INF-2700: Database system

Assignment 3

Iver Mortensen

UiT mail: imo059@uit.no

October 31, 2023

1 Introduction

This report describes the further extension of a simple database management system. This includes adding two different natural join methods, Nested-join and Block Nested-join. The I/O performance between these implementations is measured. It also covers a hypothetical case where the database is stored in a B+-Tree, and its performance impact.

2 Technical background

2.1 Join

A join operation combines two tables based on a common field between them. The join is performed by comparing values in the common field and creating a new table containing the common field and the unique fields of the two joined tables.

2.2 Nested-join

Nested join compares one value at a time from the left table to all values of the right table.

2.3 Block Nested-join

Block Nested-join compares all records within one block at a time from the left table to all records within the right table.

3 Implementation

3.1 Block Nested-loop Join

The block Nested-loop was implemented using a quadruple nested loop. The first two loops iterates through the block number of the first table, then the block number of the second table. These loops also sets the current page the table is on by using these block numbers.
The last two loops iterates through the records within the current block of the first table, and then the records within the current block of the second table. These loops set the current record the table is on by using the current page and the index of the record.
The inner most loop also compares the value that the two current records of the tables. The value within the record that is compared is the common field of the two tables. If the values are the same, a new record containing the common field and all the unique fields of each table is added to the new joined table.

Before this nested loop is started, the common field, and the unique fields needs to be determined. This is done by first adding all the fields of the left table to the joined table. After that the right table fields are compared with these added records. If a field from the right table does not appear, the field is added. If the field does appear, we know this is the common field, and the offset of these two common fields is saved. These offsets is what we use to know what records to compare in the inner loop of the nested loop.

3.2 Nested-loop Join

The Nested-loop was implemented the exact way the Block Nested-loop, with a small change. The change that was made was to switch the second and third loop within the quadruple nested loop. This results in the nested loop to be in the order of left block, left record, right block, right record. When we look at the performance we will see the impact this change will have on I/O operations.

4 Discussion

4.1 Performance

```
--- Testing Nested-join with 10000 records ---
INFO:  Number of disk seeks/reads/writes/IOs: 11953/4170415/771/4171186

--- Testing Block Nested-join with 10000 records ---
INFO:  Number of disk seeks/reads/writes/IOs: 1667/174306/769/175075
```

Figure 1: Testing Nested-join and Block Nested-join with 10,000 records in each table.

The performance of the two join methods is measured in number of I/O operations. The test was done using two tables with a single integer common field and each table having a unique string field. Each table containing 10,000 records. For simplicity in the test, the left table's common field is identical to the right table's common field. This means a test of 10,000 record has 10,000 matches.

We can see in Figure 1 that the number of writes is similar in the two implementations. The number of seeks is over 7 times higher in the Nested-join compared to the Block Nested-join. Despite this large difference, it has a small impact on the total number of I/O operations. The main factor causing a high increase in disk operations is the number of reads. The Block Nested-join has over 170,000 reads, resulting in the biggest influence on I/O operations. In contrast to this large influence, the Nested-join had an even larger read count impact. With almost 24 times higher number of reads, putting the number of reads above 4 million. We can see high read count impact on the total number of I/Os, with Block Nested-join ending up on over 170 thousand and Nested-join at over 4 million, not far from their number of reads.

4.2 B+-Tree

Lets suppose that both tables in the join is stored in a B+-Tree organized file, where the search keys is the common attribute of the tables. This structure is similar to a Hash Join, after the two tables has been hashed. When the Hash Join hashes the two tables it creates partitions of each table, where the partition of one table contains the same values as the same partition of the second table, illustrated in Figure 2. The second step of the Hash join, is to loop through the corresponding partitions of each table, and finding the matching records.

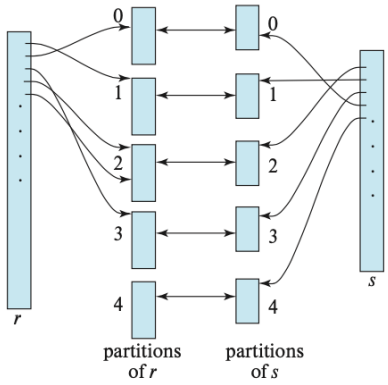

Figure 2: Illustration of Hash join partitions.

Since our hypothetical tables are stored in B+-tree organized files with the search keys as the common attribute, we get a similar structure to the second step of the Hash Joined tables (Figure 2). We can then use the second step of Hash Join on our hypothetical tables, using the leafs with the common keys as the partitions.

This approach could reduce the number of I/O operations as the data is partitioned. Block Nested-join can result in looking through the entire right table to find a match, causing many I/O operations. Since our hypothetical approach has stored all the potential matches to a to a record in a single partition, only the partition needs to be read.

4.3 Pinned pages

When I increase the amount of records used when testing, the implementation started to add duplicates of records from later blocks. When I increased the number of records again, the program tried to get records outside of a later page than the one it should have been on. I assumed this issue was because the new page I wanted to look up did not have space in memory. This would cause the program to look in memory where it would think the page was, but a previous page would be there. I fixed this issue by unpinning pages from memory, so the memory wouldn't get full, and new pages could be looked up successfully.

5 Conclusion

In conclusion, the implementation was successful in implementing the two natural join methods. The performance test illustrated the difference in I/O operation count between the two joins. The disk operation efficiency was evidently better in the Block Nested-join implementation. The performance could be further improved by incorporating a B+-Tree data structure for the database.

6 References

[2] Database System Concepts, 7th Edition, A. Silberschatz, H. Korth and S. Sudarshan, (McGraw Hill).