**CS 542 – DATABASE MANAGEMENT SYSTEMS**

**INTERNALS PROGRAMMING OPTION**

**Assignment 2 – Indexing Mechanism**

By:

Arun Vadivel

Neha Satish Mahajan

**CONTENTS**

Contents

[**INTRODUCTION** 3](#_Toc434111206)

[**PURPOSE OF THE PROJECT** 3](#_Toc434111207)

[**TECHNOLOGIES INVOLVED** 3](#_Toc434111208)

[**FILES USED** 3](#_Toc434111209)

[**INDEX DESIGN STRUCTURE** 4](#_Toc434111210)

[**CLASSES AND METHODS USED** 5](#_Toc434111211)

[**ASSUMPTIONS** 6](#_Toc434111212)

[**TEST CASES** 6](#_Toc434111213)

[Test Case 1 6](#_Toc434111214)

[Test Case 2 7](#_Toc434111215)

[Test Case 3 8](#_Toc434111216)

[Test Case 4 8](#_Toc434111217)

[Test Case 5 9](#_Toc434111218)

[Test Case 6 11](#_Toc434111219)

[Test Case 7 12](#_Toc434111220)

[Test Case 8 14](#_Toc434111221)

[Test Case 9 15](#_Toc434111222)

[Test Case 10 16](#_Toc434111223)

[Test Case 11 17](#_Toc434111224)

[Test Case 12 17](#_Toc434111225)

# **INTRODUCTION**

A database management system (DBMS) is a collection of programs that enables storage, modification, and information extraction from a database. To help the databases to search for data faster, an indexing mechanism is used. 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. There are different types of indexing like B+ Trees, Hashing, etc. with each having its own advantages and disadvantages and used for different scenarios.

# **PURPOSE OF THE PROJECT**

The project is focused on understanding how an indexing mechanism works during insertion, search and deletion when working on numbers and strings. This project is based on hashing mechanisms and it uses two hashing mechanisms here, extensible hashing and static hashing for numbers and strings. Working on different scenarios in operations of the hashing helped us understand the working of hashing and how re-hash works after insert and delete operations.

# **TECHNOLOGIES INVOLVED**

Language Implemented: Java (JDK 8)

Platform used: Eclipse

# **FILES USED**

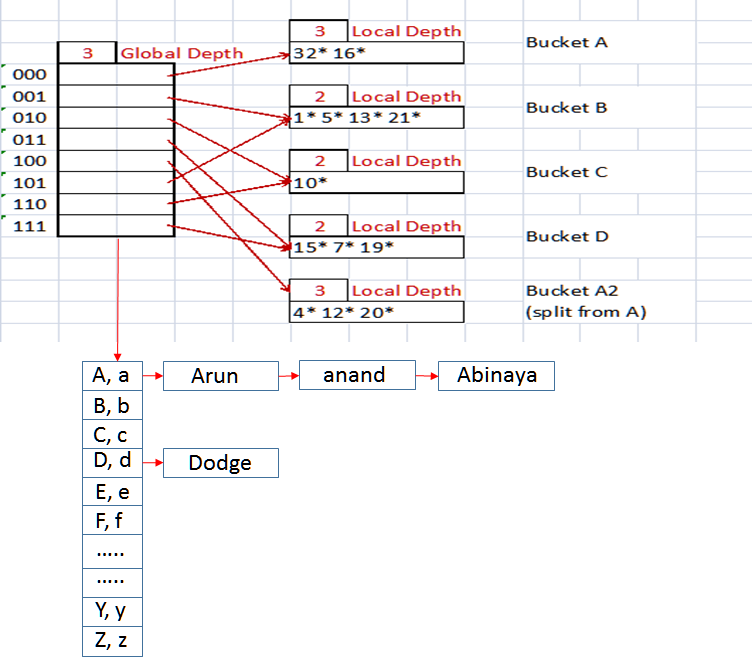
1. Main File: **Index.java**

Index.java generates the following file during the first run:

1. cs542.db
2. Global.txt
3. Local2.txt
4. Local3.txt
5. Local4.txt
6. String.txt

# **INDEX DESIGN STRUCTURE**

* The index implemented in the project is based on combining extensible and static hashing.
* Extensible Hashing is used for indexing Number keys.
* Static Hashing is used for indexing String keys.
* The global depth pointer in the extensible index points to the key in the static index.
* This strategy is implemented since the keys will mostly be numbers and rarely be strings.



**Note: Only keys are shown in the diagram above and not data.**

# **CLASSES AND METHODS USED**

**Class Hash**

1. public void First\_Run()

* Initial setup of files to execute the program. Creates the files if the program is run on a system for the first time or cleans the database if requested by the user.

1. public void Last\_Run()

* Method to commit the changes to the database

1. public void Put\_test()

* Method to read the data from the database to the main memory

1. public void Put(String key, String data)

* To insert String data

1. public String Get(String key)

* To fetch String data

1. public void Remove(char hashKey)

* To remove String data

1. public void Put(int key, String data)

* To insert number data

1. public void Put\_Level2(int key, int pos, String data,int index)

* Local Depth 2 operations during insertion

1. public void Put\_Level3(int key, int pos, String data,int index)

* Local Depth 3 operations during insertion

1. public void Put\_Level4(int key, int pos, String data,int index)

* Local Depth 4 operations during insertion

1. public void reHash(int level, int index, int pos)

* To Re-Hash after insertion

1. public void reHash\_Remove(int index)

* To Re-Hash after deletion

1. public void Remove(int key)

* To remove the key for integer data

1. public void Remove\_Level2(int key, int index)

* To remove the key for integer data in Local Depth 2

1. public void Remove\_Level3(int key, int index)

* To remove the key for integer data in Local Depth 3

1. public void Remove\_Level4(int key, int index)

* To remove the key for integer data in Local Depth 4

1. public String Get(int key)

* To fetch number data

1. public int Get\_Level2(int key, int index)

* Fetch operations in Local Depth 2

1. public int Get\_Level3(int key, int index)

* Fetch operations in Local Depth 3

1. public int Get\_Level4(int key, int index)

* Fetch operations in Local Depth 3

**Class Index – Main**

1. public static void main(String args[])

* Main function for user interaction and passing the inputs to the appropriate functions in the Class Hash.

# **ASSUMPTIONS**

1. The indexing mechanism uses two hashing techniques, extensible hashing for number keys and static hashing for string keys.
2. The extensible hashing for number has a depth up to 4 level (Local Depth 4). Thus, it can hold 64 keys for number keys.
3. The static hashing for string keys can hold up to 260 strings, 10 for each alphabet.
4. Only two data can be given to a tuple. Additional data can be given by ‘|’ for e.g. (john|fulltime|25|California) to data 2 i.e, the second entry.
5. The index will be created on the first data entered in the tuple i.e., data 1.

# **TEST CASES**

**TEST CASES FOR NUMBER DATA**

### **Test Case 1**

**Insert 4, 8, 12, 16. (At Local Depth 2)**

Is this your first run in this system (or) Do you want to reset all the indexes and data (y/n):

y

Initialization / Reset is Done

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

1

Enter the data to insert. Enter 0 to stop.

Enter data 1 (Number):

4

Enter data 2 (String):

a

Enter data 1 (Number):

8

Enter data 2 (String):

b

Enter data 1 (Number):

12

Enter data 2 (String):

c

Enter data 1 (Number):

16

Enter data 2 (String):

d

Enter data 1 (Number):

0

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

0

Do you want to commit (y/n): y

Committed.

**Data 1 in the database now: 4, 8, 12, 16**

### **Test Case 2**

**Fetch 4 from the inserted data. (At Local Depth 2)**

Is this your first run in this system (or) Do you want to reset all the indexes and data (y/n):

n

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

2

Enter the data 1 on which the index is created (Number):

4 Enter data 2 (String):

i

Local Depth 2. Position in bucket: 1. Key value: 4

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

0

Do you want to commit (y/n): n

Not Committed.

**Data 1 in the database now: 4, 8, 12, 16**

### **Test Case 3**

**Fetch 20 from the inserted data. (At Local Depth 2)**

Is this your first run in this system (or) Do you want to reset all the indexes and data (y/n):

n

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

2

Enter the data 1 on which the index is created (Number):

20

Enter data 2 (String):

e

Data not found.

**Data 1 in the database now: 4, 8, 12, 16**

### **Test Case 4**

**Remove 4 from the inserted data and check if it still exists.**

Is this your first run in this system (or) Do you want to reset all the indexes and data (y/n):

n

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

3

Enter the data 1, on which the index is created, to be removed (Number): 4

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

2

Enter the data 1 on which the index is created (Number):

4

Enter data 2 (String):

a

Data not found.

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

0

Do you want to commit (y/n): y

Committed.

**Data 1 in the database now: 8, 12, 16**

### **Test Case 5**

**Insert 4 and 20 into the existing database and check if rehashing to Local Depth 3 has been done successfully.**

Is this your first run in this system (or) Do you want to reset all the indexes and data (y/n):

n

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

1

Enter the data to insert. Enter 0 to stop.

Enter data 1 (Number):

4

Enter data 2 (String):

a

Enter data 1 (Number):

20

Enter data 2 (String):

e

Enter data 1 (Number):

0

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

2

Enter the data 1 on which the index is created (Number):

4

Enter data 2 (String):

j

Data not found.

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

2

Enter the data 1 on which the index is created (Number):

4

Enter data 2 (String):

j

Local Depth 3. Position in bucket: 1. Key value: 4

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

0

Do you want to commit (y/n): y

Committed.

1

Enter the data to insert. Enter 0 to stop.

Enter data 1 (Number):

4

Enter data 2 (String):

a

Enter data 1 (Number):

20

Enter data 2 (String):

e

Enter data 1 (Number):

0

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

2

Enter the data 1 on which the index is created (Number):

20

Enter data 2 (String):

e

Local Depth 4. Position in bucket: 3. Key value: 20

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

0

Do you want to commit (y/n): y

Committed.

**You can find from the search of 4 that re-hash has been done successfully since 4 is in Local Depth 3.**

**Also note for a different combination of data 1 and data 2, where 4 and j is given as inputs, the search fails.**

**Data 1 in the database now: 4, 8, 12, 16, 20**

### **Test Case 6**

**Insert 24, 28, 32, 36, 40 into the existing database and check if rehashing to Local Depth 4 has been done successfully.**

Is this your first run in this system (or) Do you want to reset all the indexes and data (y/n):

n

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

1

Enter the data to insert. Enter 0 to stop.

Enter data 1 (Number):

24

Enter data 2 (String):

f

Enter data 1 (Number):

28

Enter data 2 (String):

g

Enter data 1 (Number):

32

Enter data 2 (String):

h

Enter data 1 (Number):

36

Enter data 2 (String):

i

Enter data 1 (Number):

40

Enter data 2 (String):

j

Enter data 1 (Number):

0

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

2

Enter the data 1 on which the index is created (Number):

8

Enter data 2 (String):

j

Local Depth 4. Position in bucket: 1. Key value: 8

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

2

Enter the data 1 on which the index is created (Number):

32

Enter data 2 (String):

h

Local Depth 4. Position in bucket: 2. Key value: 32

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

0

Do you want to commit (y/n): y

Committed.

**You can find from the search of 8 and 32 that re-hash has been done successfully since they is in Local Depth 4.**

**Data 1 in the database now: 4, 8, 12, 16, 20, 24, 28, 32, 36, 40**

### **Test Case 7**

**Delete everything other than 20, 24, 28, 32 and 36 and check if the reverse re-hash for delete is done properly form Local Depth 4 to Local Depth 3.**

Is this your first run in this system (or) Do you want to reset all the indexes and data (y/n):

n

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

3

Enter the data 1, on which the index is created, to be removed (Number): 4

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

3

Enter the data 1, on which the index is created, to be removed (Number): 8

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

3

Enter the data 1, on which the index is created, to be removed (Number): 12

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

3

Enter the data 1, on which the index is created, to be removed (Number): 16

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

3

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

2

Enter the data 1 on which the index is created (Number):

24

Enter data 2 (String):

j

Local Depth 4. Position in bucket: 3. Key value: 24

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

Enter the data 1, on which the index is created, to be removed (Number): 40

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

0

Do you want to commit (y/n): y

Committed.

**You can find from the search of 24 that reverse re-hash has been done successfully since 24 is in Local Depth 3.**

**Data 1 in the database now: 20, 24, 28, 32 and 36**

### **Test Case 8**

**Delete everything other than 20 and 24, and check if the reverse re-hash for delete is done properly form Local Depth 3 to Local Depth 2.**

Is this your first run in this system (or) Do you want to reset all the indexes and data (y/n):

n

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

3

Enter the data 1, on which the index is created, to be removed (Number): 28

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

3

Enter the data 1, on which the index is created, to be removed (Number): 32

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

3

Enter the data 1, on which the index is created, to be removed (Number): 36

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

2

Enter the data 1 on which the index is created (Number):

20

Enter data 2 (String):

e

Local Depth 2. Position in bucket: 1. Key value: 24

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

1

1 - Insert

2 - Search

3 - Delete

2

Enter the data 1 on which the index is created (Number):

32

Enter data 2 (String):

h

Data not found.

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

0

Do you want to commit (y/n): y

Committed.

**You can find from the search of 20 that reverse re-hash has been done successfully since 20 is in Local Depth 2 and since 32 does not exist in search, it has been removed successfully.**

**Data 1 in the database now: 20 and 24**

**TEST CASES FOR STRING DATA**

### **Test Case 9**

**Insert two strings “Arun” and “anand”.**

Is this your first run in this system (or) Do you want to reset all the indexes and data (y/n):

n

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

2

1 - Insert

2 - Search

3 - Delete

1

Enter data 1 (String):

Arun

Enter data 2 (String):

Rider

Enter data 1 (String):

anand

Enter data 2 (String):

Boxer

Enter data 1 (String):

0

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

0

Do you want to commit (y/n): y

Committed.

### **Test Case 10**

**Search two strings “Arun” and “anand” and expect both should be inserted in key “a” despite the upper/lower cases.**

Is this your first run in this system (or) Do you want to reset all the indexes and data (y/n):

n

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

2

1 - Insert

2 - Search

3 - Delete

2

Enter the data 1 on which the index is created (String):

Arun

Enter data 2 (String):

Rider

Hash Key is: a

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

2

1 - Insert

2 - Search

3 - Delete

2

Enter the data 1 on which the index is created (String):

anand

Enter data 2 (String):

Boxer

Hash Key is: a

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

**Despite uppercase or lowercase starting alphabets, both “Arun” and “anand” returns ‘a’ as the hash key.**

### **Test Case 11**

**Search the string “Arun” but with a different combination for Data 2.**

Is this your first run in this system (or) Do you want to reset all the indexes and data (y/n):

n

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

2

1 - Insert

2 - Search

3 - Delete

2

Enter the data 1 on which the index is created (String):

Arun

Enter data 2 (String):

Driver

Data not found

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

0

Do you want to commit (y/n): n

Not Committed.

**As expected, fails for different combinations of inputs for the same tuple.**

### **Test Case 12**

**Delete a key and search if the data still exists.**

Is this your first run in this system (or) Do you want to reset all the indexes and data (y/n):

n

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

2

1 - Insert

2 - Search

3 - Delete

3

Enter the data 1, on which the index is created, to be removed (String): Arun

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

2

1 - Insert

2 - Search

3 - Delete

2

Enter the data 1 on which the index is created (String):

Arun

Enter data 2 (String):

Rider

Data not found

What do you want to do now?

Enter 1 for numbers

Enter 2 for Strings

Enter 0 for exit

0

Do you want to commit (y/n): y

Committed.

**The given key is deleted successfully as it is not found later during search.**