Week#9 Introduction to SQLite and AndroBench

Hyuksoo Yeo

2016312761

1. INTRODUCTION

Run Androbench by changing the SQLite environment. For example, we can change journal_mode, page_cache size, page size, locking mode, and synchronous mode. Then observe how performance changes. We can measure the performance by execution time. After that, present the experimental results by table or graph and analyze.

2. METHODS

For this experiment, build SQLite by downloading SQLite database source code and using make command. Then record and analyze the SQLite trace file. The record is composed of real, user, sys. Change the SQLite environment and repeat the recording and analyzing.

3. Performance Evaluation

3.1 Experimental Setup

System setup:

Type	Specification
OS	Ubuntu 20.04.3 LTS
CPU	Intel® Core™ i3-9100F CPU @ 3.60GHz
Memory	16GB
Kernel	5.11.0-27-generic
Data Device	Western Digital WD Blue 500GB
Log Device	Western Digital WD Blue 500GB

Benchmark setup:

Type	Configuration
DB size	1GB (10 warehouse)
Buffer Pool Size	300MB (30% of DB size)
Benchmark Tool	tpcc-mysql
Runtime	1200s
Connections	8

3.2 Experimental Results

Change the journal mode:

```
PRAGMA page_size;
PRAGMA page_size=4096;
PRAGMA foreign_keys;

/* Change journal mode here */
PRAGMA journal_mode=del;
/* Change cache size here */
PRAGMA cache_size=2000;
/* Change synchoronus mode */
PRAGMA synchronous=1;
/* Change locking mode */
PRAGMA locking_mode=NORMAL;

PRAGMA journal_mode;
PRAGMA synchronous;
```

```
real    0m25.202s
user    0m0.046s
sys     0m4.303s
```

```
PRAGMA page_size;
PRAGMA page_size=4096;
PRAGMA foreign_keys;

/* Change journal mode here */
PRAGMA journal_mode=off;
/* Change cache size here */
PRAGMA cache_size=2000;
/* Change synchoronus mode */
PRAGMA synchronous=1;
/* Change locking mode */
PRAGMA locking_mode=NORMAL;

PRAGMA journal_mode;
PRAGMA synchronous;
```

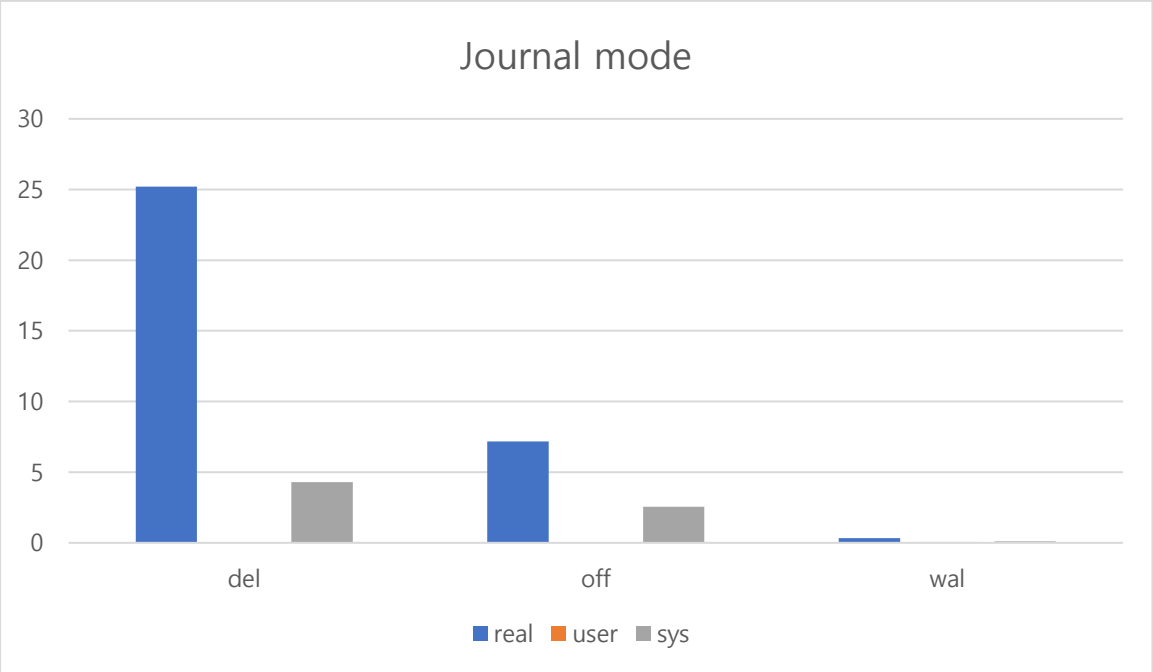
```
real    0m7.175s
user    0m0.014s
sys     0m2.561s
```

```
PRAGMA page_size;
PRAGMA page_size=4096;
PRAGMA foreign_keys;

/* Change journal mode here */
PRAGMA journal_mode=wal;
/* Change cache size here */
PRAGMA cache_size=2000;
/* Change synchoronus mode */
PRAGMA synchronous=1;
/* Change locking mode */
PRAGMA locking_mode=NORMAL;

PRAGMA journal_mode;
PRAGMA synchronous;
```

```
real    0m0.324s
user    0m0.050s
sys     0m0.113s
```



In the delete mode, the rollback journal is deleted at the conclusion of each transaction. Off mode is non-mode. Wal mode is a mode which using write-ahead log instead of a rollback journal to implement transactions.

Change the page_cache size:

```
PRAGMA page_size;
PRAGMA page_size=4096;
PRAGMA foreign_keys;

/* Change journal mode here */
PRAGMA journal_mode=del;
/* Change cache size here */
PRAGMA cache_size=2000;
/* Change synchoronus mode */
PRAGMA synchronous=1;
/* Change locking mode */
PRAGMA locking_mode=NORMAL;

PRAGMA journal_mode;
PRAGMA synchronous;
```

```
real    0m25.202s
user    0m0.046s
sys     0m4.303s
```

```
PRAGMA page_size;
PRAGMA page_size=4096;
PRAGMA foreign_keys;

/* Change journal mode here */
PRAGMA journal_mode=del;
/* Change cache size here */
PRAGMA cache_size=1000;
/* Change synchoronus mode */
PRAGMA synchronous=1;
/* Change locking mode */
PRAGMA locking_mode=NORMAL;

PRAGMA journal_mode;
PRAGMA synchronous;
```

```
real    0m23.946s
user    0m0.014s
sys     0m3.750s
```

```
PRAGMA page_size;
PRAGMA page_size=4096;
PRAGMA foreign_keys;

/* Change journal mode here */
PRAGMA journal_mode=del;
/* Change cache size here */
PRAGMA cache_size=500;
/* Change synchoronus mode */
PRAGMA synchronous=1;
/* Change locking mode */
PRAGMA locking_mode=NORMAL;

PRAGMA journal_mode;
PRAGMA synchronous;
```

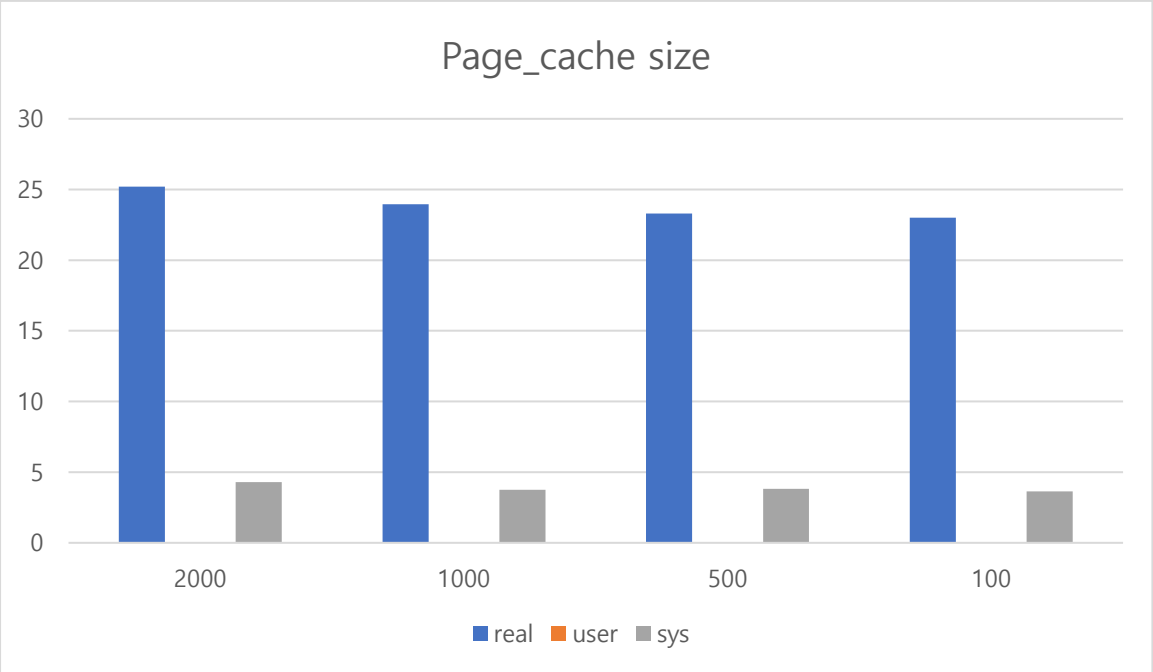
```
real    0m23.292s
user    0m0.017s
sys     0m3.818s
```

```
PRAGMA page_size;
PRAGMA page_size=4096;
PRAGMA foreign_keys;

/* Change journal mode here */
PRAGMA journal_mode=del;
/* Change cache size here */
PRAGMA cache_size=100;
/* Change synchoronus mode */
PRAGMA synchronous=1;
/* Change locking mode */
PRAGMA locking_mode=NORMAL;

PRAGMA journal_mode;
PRAGMA synchronous;
```

```
real    0m23.012s
user    0m0.022s
sys     0m3.628s
```



Smaller the page_cache size, the total execution time get decreased.

Change the page size:

```
PRAGMA page_size;
PRAGMA page_size=16384;
PRAGMA foreign_keys;

/* Change journal mode here */
PRAGMA journal_mode=del;
/* Change cache size here */
PRAGMA cache_size=2000;
/* Change synchoronus mode */
PRAGMA synchronous=1;
/* Change locking mode */
PRAGMA locking_mode=NORMAL;

PRAGMA journal_mode;
PRAGMA synchronous;
```

```
real    0m22.250s
user    0m0.004s
sys     0m3.665s
```

```
PRAGMA page_size;
PRAGMA page_size=8192;
PRAGMA foreign_keys;

/* Change journal mode here */
PRAGMA journal_mode=del;
/* Change cache size here */
PRAGMA cache_size=2000;
/* Change synchoronus mode */
PRAGMA synchronous=1;
/* Change locking mode */
PRAGMA locking_mode=NORMAL;

PRAGMA journal_mode;
PRAGMA synchronous;
```

```
real    0m23.683s
user    0m0.022s
sys     0m3.930s
```

```
PRAGMA page_size;
PRAGMA page_size=4096;
PRAGMA foreign_keys;

/* Change journal mode here */
PRAGMA journal_mode=del;
/* Change cache size here */
PRAGMA cache_size=2000;
/* Change synchoronus mode */
PRAGMA synchronous=1;
/* Change locking mode */
PRAGMA locking_mode=NORMAL;

PRAGMA journal_mode;
PRAGMA synchronous;
```

```
real    0m25.202s
user    0m0.046s
sys     0m4.303s
```

```
PRAGMA page_size;
PRAGMA page_size=2048;
PRAGMA foreign_keys;

/* Change journal mode here */
PRAGMA journal_mode=del;
/* Change cache size here */
PRAGMA cache_size=2000;
/* Change synchoronus mode */
PRAGMA synchronous=1;
/* Change locking mode */
PRAGMA locking_mode=NORMAL;

PRAGMA journal_mode;
PRAGMA synchronous;
```

```
real    0m23.813s
user    0m0.010s
sys     0m3.929s
```

```
/
PRAGMA page_size;
PRAGMA page_size=1024;
PRAGMA foreign_keys;

/* Change journal mode here */
PRAGMA journal_mode=del;
/* Change cache size here */
PRAGMA cache_size=2000;
/* Change synchoronus mode */
PRAGMA synchronous=1;
/* Change locking mode */
PRAGMA locking_mode=NORMAL;

PRAGMA journal_mode;
PRAGMA synchronous;
```

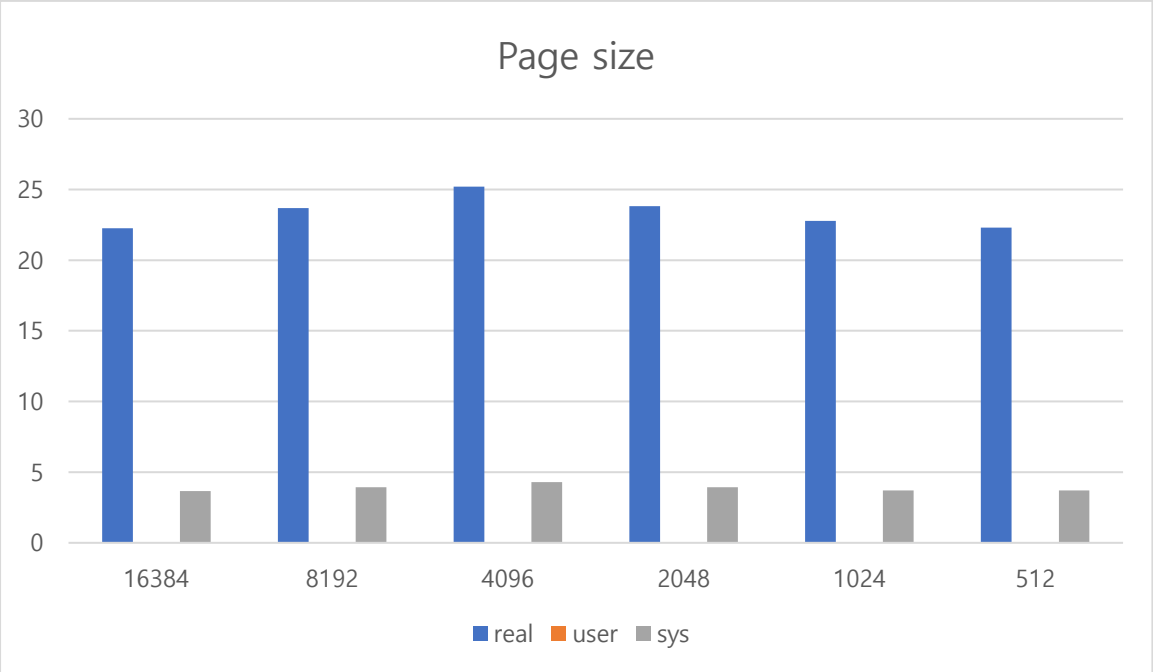
```
real    0m22.788s
user    0m0.007s
sys     0m3.697s
```

```
PRAGMA page_size;
PRAGMA page_size=512;
PRAGMA foreign_keys;

/* Change journal mode here */
PRAGMA journal_mode=del;
/* Change cache size here */
PRAGMA cache_size=2000;
/* Change synchoronus mode */
PRAGMA synchronous=1;
/* Change locking mode */
PRAGMA locking_mode=NORMAL;

PRAGMA journal_mode;
PRAGMA synchronous;
```

```
real    0m22.303s
user    0m0.000s
sys     0m3.709s
```



If the page size is 4096, the total execution time is highest.

Change locking mode:

```
PRAGMA page_size;
PRAGMA page_size=4096;
PRAGMA foreign_keys;

/* Change journal mode here */
PRAGMA journal_mode=del;
/* Change cache size here */
PRAGMA cache_size=2000;
/* Change synchoronus mode */
PRAGMA synchronous=1;
/* Change locking mode */
PRAGMA locking_mode=NORMAL;

PRAGMA journal_mode;
PRAGMA synchronous;
```

```
real    0m25.202s
user    0m0.046s
sys     0m4.303s
```

```
PRAGMA page_size;
PRAGMA page_size=4096;
PRAGMA foreign_keys;

/* Change journal mode here */
PRAGMA journal_mode=del;
/* Change cache size here */
PRAGMA cache_size=2000;
/* Change synchoronus mode */
PRAGMA synchronous=1;
/* Change locking mode */
PRAGMA locking_mode=EXCLUSIVE;

PRAGMA journal_mode;
PRAGMA synchronous;
```

```
real    0m18.788s
user    0m0.000s
sys     0m3.208s
```

When the locking mode is EXCLUSIVE, both real time and sys time are decreased compared with NORMAL.

Change the synchronous mode:

```
PRAGMA page_size;
PRAGMA page_size=4096;
PRAGMA foreign_keys;

/* Change journal mode here */
PRAGMA journal_mode=del;
/* Change cache size here */
PRAGMA cache_size=2000;
/* Change synchoronus mode */
PRAGMA synchronous=0;
/* Change locking mode */
PRAGMA locking_mode=NORMAL;

PRAGMA journal_mode;
PRAGMA synchronous;
```

```
real    0m0.723s
user    0m0.102s
sys     0m0.271s
```

```
PRAGMA page_size;
PRAGMA page_size=4096;
PRAGMA foreign_keys;

/* Change journal mode here */
PRAGMA journal_mode=del;
/* Change cache size here */
PRAGMA cache_size=2000;
/* Change synchoronus mode */
PRAGMA synchronous=1;
/* Change locking mode */
PRAGMA locking_mode=NORMAL;

PRAGMA journal_mode;
PRAGMA synchronous;
```

```
real    0m25.202s
user    0m0.046s
sys     0m4.303s
```

```
PRAGMA page_size;
PRAGMA page_size=4096;
PRAGMA foreign_keys;

/* Change journal mode here */
PRAGMA journal_mode=del;
/* Change cache size here */
PRAGMA cache_size=2000;
/* Change synchoronus mode */
PRAGMA synchronous=2;
/* Change locking mode */
PRAGMA locking_mode=NORMAL;

PRAGMA journal_mode;
PRAGMA synchronous;
```

```
real    0m29.269s
user    0m0.008s
sys     0m4.656s
```

4. Conclusion

In this experiment, I learned how to run androbench by changing the SQLite environment parameters. I still don't know the meaning of each parameters but the change of execution time could be easily understood.

5. REFERENCES

[1] https://github.com/meeeejin/SWE3033-F2021/tree/main/week-10

Proposal>

1. Compare with the SQLite and client-server RDBMS (e.g., MySQL, RocksDB, and Oracle)

SQLite 는 다양한 플랫폼에서 실행 가능한 것이 장점입니다. 그리고 SQLite 는 트랜잭션과 원자성 동작을 지원하기 때문에 프로그램 충돌이 발생하더라도 데이터베이스가 손상될 위험이 없습니다. 또한 C 언어로 작성된 라이브러리를 활용할 수 있다면 다른 언어를 사용할 필요가 없다. Mysql 과 같은 client-server RDBMS 와의 차이점은 동시 사용자의 수이다. SQLite 는 동시 사용자가 한 명인 애플리케이션에 적합하지만, Mysql 의 경우 여러 명의 동시 사용자에 대응할 수 있도록 설계되었다. 또한 Mysql 은 클러스터 및 수평 확장 기능이 있지만 SQLite 는 불가능하다. 또한 SQLite 의 데이터 형식은 NULL, INTEGER, TEXT 정도로 적다. 반면 Mysql 은 날짜와 시간을 위한 전용 데이터 형식, 정수와 실수를 위한 정밀한 형식 등 많은 형식이 추가로 존재한다.

2. How SQLite supports cross-platfrom database

VFS (Virtual Filesystem) 는 SQLite API 와 소통하고, 실제 파일 시스템과도 별도로 소통하면서 SQLite 와 파일 시스템을 별도의 시스템으로 연결을 해준다. 따라서 read 나 write 과 관련된 메소드를 운영체제와 관계없이 통일된 방법을 사용할 수 있게 되면서 운영체제에 관계없이 모두 호환성을 갖게 해준다.

3. Compare the SQLite and Filesystem

SQLite reads and writes small blobs such as thumbnail images 35% faster than the same blobs can be read or written to individual files(filesystem) on disk. Furthermore, a single SQLite database holding 10KB blobs uses about 20% less disk space than storing in individual files.

The performance difference arises because when working from an SQLite database, the open() and close() system calls are invoked only once, while invoked once for each blob when using individual files. The overhead will go greater than using the database.