# Visvesvaraya Technological University

# Belagavi-590018, Karnataka

A Mini Project Report on

## "SIMPLE INDEXING FOR SALES RECORDS"

**Submitted in partial fulfillment of the requirement for the**

**File structures Laboratory with mini project**

**[17ISL68]**


**Submitted by**

**SHASHANK N [1JT17IS037]**


Under the support and guidance of

**Mr.Vadiraja A**

Asst.prof, Dept of ISE



## Department of Information Science and Engineering

## Jyothy Institute of Technology

## Tataguni, Bengaluru-560082

# Jyothy Institute of Technology

## Tataguni, Bengaluru-560082

## Department of Information Science and Engineering

# CERTIFICATE

Certified that the mini project entitled **"SIMPLE INDEXING FOR SALES RECORDS"** carried out by **SHASHANK N [1JT17IS037]** bonafide student of Jyothy Institute of Technology, in partial fulfillment for the award of **Bachelor of Engineering** in **Information Science and Engineering** department of **Visvesvaraya Technological University, Belagavi** during the year **2020-2021**. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library.The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree

**Mr.Vadiraja.A**
Asst. Professor
Dept. Of ISE

**Dr.Harshvardhan Tiwari**
Assoc.Professor and HoD
Dept. Of ISE

External Viva Examiner

1.
2.

Signature with Date :

# ACKNOWLEDGEMENT

Firstly, I am very grateful to this esteemed institution **Jyothy Institute of Technology** for providing me an opportunity to complete my project.

I express my sincere thanks to our Principal **Dr. Gopalakrishna K** for providing me with adequate facilities to undertake this project.

I would like to thank **Dr. Harshvardhan Tiwari, Professor and Head of Information Science and Engineering Department** for providing for his valuable support.

I would like to thank my guide **Mr. Vadiraja A, Asst. Prof.** for his interest and guidance in preparing this work.

Finally, I would thank all our friends who have helped me directly or indirectly in this project.

**SHASHANK N [1JT17IS037]**

# ABSTRACT

This project titled “SIMPLE INDEXING FOR SALES RECORDS” has been done using Eclipse IDE with the platform Windows and language Java.The database used for the project is ‘Sales’ records.

The project mainly focuses on building the index for the records which is fed in CSV format file ,then various operations with a menu choice is displayed to the end user, such as Insert ,Search ,Delete and Modify.

For the purpose of searching efficiently,binary search algorithm is being used.The inserted record will be initially packed and then will be put in the record file.The user can also Unpack all the records in the file which will be displayed.

The index files generated comprises of a key value and its respective position in the record file.This position will be used for various operations.

So,Indexing is a ‘way to optimize the performance of file access by minimizing the number of disk accesses required to process the required data’.

# *CHAPTER 1*

# *INTRODUCTION*

# INTRODUCTION

## 1.1 Introduction to File Structures

In simple terms, a file is a collection of data stored on mass storage (e.g., disk or tape).But there is one important distinction that must be made at the outset when discussing file structures. And that is the difference between the logical and physical organization of the data.

On the whole a file structure will specify the logical structure of the data, that is the relationships that will exist between data items independently of the way in which these relationships may actually be realized within any computer. It is this logical aspect that we will concentrate on. The physical organization is much more concerned with optimizing the use of the storage medium when a particular logical structure is stored on, or in it. Typically for every unit of physical store there will be a number of units of the logical structure (probably records) to be stored in it.

For example, if we were to store a tree structure on a magnetic disk, the physical organization would be concerned with the best way of packing the nodes of the tree on the disk given the access characteristics of the disk.

Like all subjects in computer science the terminology of file structures has evolved higgledy-piggledy without much concern for consistency, ambiguity, or whether it was possible to make the kind of distinctions that were important.

It was only much later that the need for a well-defined, unambiguous language to describe file structures became apparent. In particular, there arose a need to communicate ideas about file structures without getting bogged down by hardware considerations.

## 1.2 Introduction to File Systems

In computing, a file system or file system controls how data is stored and retrieved. Without a file system, information placed in a storage medium would be one large body of data with noway to tell where one piece of information stops and the next begins. By separating the data into pieces and giving each piece a name, the information is easily isolated and identified.

Taking its name from the way paper-based information systems are named, each groups of data is called a “file”. The structure and logic rules used to manage the groups of information and their names is called a “file system”.

There are many different kinds of file systems. Each one has different structure and logic, properties of speed, flexibility, security, size and more. Some file systems have been designed to be used for specific applications.

File systems can be used on numerous different types of storage devices that use different kinds of media. The most common storage device in use today is a hard disk drive. Other kinds of media that are used include flash memory, magnetic tapes, and optical discs. In some cases, such as with tmpfs, the computer's main memory (random-access memory, RAM) is used to create a temporary file system for short-term use.

Some file systems are used on local data storage devices; others provide file access via a network protocol. Some file systems are “virtual”. Meaning that the supplied “files” are computed on request or are merely a mapping into a different file system used as a blacking store. The file system manages access to both the content of files and the meta data about those files. It is responsible for arranging storage space; reliability, efficiency, and tuning with regard to the physical storage medium are important design considerations .

## 1.3 Introduction to Simple Indexing

We know that data is stored in the form of records. Every record has a key field, which helps it to be recognized uniquely.Indexing is a data structure technique to efficiently retrieve records from the database files based on some attributes on which the indexing has been done. Indexing in database systems is similar to what we see in books.

Indexing not only helps in querying, but can also be used in any algorithm being built to reduce the time taken by that algorithm.

Simple indexes use simple arrays. An index lets us impose order on a file impose order on a file without rearranging the file. Indexes provide multiple access paths multiple access paths to a file─ multiple indexes multiple indexes(like library catalog providing search for author, book and title) An index can provide keyed access to variable-length record files.

Index is sorted (main memory) .Records appear in file in the order they entered.An index is defined on one or more columns, called key columns. The key columns (also referred to as the index key) can be likened to the terms listed in a book index. They are the values that the index will be used to search for. As with the index found at the back of a text book , the index is sorted by the key columns.

Primary index is defined on an ordered data file. The data file is ordered on a key field. The key field is generally the primary key of the relation.

Secondary index may be generated from a field which is a candidate key and has a unique value in every record, or a non-key with duplicate values. Many operations can be performed efficiently like search,delete , modify after building the index.

# *CHAPTER 2*

# *DESIGN*

## 2.1 Scope and importance of work

When a large number of files are maintained,the necessity of maintaining the index is increased. Indexing increases the utility of filing by providing an easy reference to the files. The very purpose of maintaining the index is that it is easy and quicker to find location of the files.

When a large number of files are maintained,the necessity of maintaining the index is increased. Indexing increases the utility of filing by providing an easy reference to the files. The very purpose of maintaining the index is that it is easy and quicker to find location of the files.

The advantage of using index lies in the fact is that index makes search operation perform very fast. So,adding an index to a column will allow you to query based on a column faster.

Suppose a table has a several rows of data, each row is 20 bytes wide. If you want to search for the record number 100, the management system must thoroughly read each and every row and after reading 99*20 = 1980bytes it will find record number 100. If we have a index, the management system starts to search for record number 100 not from the table,but from the index. The index,containing only two columns, may be just 4 bytes wide in each of its rows. After reading only 99*4 = 396 bytes of data from the index the management system finds an entry for record number 100.

The importance of indexing can be explained with the help of the following points:

1.Systematic arrangements of files:

Indexing helps to develop a modern scientific method of filing because indexing is not possible if the documents are not arranged in a systematic manner

2.Prompt location of files:

Indexing provides signs, symbol and guide to specific file in drawers. A person who needs a document in a file can make the use of an index to locate it.

3.Saves time and effort:

Indexing helps to locate the position of the specific document in files at a short period of time. It helps to make a quick decision by providing necessary information stored in files.

4.Develop efficiency:

Indexing facilitates the systematic arrangement of files and document.It saves the time required to search the information and space required to protect valuable document and information.

5.Reduce expenses:

The systematic arrangement and preservation of file reduce the overhead expenses of the office. The systematic arrangement reduces the space requirement to store the document.

## 2.2 Classification of Requirements

1. JDK - Java Development Kit.
2. Any Text editors (IDE preferred - Eclipse,Net-beans etc.)
3. Any Windows/Mac/Linux distributions

## 2.3 System Analysis

When the program is executed, two index files(primary and secondary index files) are created. Then the user is presented with the menu choice comprising of four operations namely:

**Insert**, **Search**, **Modify** and **Delete** which can performed on both primary and secondary indexes respectively.

The modify operation handles the records in such a way that, if the length of the modified record is greater than the length of the existing record then the modified record will be appended at end of record file. Or else it will be modified at the same position.

# *CHAPTER 3*

# *IMPLEMENTATION*

**The implementation of the entire process has been briefly described in the following stages below..**

# BUILDING INDEX FILE

Simple Indexing on sales records creates two index files namely, primary-index file and secondary-index file.

**Primary Index File (index.txt)**

The primary index file comprises of the primary key and it's starting position in the record file. Usually the first column in the record file will be unique and it will be considered as primary key column. In the Sales records data-set used in this project "order_id" is the first column which is unique, so it is being used as primary key for the purpose of running various operations based on this primary key like searching the record based on primary index, deleting the record based on primary index and also modifying the records based on this primary index file.

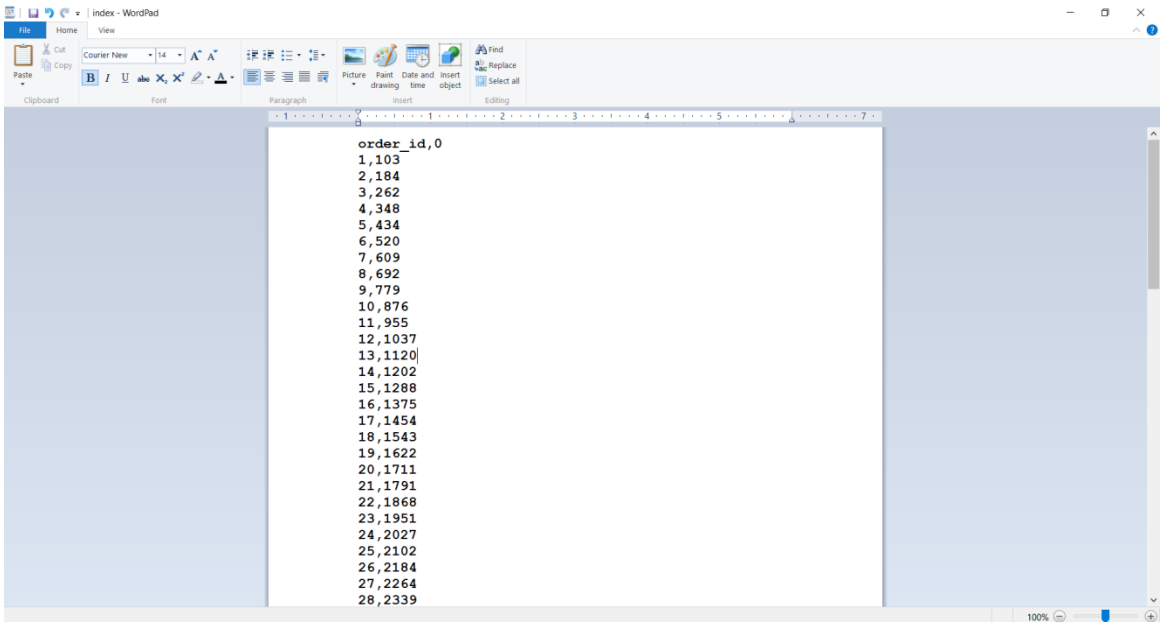


**Fig-3.1**

**Secondary Index File (index1.txt)**

The secondary-index file comprises of the secondary key and it's starting position in the record file. Any column in the record file can be considered as secondary key column. In the sales records data-set used in this project "place" is being used as secondary key for the purpose of running various operations based on this secondary key.

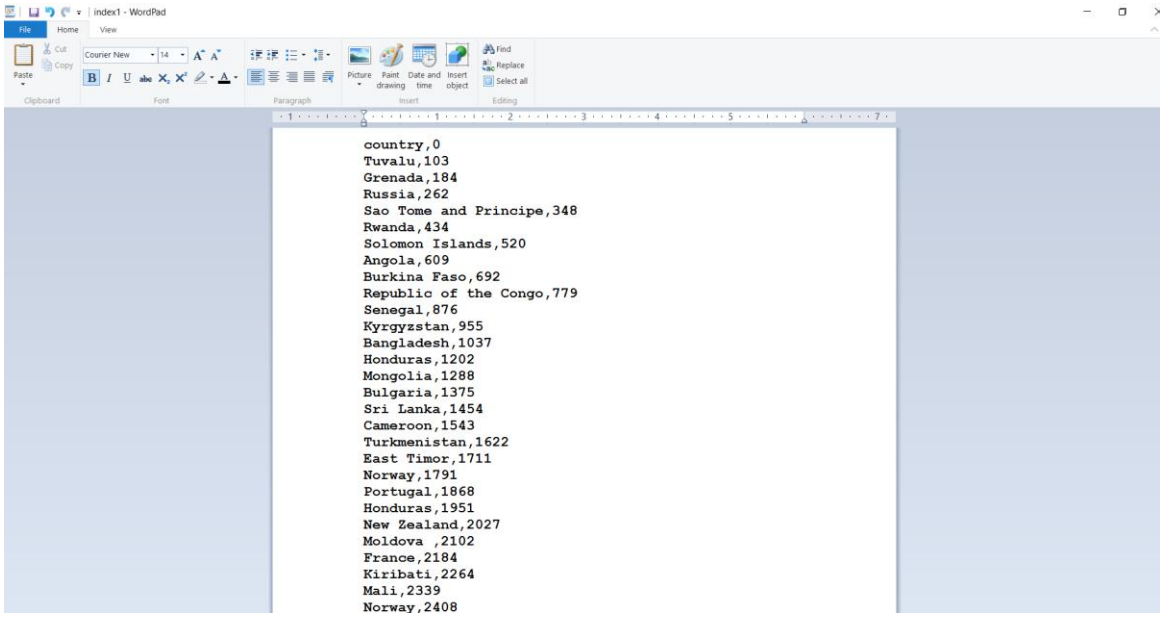


**Fig:3.2**

# INSERTING NEW RECORDS

The inserting operation is initial and important operation where various detatils regarding Sales records is accepted from the user using getData method and furthur the accepted Data is packed using the add method , then written onto the record and index files.

These records are appended onto the end of the record file.Then upon insertion of another Record file, the index files are updated.

## SEARCHING & MODIFYING RECORDS

The user can search a record from the record file based on primary- index or secondary-index.

Searching using primary-index file prompts the user to enter the primary key, this key will be used to locate the record in the record file. This search operation makes use of the binary search algorithm in order to efficiently search the primary key and seek its position.

Searching using secondary-index prompts the user to enter the secondary key, this key will be used to locate all the records which contains this secondary key.

The user can also modify the records present in the record file. This modify() method permits the user to modify any field of the record which he has searched.

## DELETING THE RECORDS

Similar to search operation, in delete operation the user has the privilege to delete the records. Deletion of records can again be done based on primary-index or secondary-index.

The record to be deleted will be prefixed with a asterisk (*) symbol, which indicates that the record is deleted and this record will not be considered while unpacking the records.

## UNPACKING THE DATA

The records inserted into the record files can be unpacked which displays all the fields of the record file with its respective values.

The unpacking method allows the user to view the records with all fields to furthur carry on with other operarions based on it.The Unpack function is executed by considering ‘,’ (comma) as a delimiter.

# TIME COMPLEXITY

The time duration to build the index files depends on the number of records in the record file ,larger the number of records it takes a large time to build the index for the records.

The graphical representation of time durations of building index files of various records comprising different no of records is illustrated below:

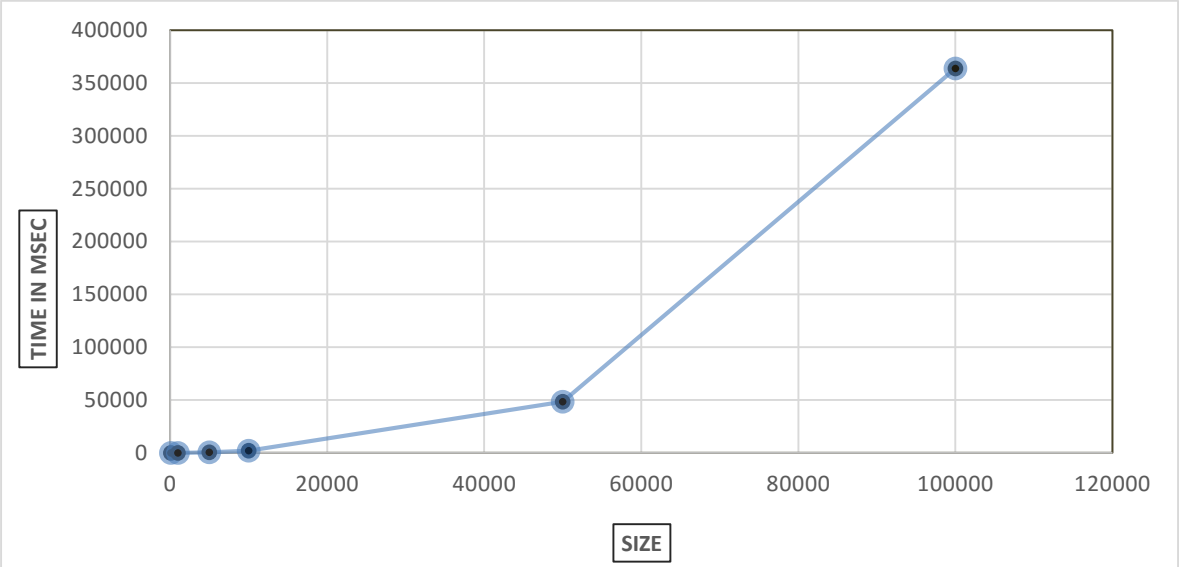


**Fig: Time duration for building primary index**

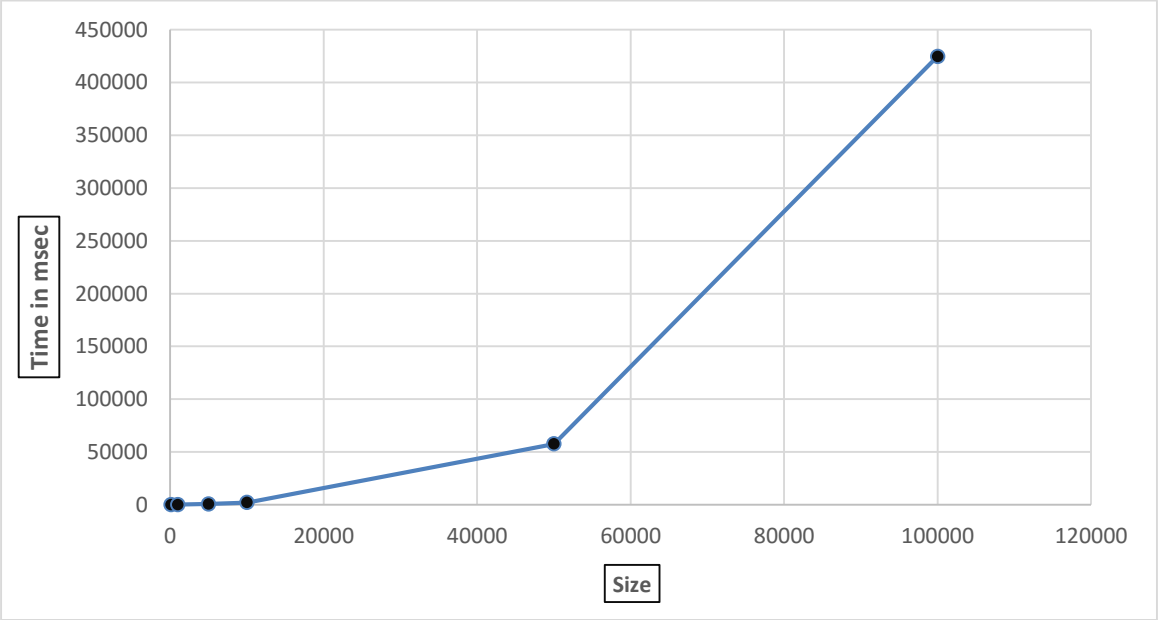


**Fig: Time duration for building secondary index**

# *CHAPTER 4*
# *RESULTS AND SNAPSHOTS*

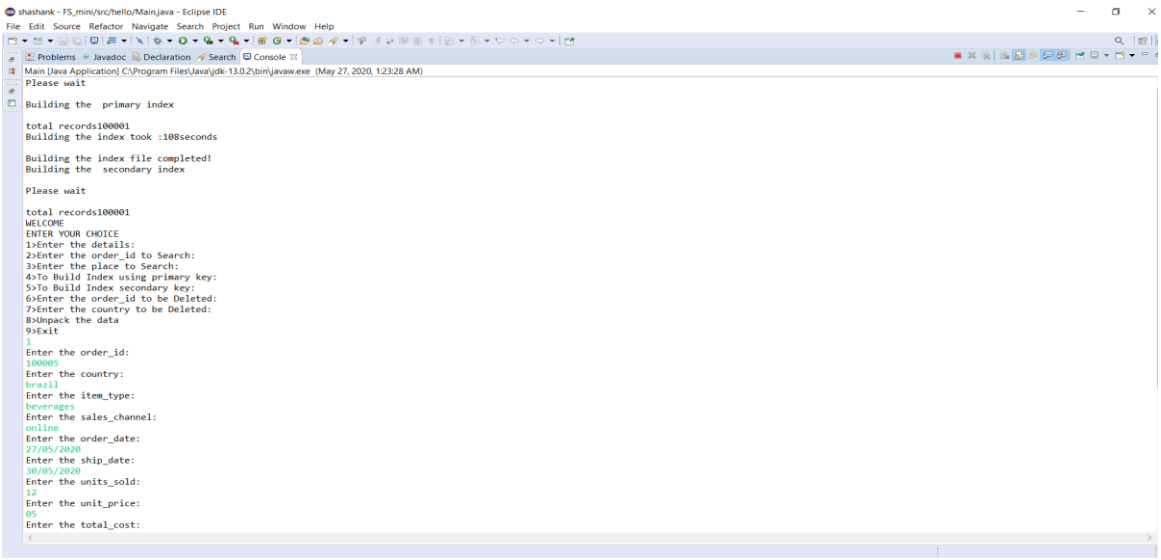


**Fig 1: Building primary index and secondary index files**

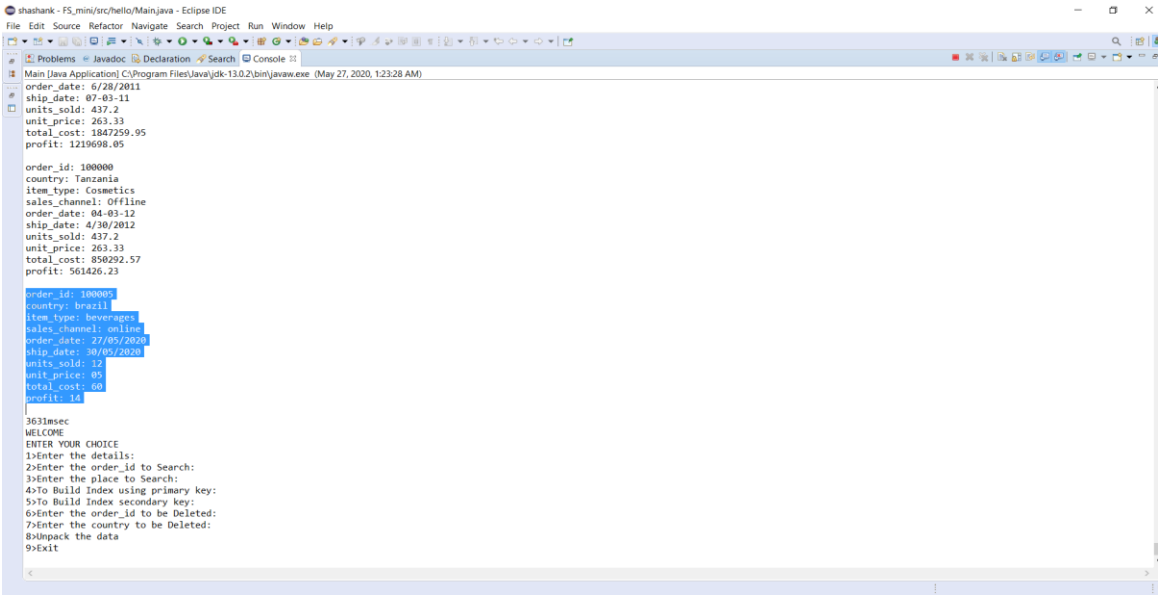


**Fig 2: Unpacked data from record file**

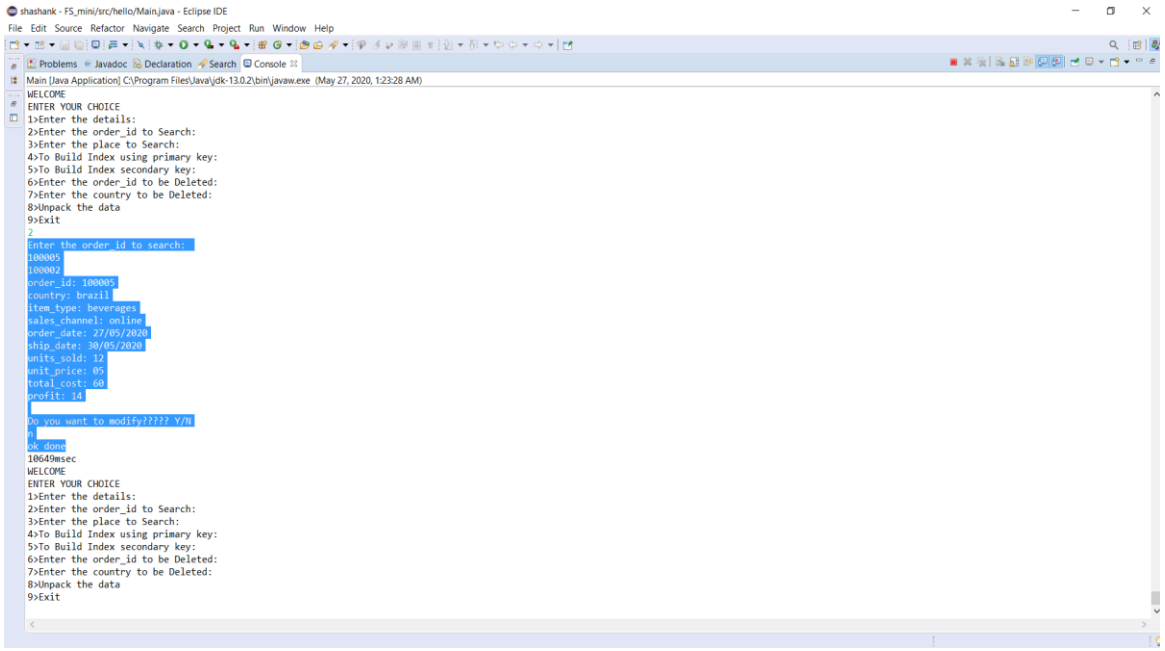


**Fig 3: Searching records using the primary key**

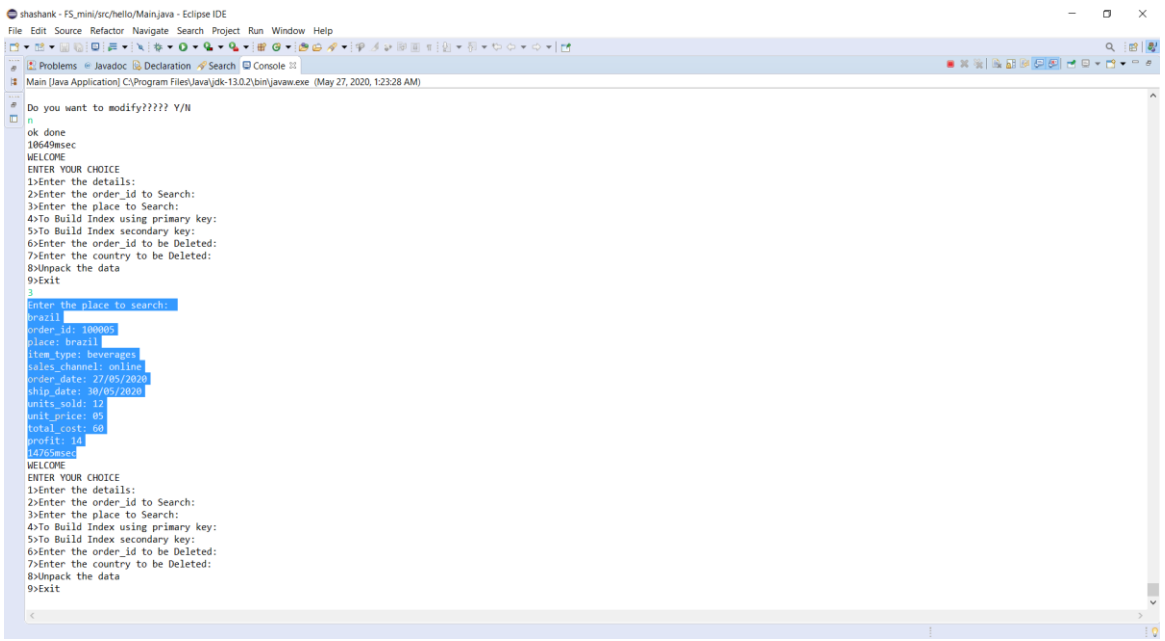


**Fig 4: Searching records using the secondary key**

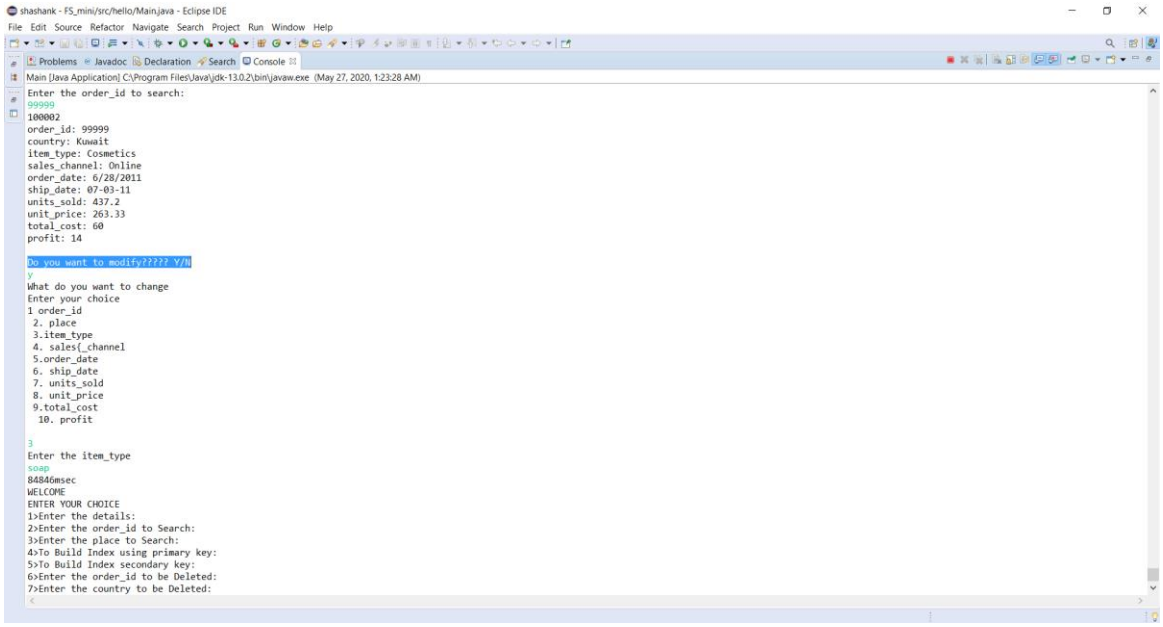


**Fig 5: Modification of record**

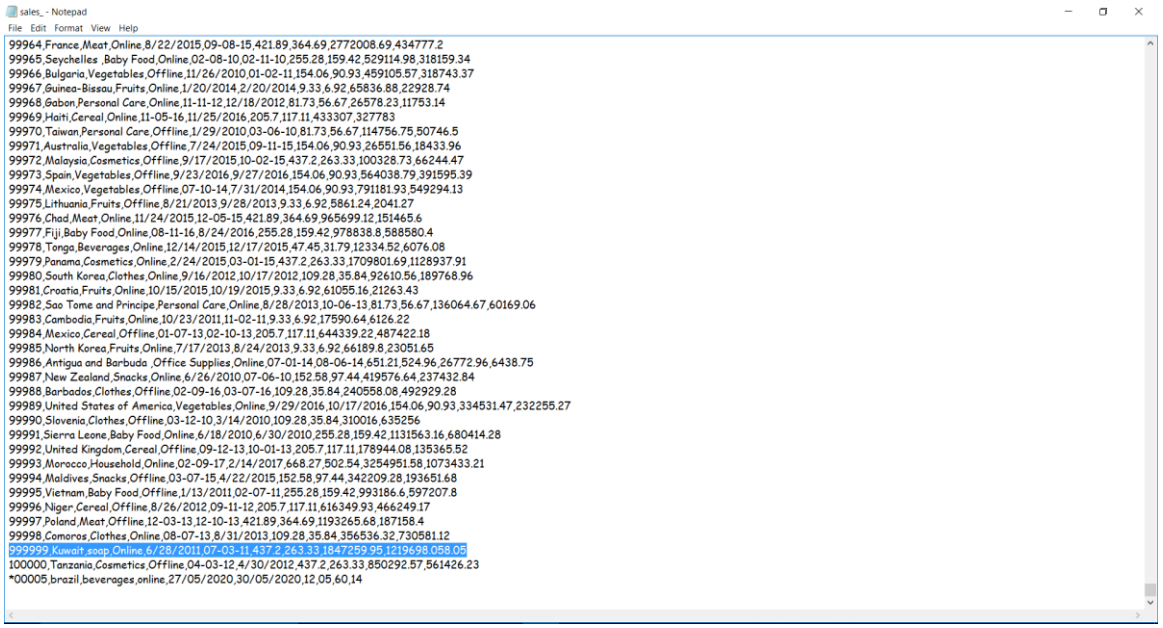


**Fig 6: Modified record in record file**

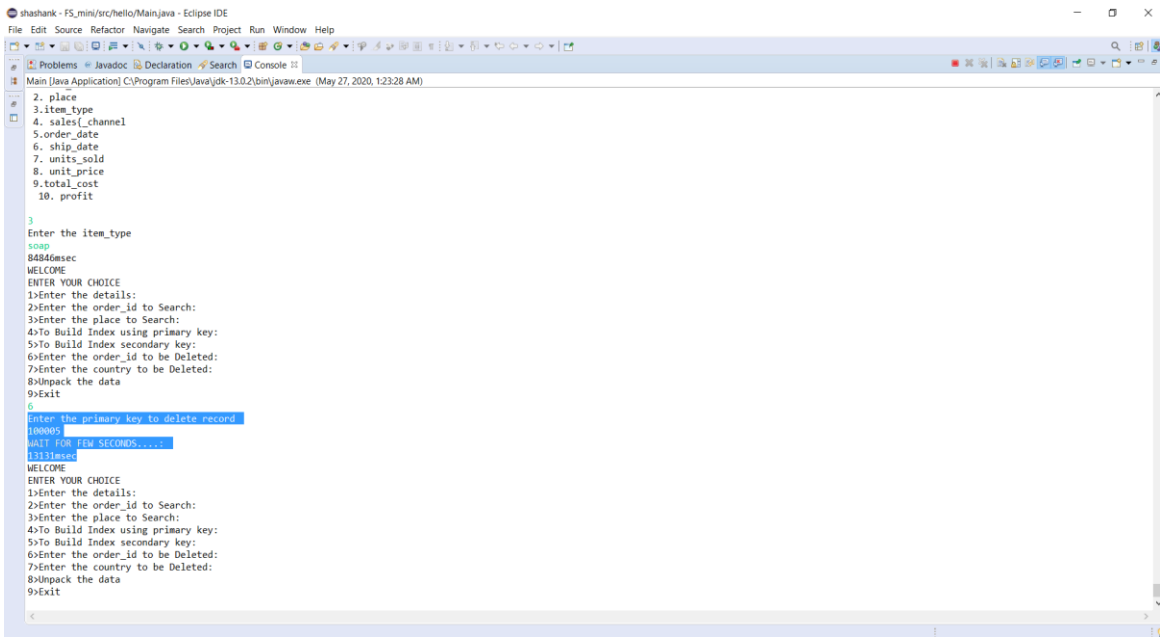


**Fig 7: Deleting the record using the primary key**

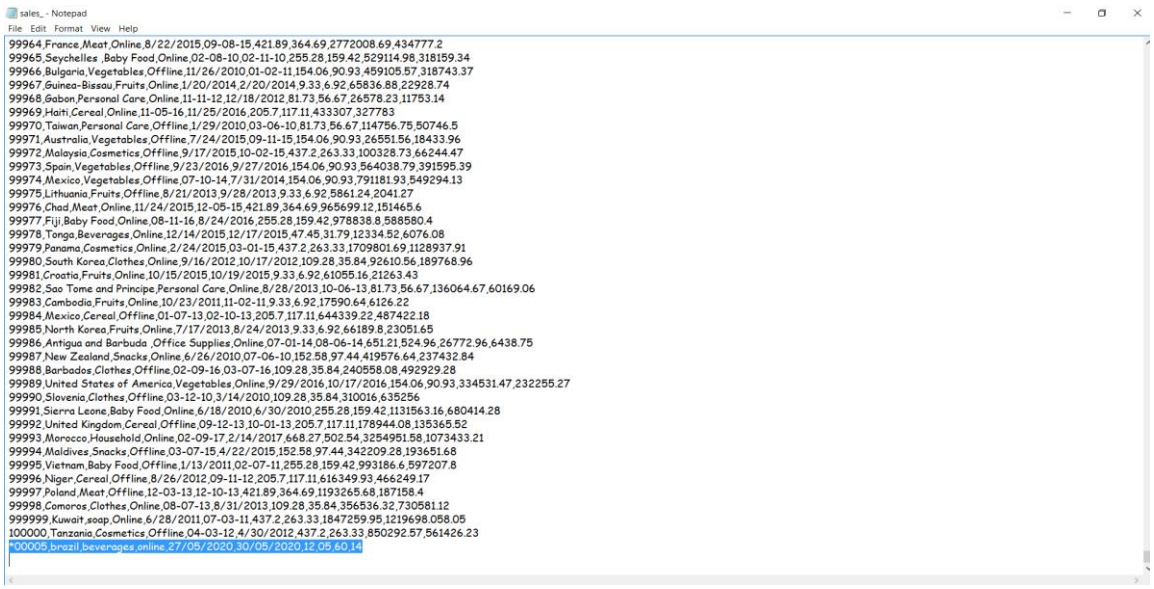


**Fig 8: Deleted record in the record file (*)**

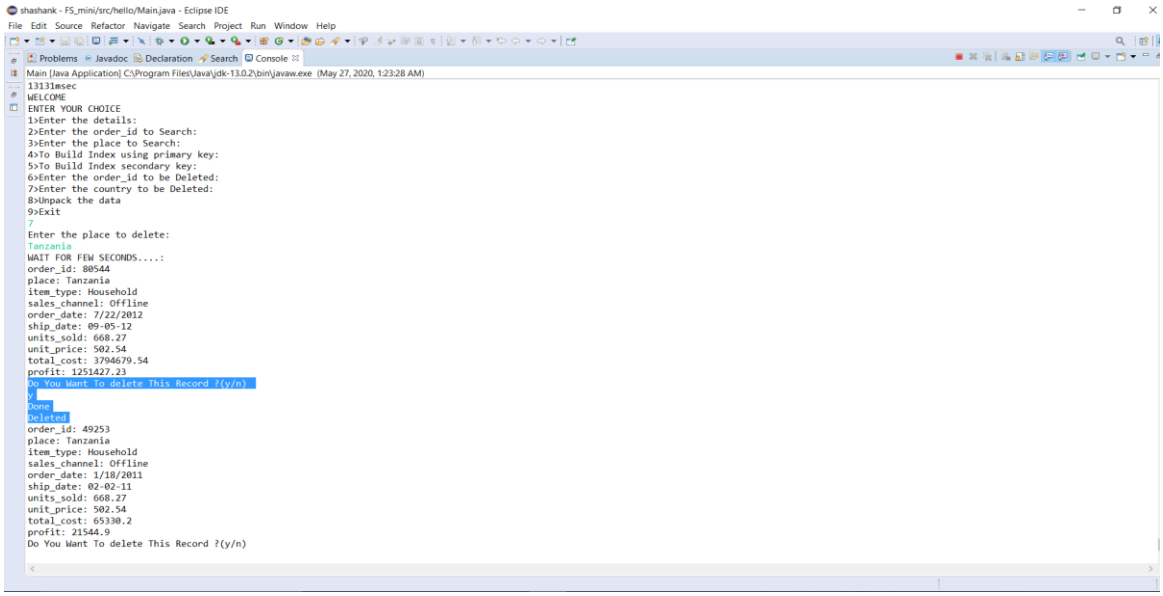


**Fig 9:Time duration for building primary index**

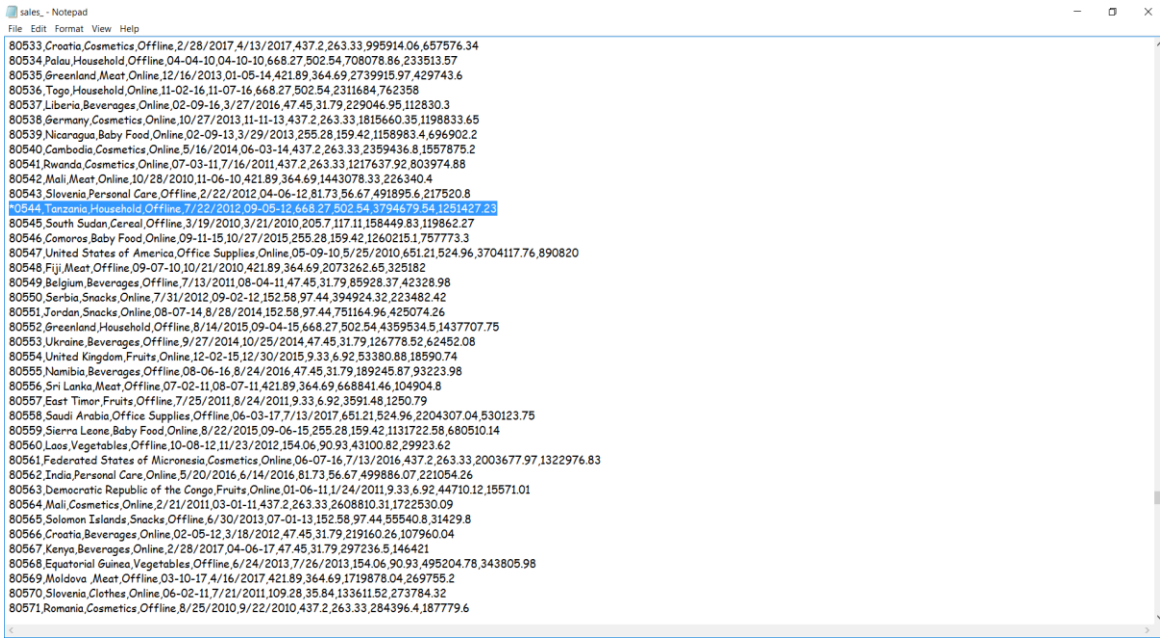


**Fig 10: Deleted record in the record file (*)**

## CONCLUSIONS:

The main objective of Indexing concept that is to optimize the performance of the number of accesses required to process the required data has been clearly achieved .

Here, Indices are used to quickly locate data without having to search every row in a database table every time a record is accessed.

## FUTURE ENHANCEMENTS:

In this project, building the index files takes up a lot of time depending on the size of the record file.

The larger the size of the record file , the more time it takes to build the index files.In case when the size of the index file is too large and exceeds the size of the main memory, then handling the index file itself will be not possible.

Therefore, we can rely on other data structures such as B-trees ,Hashing,Extensible Hashing etc to build index files for large record files efficiently and within short period of time.

# REFERENCES:

[1]https://www.youtube.com/watch?v=Hq4tAXSfnvw&list=PLhpq7_v__PAzRbm4a6GrvFs0Q6zcW_Ezd

[2] https://www.tutorialspoint.com/dbms/dbms_indexing.htm

[3] http://reddyfsproject.blogspot.com/2012/11/indexing.html

[4]https://www.kullabs.com/classes/subjects/units/lessons/notes/note-detail/5668#:~:text=Indexing%20helps%20to%20locate%20the,time%20and%20effort%20of%20employees

[5] https://www.cs.uct.ac.za/mit_notes/database/htmls/chp11.html

[6] file structures an object-oriented approach with c++ 3rd edition

[7] https://www.researchgate.net/publication/333843847_A_Study_on_Indexes_and_Index_Structures