Week8 WAL / Manifest

Made by Suhwan Shin, Isu Kim

E-Mail: tlstnghks77@naver.com

Content

WAL Testing with RocksDB

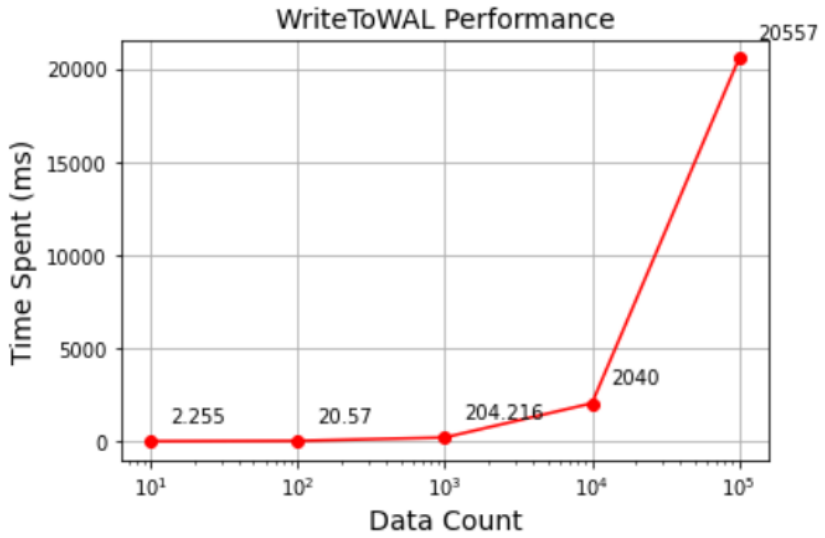
- Was interested in WAL's real performance.
- Tried making a --disable-wal option for LevelDB, but failed.
- Thus, using RocksDB for WAL enabling and disabling.
- Decided to make a simple C++ program that tests this topic.

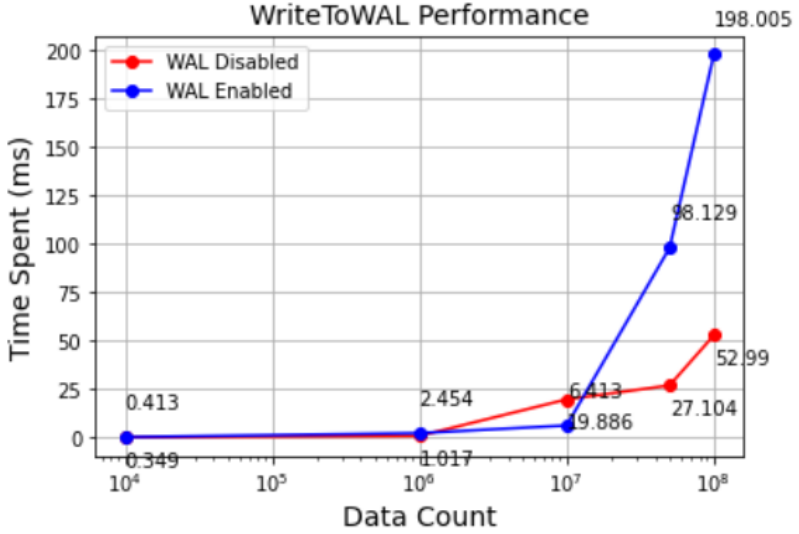
```
drwxrwxr-x 4 gooday2die gooday2die 4096  8월  21 22:46 .
drwxrwxr-x 6 gooday2die gooday2die 4096  8월  21 22:45 ..
drwxrwxr-x 8 gooday2die gooday2die 4096  8월  21 22:47 .git
-rw-rw-r-- 1 gooday2die gooday2die  297  8월  21 22:45 CMakeLists.txt
-rw-rw-r-- 1 gooday2die gooday2die 1065  8월  21 22:45 LICENSE
drwxrwxr-x 2 gooday2die gooday2die 4096  8월  21 22:46 build
-rw-rw-r-- 1 gooday2die gooday2die 1845  8월  21 22:45 db_test.cpp
-rw-rw-r-- 1 gooday2die gooday2die 1435  8월  21 22:45 db_test.h
-rw-rw-r-- 1 gooday2die gooday2die  636  8월  21 22:45 rocksdb_get.cpp
-rw-rw-r-- 1 gooday2die gooday2die 1217  8월  21 22:45 rocksdb_put.cpp
-rw-rw-r-- 1 gooday2die gooday2die  571  8월  21 22:45 utils.cpp
-rw-rw-r-- 1 gooday2die gooday2die  130  8월  21 22:45 utils.h
gooday2die@flagship:~/projects/School/2022_0.5/LevelDB/wal_test/WAL-Testing$
```

Code can be found in the link below
https://github.com/gooday2die/WAL-Testing

WAL Testing with RocksDB – Hypothesis

- WAL Enabled *put* will be guaranteed and stored properly.
- WAL Disabled *put* will have potential loss of data (or all).
- With prior knowledge:





WAL Testing with RocksDB – Design (Put)

putRocksdb

- **Put** key - value pair into RocksDB.
- Has option of enabling or disabling WAL.
- Key represents iteration count.
- Values are made up of randomly generated string length of 20.
- Will put key – value pair 1,000,000 times.

manualFlushPutRocksdb

- Same as putRocksdb but will manually flush in every 1000 iterations.

```
[+] Putting 69178 : 5ptNhRF9ZEGX16FTtso4
[+] Putting 69179 : BIWng3ZveKMj7F6neKwE
[+] Putting 69180 : YAjJEyl8pXA0pelUiLNK
[+] Putting 69181 : dj2iw6VaQQowYXGnUztJ
[+] Putting 69182 : W3HJh0nNLAgwtifpoAOD
[+] Putting 69183 : YC97iNuAMlRqoiATixp2
[+] Putting 69184 : 8VwzBbnxkBAILKN1hHB2
[+] Putting 69185 : 2asoG2Gxy5z4Yv3hVoeD
[+] Putting 69186 : xnVJ7tKm8Uo94fvLfBId
[+] Putting 69187 : EFhm8jRdV6qTtKmyB4kJ
[+] Putting 69188 : YXQbAMwpVCRkR6UZnwDJ
[+] Putting 69189 : 0lmtLWrUaZo74CiEWc22
[+] Putting 69190 : oTmDXEmLAxeAxOlIuqlS
[+] Putting 69191 : QZXSjDfGnhGZ82mfEX0P
[+] Putting 69192 : ScXP0ZgsPRJnyqGf2tvp
[+] Putting 69193 : aBNgB9JQeKn7uKUvrAlF
[+] Putting 69194 : Z40VtEBv76ifG5LPDfpr
[+] Putting 69195 : xayrtRkkbWx9ayeRCpMK
[+] Putting 69196 : u3xA8JZLwNDrv9ioYRX8
[+] Putting 69197 : xUHVQvxdjHvdKsnT9Km3
[+] Putting 69198 : fxub5bPd2ulxP2SnwNOf
[+] Putting 69199 : fJIzA3QHLDLlAFaDoxrq
[+] Putting 69200 : scnFdE2ZbREGiUEqVe8q
[+] Putting 69201 : pTpygNBVL0JBd5QEJQnu
[+] Putting 69202 : rz9YRNOwzWkpxalcxw5G
[+] Putting 69203 : xORYTpmkGXf7Umfv92p8
[+] Putting 69204 : YZvW7g63bBJYXj4zYojo
```

WAL Testing with RocksDB – Design (Get)

getRocksdb

- Opens generated RocksDB from putRocksdb.
- **Get** keys starting from 0 to a key that has empty value.

```
[+] Found 69179 : BIWng3ZveKMj7F6neKwE
[+] Found 69180 : YAjJEyl8pXA0pelUiLNK
[+] Found 69181 : dj2iw6VaQQowYXGnUztJ
[+] Found 69182 : W3HJh0nNLAgwtifpoAOD
[+] Found 69183 : YC97iNuAMlRqoiATixp2
[+] Found 69184 : 8VwzBbnxkBAILKNlhHB2
[+] Found 69185 : 2asoG2Gxy5z4Yv3hVoeD
[+] Found 69186 : xnVJ7tKm8Uo94fvLfBId
[+] Found 69187 : EFhm8jRdV6qTtKmyB4kJ
[+] Found 69188 : YXQbAMwpVCRkR6UZnwDJ
[+] Found 69189 : 0lmtLWrUaZo74CiEWc22
[+] Found 69190 : oTmDXEmLAxeAxOlIuqlS
[+] Found 69191 : QZXSjDfGnhGZ82mfEX0P
[+] Found 69192 : ScXP0ZgsPRJnyqGf2tvp
[+] Found 69193 : aBNgB9JQeKn7uKUvrAlF
[+] Found 69194 : Z40VtEBv76ifG5LPDfpr
[+] Found 69195 : xayrtRkkbWx9ayeRCpMK
[+] Found 69196 : u3xA8JZLwNDrv9ioYRX8
[+] Found 69197 : xUHVQvxdjHvdKsnT9Km3
[+] Found 69198 : fxub5bPd2ulxP2SnwNOf
[+] Found 69199 : fJIzA3QHLDLlAFaDoxrq
[+] Found 69200 : scnFdE2ZbREGiUEqVe8q
[+] Found 69201 : pTpygNBVL0JBd5QEJQnu
[+] Found 69202 : rz9YRNOwzWkpxalcxw5G
[+] Found 69203 : xORYTpmkGXf7Umfv92p8
[+] Found 69204 : YZvW7g63bBJYXj4zYojo
[+] Found 69205 : LOunAXiHZVO62HaAygDZ
```

WAL Testing with RocksDB – Design

- Execute **putRocksdb**
- Kill putRocksdb using *CTRL* + *C* in Linux.

```
[+] Putting 36346 : Kg2vyQXzn9Wv5F8KeGg0
[+] Putting 36347 : wGYVqJm69vHTbHPXhuWV
[+] Putting 36348 : 20O7DWQrk6rhKNC8gyEn
[+] Putting 36349 : rTHSlexSWSvYSJgen6VY
^C
```

- **Get** stored values from RocksDB using **getRocksdb.**

```
[+] Found 36346 : Kg2vyQXzn9Wv5F8KeGg0
[+] Found 36347 : wGYVqJm69vHTbHPXhuWV
[+] Found 36348 : 20O7DWQrk6rhKNC8gyEn
[+] Found 36349 : rTHSlexSWSvYSJgen6VY
[+] Found 36350 : ANFSiPaNLmACDPcy3ZPY
[+] Found 36351 :
```

- Does have offset of 1 or 2 entries since *CTRL* + *C* has delay of killing process.

WAL Testing with RocksDB – Results : WAL Enabled

WAL Enabled

- Could **get** all data that were **put** before into RocksDB

```
[+] Found 36343 : v6nb3kx0UmOG7Hiv8xZE
[+] Found 36344 : Cs2Fii5uvUXqbIQe2NcX
[+] Found 36345 : 7ylFFR8LMiaYYcnHIsBD
[+] Found 36346 : Kg2vyQXzn9Wv5F8KeGg0
[+] Found 36347 : wGYVqJm69vHTbHPXhuWV
[+] Found 36348 : 20O7DWQrk6rhKNC8gyEn
[+] Found 36349 : rTHSlexSWSvYSJgen6VY
[+] Found 36350 : ANFSiPaNLmACDPcy3ZPY
[+] Found 36351 :
gooday2die@flagship:~/projects/School/2022_0.5/LevelDB/wal_t
gooday2die@flagship:~/projects/School/2022_0.5/LevelDB/wal_t
```

```
[+] Putting 36340 : n3Fmf2oST3bXiSwpsI2f
[+] Putting 36341 : foZunLZZPfhCiwyOxkoQ
[+] Putting 36342 : lNvUorHg8HJn5sgsDFPd
[+] Putting 36343 : v6nb3kx0UmOG7Hiv8xZE
[+] Putting 36344 : Cs2Fii5uvUXqbIQe2NcX
[+] Putting 36345 : 7ylFFR8LMiaYYcnHIsBD
[+] Putting 36346 : Kg2vyQXzn9Wv5F8KeGg0
[+] Putting 36347 : wGYVqJm69vHTbHPXhuWV
[+] Putting 36348 : 20O7DWQrk6rhKNC8gyEn
[+] Putting 36349 : rTHSlexSWSvYSJgen6VY
^C
```

- WAL definitely guarantees **put** action into RocksDB and does not lose any data even if shutdown abruptly.

WAL Testing with RocksDB – Results : WAL Disabled

WAL Disabled

- Could not **get** any data that were **put** before into RocksDB

```
gooday2die@flagship:~/projects/School/2022_0.5/LevelDB/wal_t
est/build$
gooday2die@flagship:~/projects/School/2022_0.5/LevelDB/wal_t
est/build$ ./getRocksdb
WAL Disabled : 1
[+] Found 0 :
gooday2die@flagship:~/projects/School/2022_0.5/LevelDB/wal_t
est/build$
```

```
[+] Putting 114860 : CNYtcafRApSux1kwgt9K
[+] Putting 114861 : ytEThmrekKhufFnIpShz
[+] Putting 114862 : F7sB8c7mTG4R7Ism3kPl
[+] Putting 114863 : 26fiLSyAtd96klHqbMc4
[+] Putting 114864 : agThxMSy4rj4vOkGoiQf
[+] Putting 114865 : JXm3Zlt8OTByAee5y630
[+] Putting 114866 : vk3q8n4uUUanzKpYLgfj
[+] Putting 114867 : AqiISMNPQONJ8Q7EEB8g
^C
```

- Something is strange!
- Lost **all** data

```
drwxr-xr-x 2 gooday2die gooday2die  4096  8월  21 23:23 .
drwxrwxr-x 4 gooday2die gooday2die  4096  8월  21 23:23 ..
-rw-r--r-- 1 gooday2die gooday2die     0  8월  21 23:23 000006.log
-rw-r--r-- 1 gooday2die gooday2die    16  8월  21 23:23 CURRENT
-rw-r--r-- 1 gooday2die gooday2die    37  8월  21 23:23 IDENTITY
-rw-r--r-- 1 gooday2die gooday2die     0  8월  21 23:23 LOCK
-rw-rw-r-- 1 gooday2die gooday2die 23548  8월  21 23:23 LOG
-rw-rw-r-- 1 gooday2die gooday2die 21439  8월  21 23:23 LOG.old.1661091792925634
-rw-r--r-- 1 gooday2die gooday2die    59  8월  21 23:23 MANIFEST-000005
-rw-r--r-- 1 gooday2die gooday2die  5293  8월  21 23:23 OPTIONS-000008
```

WAL Testing with RocksDB – Results : Manual WAL Flush

WAL Enabled & Manual Flush

- Could **get** all data that were **put** before into RocksDB

```
[+] Found 6349 : rnTOhHoYEnuUfr4wWphP
[+] Found 6350 : J5BNmE2DeSDWDesuufS8
[+] Found 6351 : SNc5CgzhVg4nlFAXUCj8
[+] Found 6352 : cwcpaVhU8AcYVFdftcMN
[+] Found 6353 : IRAlgIZ8SIH4CrrmKWFS
[+] Found 6354 :
```

```
[+] Putting 6348 : IsxZJhnntWOn3g5hRQQl
[+] Putting 6349 : rnTOhHoYEnuUfr4wWphP
[+] Putting 6350 : J5BNmE2DeSDWDesuufS8
[+] Putting 6351 : SNc5CgzhVg4nlFAXUCj8
[+] Putting 6352 : cwcpaVhU8AcYVFdftcMN
^C
```

- Enabling WAL and manual flush is in fact **meaningless**.
- Since enabling WAL will automatically generate state that can already recover data from being lost.

WAL Testing with RocksDB – Results : Manual WAL Flush

WAL Disabled & Manual Flush

- Could **get** partial data that were **put** before into RocksDB

```
[+] Found 1994 : Z7xLP6vJmtHE0vop63B7
[+] Found 1995 : ofl8AjZ49oXivUlIYwcL
[+] Found 1996 : orXmmLZsOkxCPzIZgqbp
[+] Found 1997 : e6VZaXr7RRQDJxx3IWue
[+] Found 1998 : Grodo4ASukFYql7PGxWh
[+] Found 1999 : Owtftqh9NblbSZDFdNhY
[+] Found 2000 : 7x6wgCLw7rbTlS9cJole
[+] Found 2001 :
```

```
[+] Putting 2220 : 0ravXLWYbql0IW9M3pi5
[+] Putting 2221 : OWvPee3vie7iThc006Xb
[+] Putting 2222 : uIcCmjWoZFtvjmKLPLG7
[+] Putting 2223 : xNnQ2PQ2TxbNDBXyv4mU
[+] Putting 2224 : HdN0QfJp0ZwxujNu7lwa
[+] Putting 2225 : jVyuhTscXc4oGRme65T4
^C
gooday2die@flagship:~/projects/School/20
```

- In every 1000 iterations, we manually flush WAL.
- Thus will flush at 2000th key.
- This will store data till 2000th key, but will lose from 2001th key.

```
void db_test::flush() {
    FlushOptions fo = FlushOptions();
    this->db->Flush(fo);
}
```

WAL Testing with RocksDB – Conclusion

Summarization

WAL Type	Speed	Data Integrity	Scenario
WAL Disabled	1.843s	Losing All	Speed demanding, but data is not critical.
WAL Enabled	3.604s	Saving All	Data is very critical.
Manual Flush & WAL Enabled [0]	40.502s	Saving All	Meaningless.
Manual Flush & WAL Disabled [0]	39.128s	Partial Loss	In compromisable situations, but tuning is required.
Manual Flush & WAL Enabled [1]	7.773s	Saving All	Meaningless.
Manual Flush & WAL Disabled [1]	5.611s	Partial Loss	In compromisable situations, but tuning is required.

- Did not print out putting values for removing IO delays.
- Execution time measured by time from Linux.
- Put values of 1,000,000 entries.

[0] Manual flush in every 1,000 entries.
[1] Manual flush in every 10,000 entries.

VersionSet

- The entire set of versions is maintained in a VersionSet
- Current : The newest version (Older version may be kept or not)

- Status
  - LogAndApply() // Apply *edit to the current version to form a new descriptor
  - Recover() // Recover the last saved descriptor from persistant storage

- Function
  - LogNumber() // Return current log file number
  - PrevLogNumber() // Return log file number currently being compacted OR 0
  - …

VersionEdit

- Function
  - SetComparatorName()
  - SetLogNumber()
  - SetPrevLogNumber()
  - SetNextFile()
  - SetLastSequence()
  - Addfile() // MetaData
  - Removefile() // MetaData
  - …

NewDB

- NewDB
  - VersionEdit::SetComparatorName → Comparator
  - VersionEdit::SetLogNumber → log_number (0)
  - VersionEdit::SetNextFile → next_file_number (2)
  - VersionEdit::SetLastSequence → last_sequence (0)

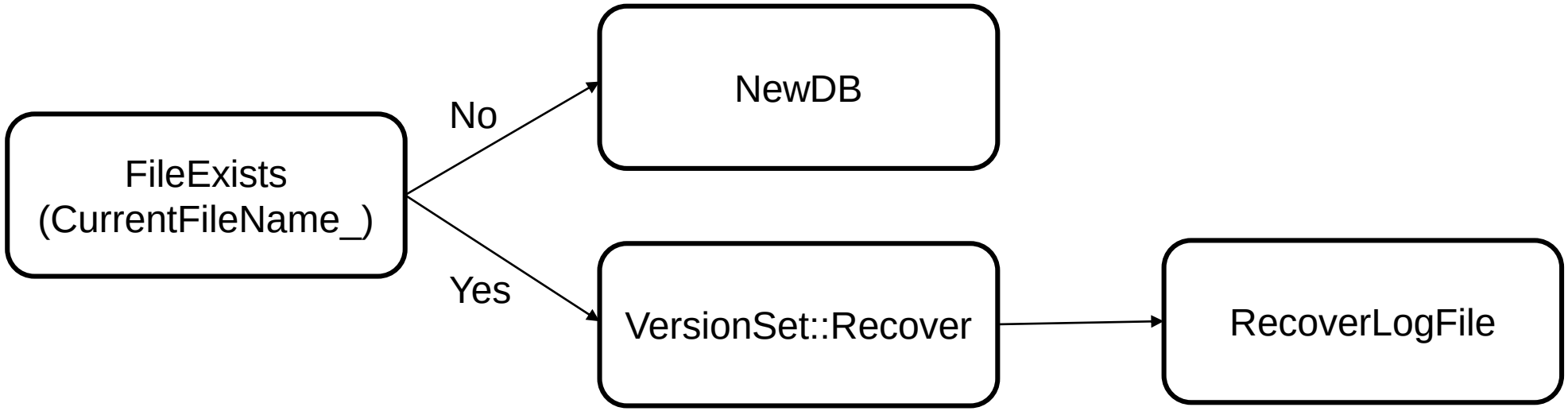
  - Make manifest file
    - NewWritableFile(manifest, &file)

  - Create Log(file)
    - AddRecord() → EmitPyhsicalRecord()

      // Checks Record Type and writes data using EmitPhysicalRecord.

Recover

- DBImpl::Recover (VersionEdit* edit, bool* save_manifest) // Recover the descriptor
- VersionSet::Recover (bool* save_manifest) // Recover the last saved descriptor



Recover from all newer log files than the ones named in the descriptor

Recover

- RecoverLogFile()
  - Open the log file
  - Create the log reader ⟶ log::Reader
  - Read all the records and add to a memtable

1) DBImpl::Recover → VersionSet::Recover / DBImpl::NewDB
2) DBImpl::Recover → VersionSet::AddLiveFiles // Add all files listed in any live version
3) DBImpl::Recover → VersionSet::MarkFileNumberUsed // Manually update the file number
4) DBImpl::Recover → VersionSet::SetLastSequence

➢ return Status::OK

CompactMemTable

- Compact the in-memory write buffer to disk.
- Switches to a new log-file/memtable and writes a new descriptor iff successful.

1. Save the contents of the memtable as a new Table

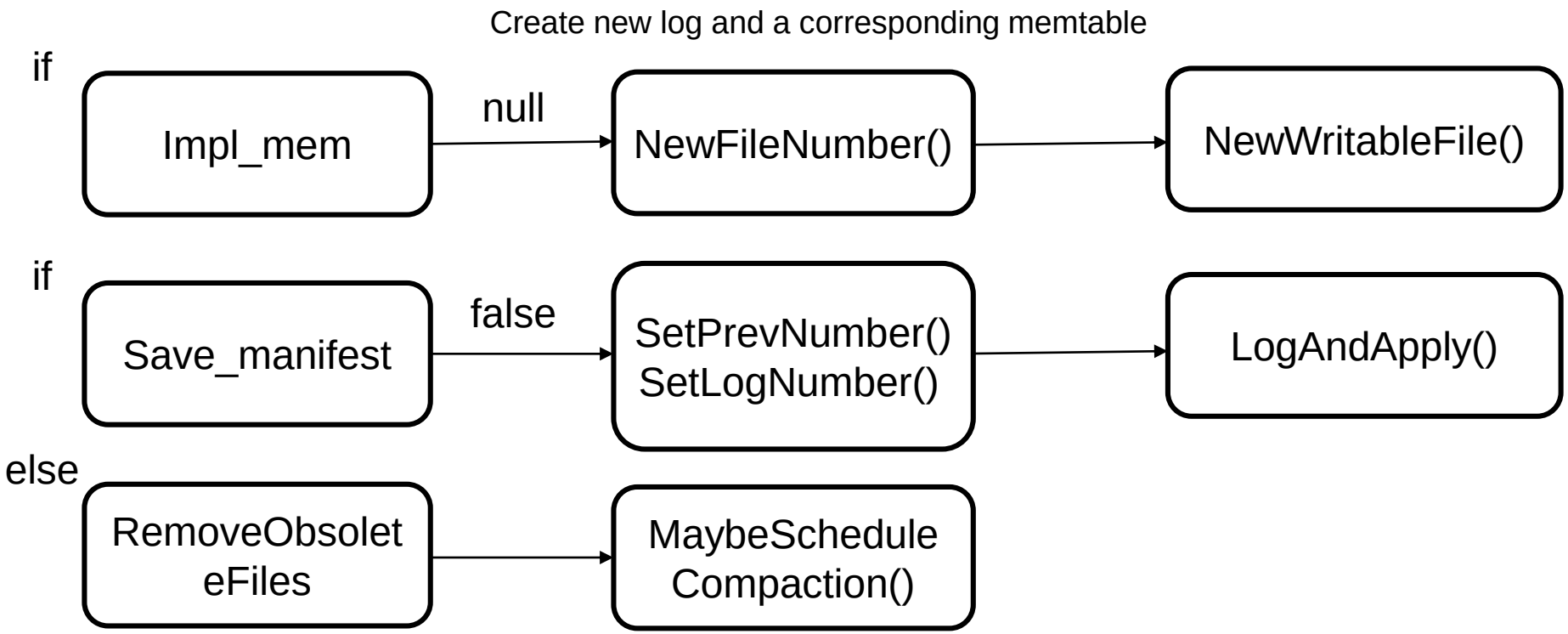
    - WriteLevel0Table() : Manage MetaData

2. Replace immutable memtable with the generated Table

    - SetPrevLogNumber(0)
    - SetLogNumber(logfile_num) : Ealier log no longer needed
    - LogAndApply()

DB::Open

1. DBImpl::Recover (VersionEdit* edit, bool* save_manifest (=false))



Question