the ones that would take a string input are defined as character array (char arrays) and to input data into them the %s format in the scanf is used, whereas the integers are defined as int.

*Note: check the C program attached for the full mapping process in the functions

D. Criticizing the process of SQL Queries Mapping to C:

After creating the SQL queries and mapping into C as CRUD statements, I become more knowledgeable about the process of mapping the queries into C. the process was straightforward and simple to implement since most of the requirements needed to map the query into C are present from while loops to if statements to structures (struct) to save the data. But there where things in SQL that couldn't be implemented in C, or they need a complex function to be implemented. The main thing was the DATE datatype in SQL, since there was a field called the date of arrival that required the DATE datatype as it was written in the SQL query, however there is no DATE datatype in C which made the mapping from SQL a complex task where we had to settle the date of arrival datatype as a character array for now then in the future a complex function in C can be implemented to transform this string array to a date type. What me and my team found that the SQL queries are have a complex structure hidden within the statement as its not just an inserting or deleting process, there is more to it and that what we saw in C when we implemented the CRUD statements as it took multiple of steps to build each single function.

IV. Part 3: Principles of Concurrency Control:

This section of the report will talk about the standard DBMS concurrency control techniques and then implementing a parallel data processing functionality in the C program to handle concurrent database operations.

A. Description of the Standard DBMS Concurrency Control Techniques:

As previously mentioned, concurrency control is a concept which helps in the management of multiple simultaneous processes to execute without conflicting each other (Priya, 2021).

There are multiple of advantages when using concurrency control in the DBMS, which are that the waiting time and the response time will decrease along with increasing the resource utilization, system performance and efficiency.

Concurrency Control concept description can be split into two ways, the first one is explaining how the computational power is distributed among different concurrent requests, and the next one is describing the concurrent techniques that can be used in the DBMS.

1. Distribution of Computational Power among Different Concurrent Requests:

One of the biggest features of concurrency control is the ability to distribute the computational power of a server or device to process multiple concurrent requests. This process is known as threading, where a thread is created to handle a concurrent request then when the result is produced, it is joined with the main process. Each concurrent request is handled on a different thread, meaning the computation power of the server or the devices is distributed to handle multiple concurrent requests then when the request is over and a result is produced, all distributed computational power will be joined to the main device along with the result of the request.

This process is implemented for the Miles DBMS, where multiple of threads are created for deleting a specific product, where for each deletion a thread is created and one the deletion is successful, all the created threads are joined in the main process along with their results.

2. Concurrency Control Techniques in DBMS:

There are many concurrencies control in a DBMS is provided to enforce isolation among different transactions, preserver the consistency of the database through preserving execution of transactions, and finally resolve read-write and write-read conflicts (geeksforgeeks, 2019). There are many concurrency techniques used in the DBMS, which are:

Two-phase locking Protocol

Locking is an operation that guaranties exclusive use of data items to current transaction by which it secures the permission to read and write. Two phase locking is a process used to gain ownership of shared resources without creating the possibility of deadlock by releasing all the resources it has acquired. This means that there will be no process in a state where its holding shared resources (Priya, 2021) (geeksforgeeks, 2019).

Timestamp ordering Protocol

Timestamp means a tag which is attached to any transaction, this tag denotes the tie for which the transaction has been used or requested. Time stamps can be generated using the system clock or logical counter which starts when the transaction begins (Priya, 2021) (geeksforgeeks, 2019).

Multi version concurrency control

Multi version is a concurrency control technique used to keep old version of the data item to increase concurrency, whereby each successful write would result in the creation of a new version which would be tagged by a timestamp. So, when a read(X) operation is issued to read the data item, then the appropriate version X would be selected based on the timestamp of the transaction (geeksforgeeks, 2019).

Validation concurrency control

Validation concurrency control is also known as the optimistic approach where this technique is based on assumption that it would be rare is face a conflict and would be more efficient to allow the transactions to proceed without any delays. This approach would not require neither the timestamping technique nor the locking techniques, meaning that there would be no delays in proceeding with the transaction (Priya, 2021) (geeksforgeeks, 2019).

B. Implementing Parallel Data Processing:

After learning about concurrency and their techniques, I have gained a better understanding to know where to implement the parallel data processing for Miles DBMS. The method that the concurrency would be implanted is by threading using a library in C called pthread to help implement threading the program. Threading would be implemented on the delete process where a new thread would be created for each delete process then the results are joined with the main process.

*Note: check the C program attached to find the delete function with threading.

V. Part 4: Testing and Evaluating the Program:

This section of the report will suggest a test plan used to test the whole system functionality after the implementing all the components necessary and creating a menu to help in navigating all the components in Miles DBMS. Also, there will be another section to critically evaluate the whole system in terms of functionality, interface design, and the ability to handle concurrent databases in terms of user and system requirements.

A. Creation and Implementation of the Test Plan:

After the implementation of Miles DBMS on C was done, the system must be tested to ensure the compatibility and functionality of the developed DBMS components.

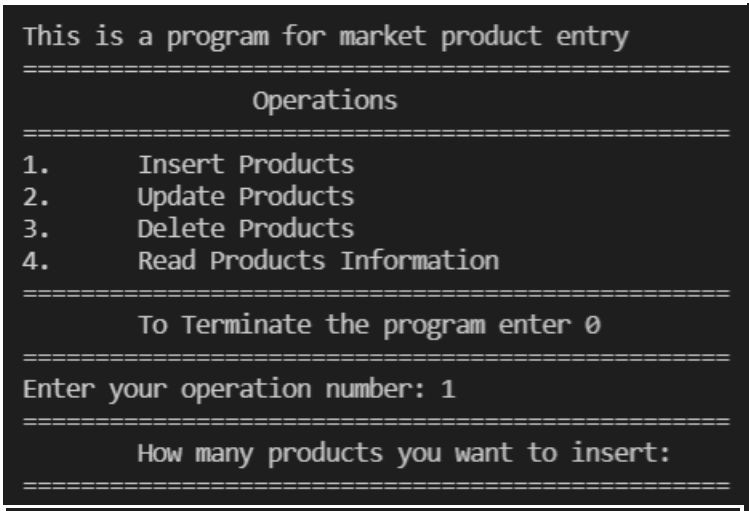
Creation of the Test Plan:

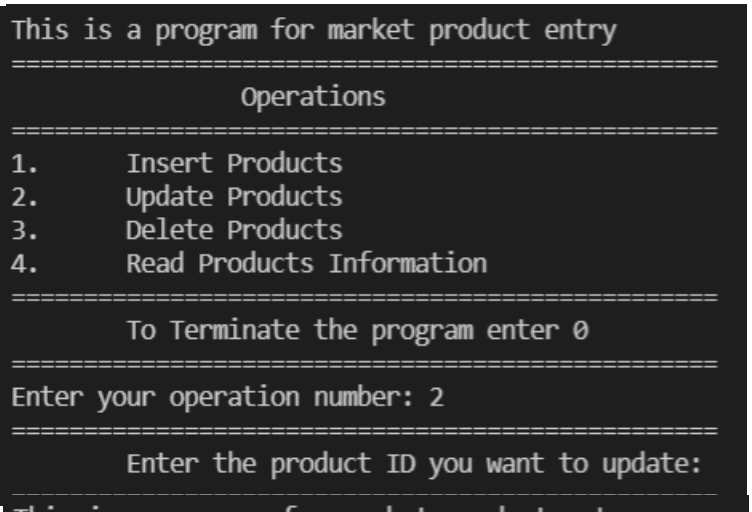
Test	**Description**	**Expected Results**
Menu Redirection	For this test, the menu must redirect the user to the corresponding page	1. Redirection to the desired page 2. A message prompting any invalid input
Insertion of Products	For this test, the products must be inserted into a linked list with unique ID's. and to test if the ID entered are unique	1. The product is successfully inserted into the linked list 2. An error message prompts when entering an already existing ID and the data inserted by the user aren't inserted in the linked list
Deletion of Products	For this test, the intended product must be deleted not any other product, along with the creation and joining of threads.	1. Successful deletion of the intended product 2. Successful creation of threads 3. Successful Joining of threads 4. Error message if the user uses an ID that is not present in the linked list.
Updating Product's Information	For this test, the intended product's information must be updated and then displayed with the new information.	1. Successful update for the product information 2. Error message if the user enters an ID is not present in the linked list
Reading Different Products	For this test, there would be two searches. One for searching for a specific product and another search for searching all the products that are within the Dairy Category, products that arrived at 13th of March 2020 and finally products that have been brought by the same supplier.	1. Specific product displayed in the terminal 2. Products within the dairy category are displayed in terminal 3. Products that have arrived at 13th of March are displayed in the system 4. Products with the same supplier Name are displayed in the terminal 5. Error Message if the product is not present.

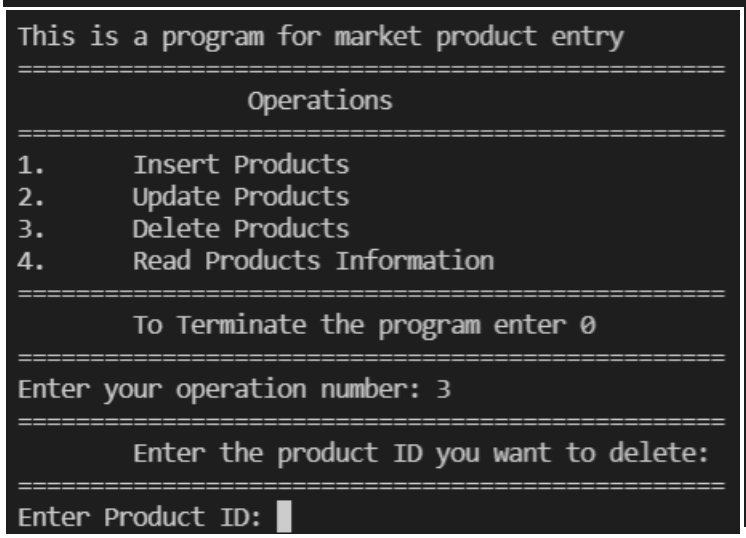
Implementation of the Test Plan:

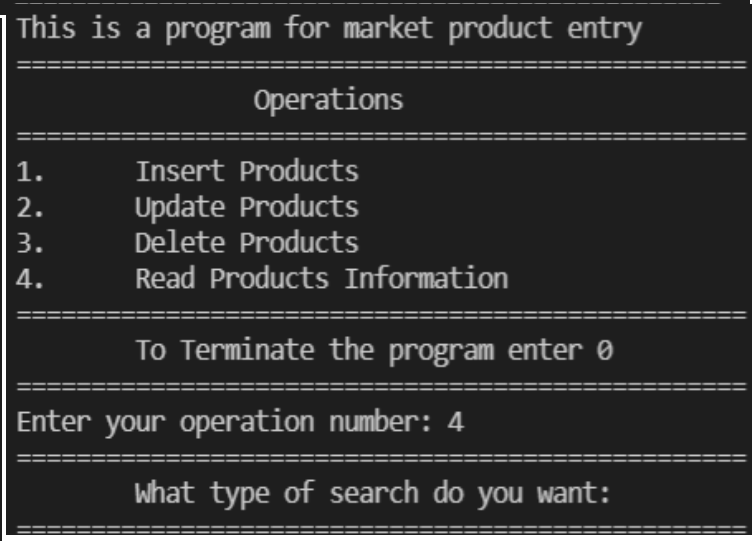
Here is the actual result from every test in the test plan:

Menu Redirection:









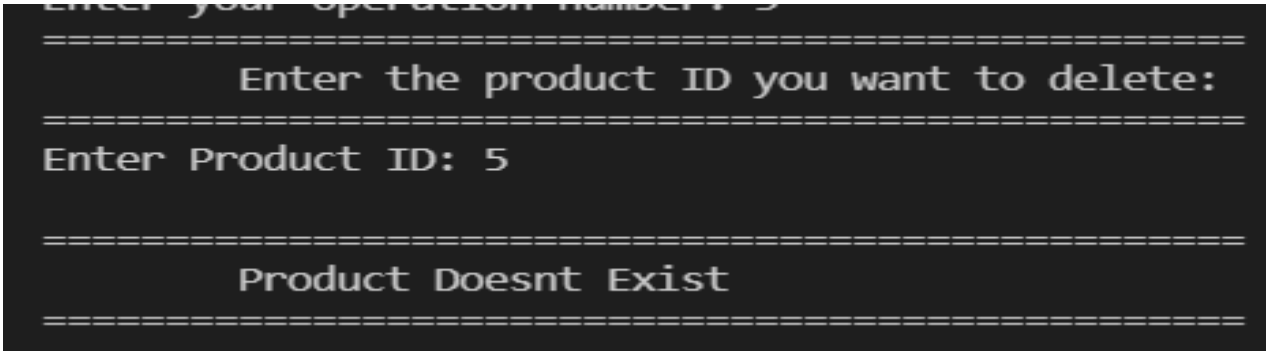
As seen in the screenshots above, the actual result was as expected with the successful redirection to each operation in the menu.

Insertion of Products:

As seen in these screenshots the results was as expected with the successful insertion of Products and the prompting of the error message indicating that the ID exists.

Deletion of Products:

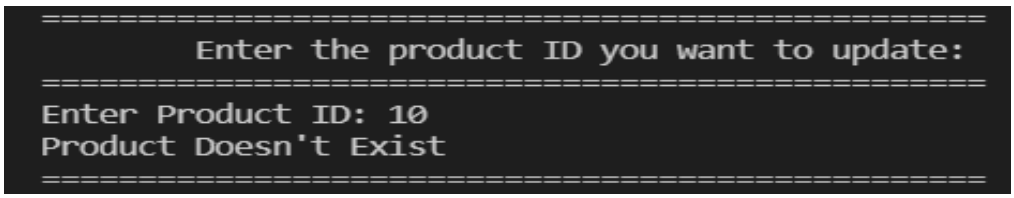
A new product called “4” is inserted into the system to test the deletion.



As seen in the screenshots the deletion successfully worked and the creation and joining of threads also worked since there was no error message indicating that the thread nor the join had failed. And as for the product 5, since it wasn’t I the system, it prompted that the product doesn’t exist.

Updating Product's Information:

For this test a new product called Pepsi would be inserted in the system.

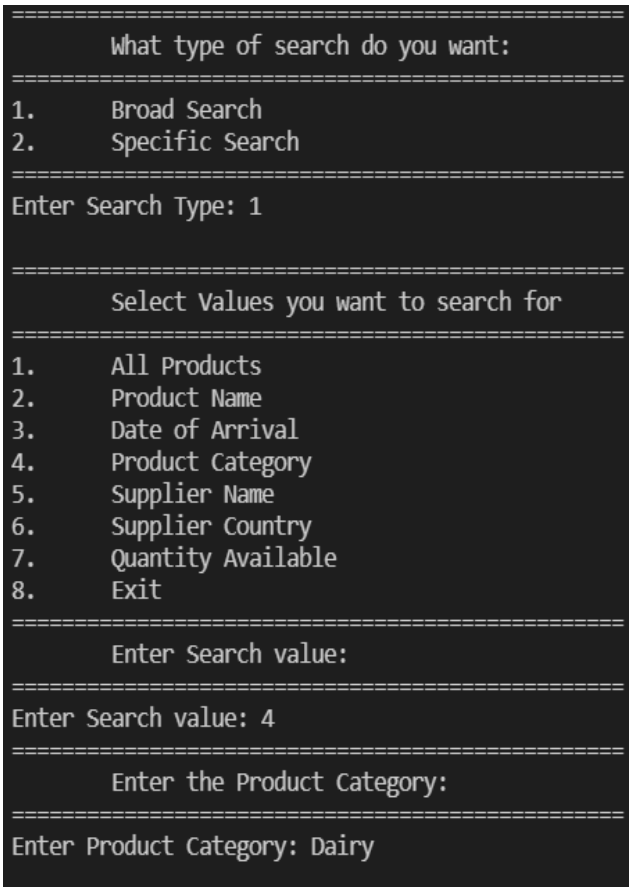


As seen in these screenshots, as expected, the intended product was modified where the modified value was the supplier country from USA to UK then save back to the Linked list successful. Also, as product ID 10 is not present in the system, then an error message as expected was prompted indicating that the product doesn't exist.

Reading Different Products:

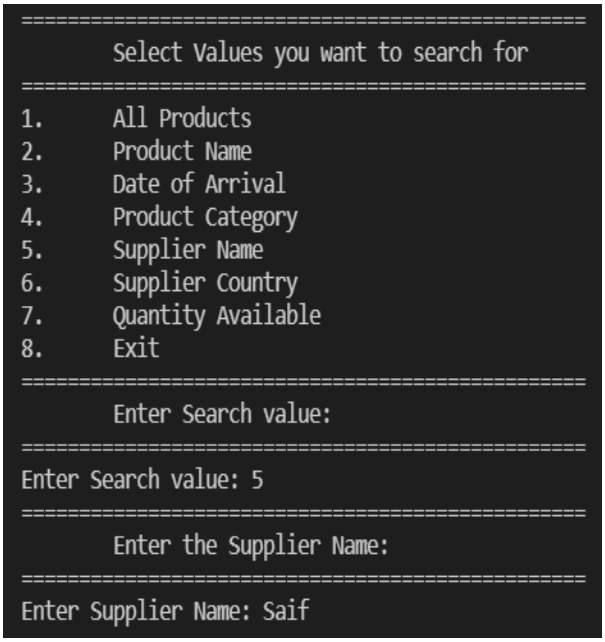
For this test, the product that is going to be search specifically is Chips with ID of 2.

As Expected the Specific Search worked and displayed the information for Chips.

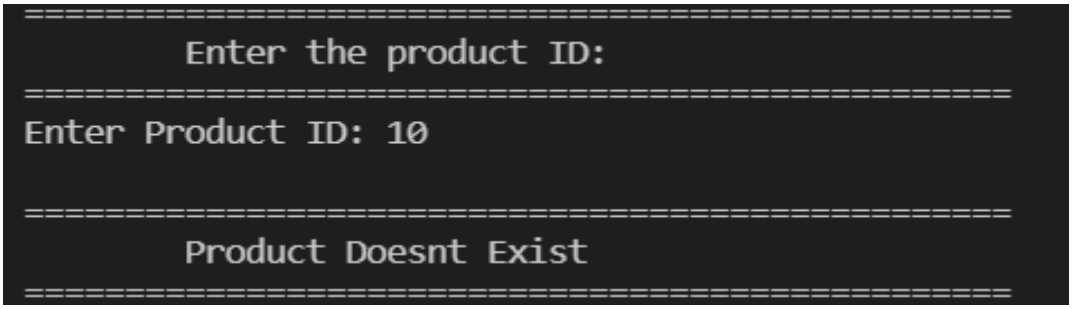


As expected, the all the products within dairy have been displayed on the terminal.

As expected the products that have arrived at 13th March 2020 were displayed on the terminal.



As expected, all the products that have been supplied by Saif have been displayed on the terminal.



As expected, an error message will be prompted if the product searched doesn’t exsist in the system.

B. Critical Evaluation of Miles DBMS:

After finishing the implementation of the design into C format and creating a menu to redirect each operation to its corresponding page along with creating and implementing a test plan to test the functionalities of the DBMS. We gain a lot of new knowledge about the process behind building, designing, and managing a DBMS.

For Miles DBMS, as seen through implementation of the test plan, we can say with full confidence that the system is fully functional as it passed every check within the test plan. However, that doesn't mean that there aren't t room for improvements, one main improvement should be on the test plan to create a more rigorous test that can confirm that the DBMS is fully function in the best and worst cases. Also, one improvement that needs to be implemented to make the DBMS more functional than its now is to implement more error handling techniques to avoid system crashes. Finally, one major functionality that needs to be implemented ASAP is the ability backup the data in the DBMS along with the ability to recover them, as any system crash or downtime may result in the loss of these data, that's why this functionality is essential to be implemented in the future.

As for the interface design, we have created a small menu that can help the user redirect to page they want, along with giving the ability to display error and success messages along with the output. However, the design of the menu is purely basic and must not be allowed to remain when Miles scales up and have more users, because this design would provide a poor user experience as there are no colors nor graphics to make it more appealing. Also, the user must write on the terminal the operations that they want to use which is a recipe for poor user experience. My team and I found a solution that needs to be implemented as soon as possible, which is to migrate to an advanced menu that uses graphical user interface (GUI) to present the menu and the user can navigate within the DBMS using their mouse clicks which makes the user experience much better than before.

Finally, as for the ability to handle concurrent databases in terms of user and system requirements. I can say that all the requirements that Miles have required to be implemented in the system have been successfully implemented by us as seen in the menu where the user can insert, delete, update, and read products as intended by Miles. As for the system requirements, every component that has been talked about in the architecture has been implemented in the DBMS with a design to show how the data moves between each DBMS component in the architecture. Also, we have implemented the ability to handle concurrent request on the delete functionality so that every delete request would have its own computational power to handle this request and once done it can be then joined to the main process as it was explained when designing the DBMS components. In my opinion, what we did here was an excellent job, but there are also rooms to improve a lot of the implemented components to make the performance of the DBMS match or even succeed the completion.

To conclude, as team who had little knowledge to build a full database management system for Miles Hypermarkets, we think we did a great job not an excellent one because of all the necessary improvements that needs to be implemented as fast as possible. Luckily, Miles knows our work and potential, and they are excited to watch us grow as DBMS developers as we are working on modifying and improving their Newly created DBMS.

VI. Bibliography

geeksforgeeks, 2019. *Concurrency Control Techniques.* [Online]
Available at: https://www.geeksforgeeks.org/concurrency-control-techniques/
[Accessed 30 January 2022].

geeksforgeeks, 2021. *Circular Linked List | Set 1 (Introduction and Applications).* [Online]
Available at: https://www.geeksforgeeks.org/circular-linked-list/
[Accessed 30 January 2022].

Priya, B., 2021. *What do you mean by buffer in C language?.* [Online]
Available at: https://www.tutorialspoint.com/what-do-you-mean-by-buffer-in-c-language
[Accessed 30 January 2022].

Priya, B., 2021. *What is concurrency control in DBMS?.* [Online]
Available at: https://www.tutorialspoint.com/what-is-concurrency-control-in-dbms#:~:text=Concurrency%20control%20concept%20comes%20under,occur%20in%20multi%20user%20systems.
[Accessed 30 January 2022].