Principles of Database Systems(H)
Southern University of Science and Technology
Mengxuan Wu
12212006

Project 1 Report

Mengxuan Wu

1 Introduction

DBMS(Database Management System) is a powerful tool that can perform data queries and updates efficiently. To better understand the high efficiently of DBMS, this report presents experiments comparing the performance of DBMS and direct file access and manipulation.

To be precise, this report compares the execution time of retrieval test and update test between three different combinations: a Java client application using PostgreSQL, a DataGrip client application using PostgreSQL and a Java program performing direct file access and manipulation.

2 Experiment Setup

2.1 Environment

Item	Model
OS	Windows 11 22H2
CPU	Intel Core i9-12900H
Memory	16 GB 4800 MHz DDR5
Storage	1 TB SSD

2.2 Data

The data used in this experiment comes from The Movies Dataset on Kaggle. The dataset is 943.76 MB in size and contains metadata for 45459 movies, including title, cast, crew, ratings, overview and other attributes.

2.3 Experiment Design

The experiment consists of two parts: retrieval test and update test.

2.3.1 Retrieval Test

The retrieval test is designed to compare the execution time of finding all movies with the word "XXX" in their titles. This repost uses the 34.45 MB movies_metadata.csv file in the dataset.

The retrieval test is conducted 10 times for each combination and the average execution time is calculated. When calculating the execution time using a Java client application, the execution time is calculated from the time the SQL statement is executed to the time the result set is received. When calculating the execution time using a DataGrip client application, this report uses the execution time shown in the console. When calculating the execution time using a Java program performing direct file access and manipulation, the execution time is calculated from the time when the program accesses the file to the time the result set is received. Also, the time of establishing the connection to the database is not included in the execution time.

2.3.2 Update Test

The update test is designed to compare the execution time of replacing all "To" in person name to "TTOO". This repost uses the 189.92 MB credits.csv file in the dataset.

The update test is conducted 10 times for each combination and the average execution time is calculated. The way of calculating the execution time is the same as the retrieval test, except that there is no result set in the update test.

3 Results

Note: All the results are in milliseconds.

3.1 Retrieval Test

Test Case	DataGrip + PostgreSQL	Java + PostgreSQL	Java + File
1	42	33	189
2	38	32	177
3	35	33	179
4	34	31	181
5	35	36	185
6	33	30	186
7	38	35	181
8	40	40	162
9	39	34	175
10	33	37	179
Avg	36.7	34.1	179.4

3.2 Update Test

Test Case	DataGrip + PostgreSQL	Java + PostgreSQL	Java + File
1	2313	1928	5360
2	2560	1703	5642
3	2004	1707	5604
4	2386	1729	5389
5	2049	1714	5592
6	1978	1720	5512
7	2137	1742	5674
8	2376	2003	5429
9	2085	1721	5679
10	2090	1659	5607
Avg	2197.8	1762.6	5548.8

4 Conclusion

In retrieval test, the execution time using direct file access and manipulation is approximately four times longer than using PostgreSQL. In update test, it's approximately three times longer.

Some possible reasons are listed below:

1. PostgreSQL uses caching to store frequently accessed data in memory. Since accessing memory is much faster than accessing disk, when we run the same query again, the execution time will decrease. This could be seen in both test where the execution time of the first test case is often longer than the rest.

2. PostgreSQL creates indexes to locate data more efficiently. Since indexes are also stored in memory, they are faster to access than file in disks, which also contribute to high efficiently of PostgreSQL.

3. By creating a table with different attributes, PostgreSQL can scan and search for data in certain column, while direct file access and manipulation has to traverse the whole file.

5 Extra Experiment

5.1 Experiment Design

In addition to the experiments above, this report conducts two extra experiments: a traverse test and a retrieval test without PostgreSQL features by applying a function.

5.1.1 Traverse Test

In this experiment, the traverse time of PostgreSQL and direct file access and manipulation is compared.

The test is conducted 10 times for each file and the average traverse time is calculated. When calculating the traverse time of PostgreSQL, this report uses the execution time of

retrieving the whole table. When calculating the traverse time of Java program performing direct file access and manipulation, this report uses the time of reading the whole file by each line, splitting each line into a string array that corresponds to each attribute and write each line into result set.

Since the result set would be large, the time of receiving the result set cannot be ignored in this experiment. However, in the combination of Java client application and PostgreSQL, the time of receiving the result set cannot be measured or excluded. Therefore, this experiment is not conducted in this combination.

5.1.2 Retrieval Test without PostgreSQL Features

In this experiment, the features of PostgreSQL are deliberately broken by applying a function to the column name in the 'where' clause. By doing so, the table content will be first modified by the function and then filtered by the 'where' clause. Therefore, caching and indexing will not be used in the filtering process.

When calculating the retrieval time, this report compares the execution time to find all titles with the word 'man' in them with a normal SQL statement and a SQL statement with the function applied to the column name as mentioned above. Since no comparison with file access and manipulation is needed in this experiment, the test is only conducted in DataGrip. The test is conducted 10 times and the average time is calculated. When calculating the execution time, this report uses the execution time shown in the console.

5.2 Results

Note: All the results are in milliseconds.

5.2.1 Traverse Test

Test Case	credits.csv		movies_metadata.csv	
	DataGrip + PostgreSQL	Java + File	DataGrip + PostgreSQL	Java + File
1	23	412	5	108
2	16	474	5	116
3	12	527	7	121
4	11	421	8	107
5	10	414	5	112
6	12	421	5	106
7	12	443	6	109
8	11	445	4	107
9	9	424	4	114
10	11	412	4	109
Avg	12.7	439.3	5.3	110.9

5.2.2 Retrieval Test without PostgreSQL Features

Test Case	Normal	Without PostgreSQL Features
1	20	70
2	19	66
3	17	81
4	18	69
5	18	70
6	17	72
7	16	80
8	21	66
9	18	92
10	18	84
Avg	18.2	75

5.3 Conclusion

The traverse time of PostgreSQL is much shorter than direct file access and manipulation. The possible reasons could be caching which are the same as the reasons listed in the conclusion of the previous experiments.

The retrieval time without PostgreSQL features is approximately four times longer than normal retrieval time. Therefore, we can conclude that the features of PostgreSQL contribute a lot to the high efficiency of PostgreSQL.

A Code

This appendix contains SQL code used in this experiment. The Java code is available in attachment.

Listing 1: SQL for Retrieval Test

```
SELECT title
FROM movies_metadata
WHERE title LIKE '%XXX%';
```

Listing 2: SQL for Update Test

```
UPDATE credits
SET crew = REPLACE(crew, 'To', 'TTOO')
WHERE crew LIKE E'%\'name\': \'%To%\'%';
UPDATE credits
SET "cast" = REPLACE("cast", 'To', 'TTOO')
WHERE "cast" LIKE E'%\'name\': \'%To%\'%';
```

Listing 3: SQL for Traverse Test(1)

```
SELECT * FROM credits;
```

Listing 4: SQL for Traverse Test(2)

```
SELECT * FROM movies_metadata;
```

Listing 5: SQL for Retrieval Test without PostgreSQL Features(1)

```
SELECT title
FROM movies_metadata
WHERE UPPER(title) LIKE '% MAN %'
OR UPPER(title) LIKE 'MAN %'
OR UPPER(title) LIKE '% MAN';
```

Listing 6: SQL for Retrieval Test without PostgreSQL Features(2)

```
SELECT title
FROM movies_metadata
WHERE title LIKE '% Man %'
OR title LIKE 'Man %'
OR title LIKE '% Man'
OR title LIKE '% man %'
OR title LIKE 'man %'
OR title LIKE '% man';
```