

# LevelDB-Study

Team\_Cache Code Analysis

Made by Subin Hong, Seungwon Ha

E-Mail: zed6740@dankook.ac.kr, 12gktmddnjs@naver.com

# Contents

1. Overall flow chart
2. Cache – Code flow
3. After Analysis

- Overall flow chart
- Code flow
  - Read, Insert, Delete

# How to Understand

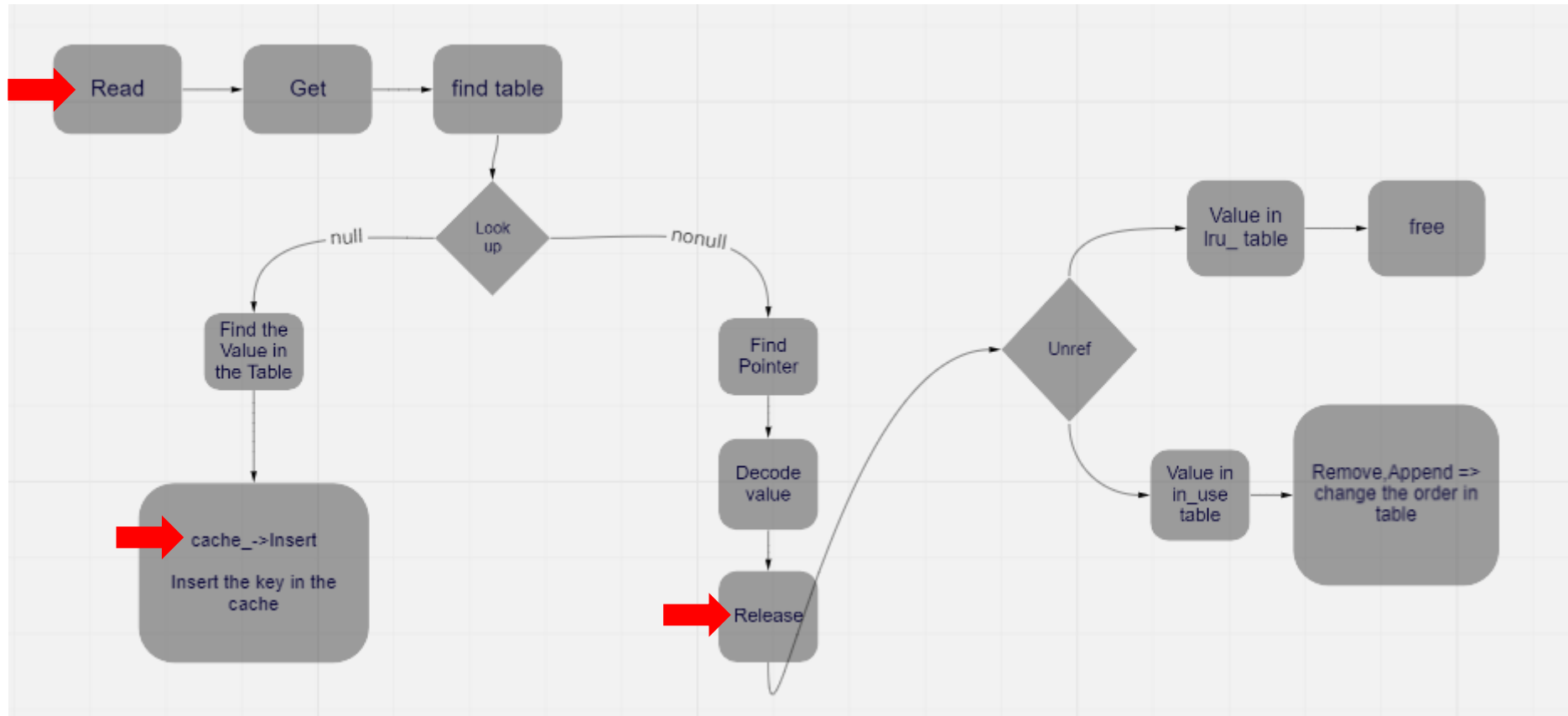
```
LEVELDB_RELEASE
util > cache.cc > {} leveldb > {} 'anonymous-namespace' > Insert(const Slice &, uint32_t, void *, size_t, void*) (const Slice &, void *, void *)
iterator.cc
merger.cc
merger.h
table_builder.cc
table_test.cc
table.cc
two_level_iterator.cc
two_level_iterator.h
third_party
util
.vscodes
arena_test.cc
arena.cc
arena.h
bloom_test.cc
bloom.cc
cache_test.cc
cache.cc 2
coding_test.cc
coding.cc
coding.h
comparator.cc
crc32c_test.cc
crc32c.cc
crc32c.h
env_posix_test_help...
env_posix_test.cc
env_posix.cc
env_test.cc
env_windows_test_h...
env_windows_test.cc
env_windows.cc

201 lru_prev = &lru;
202 in_use_next = &in_use;
203 in_use_prev = &in_use;
204 }
205
206 ~LRUCache() {
207     assert(in_use_next == &in_use); // Error if caller has an unreleased handle
208     for (LRUHandle* e = lru_next; e != &lru;) {
209         LRUHandle* next = e->next;
210         assert(e->in_cache);
211         e->in_cache = false;
212         assert(e->refs == 1); // Invariant of lru_list.
213         Unref(e);
214         e = next;
215     }
216 }
217
218 void LRUCache::Ref(LRUHandle* e) {
219     if (e->refs == 1 && e->in_cache) { // If on lru_list, move to in_use_list.
220         LRU_Remove(e);
221         LRU_Append(&in_use, e);
222     }
223     e->refs++;
224 }
225
226 void LRUCache::Unref(LRUHandle* e) {
227     assert(e->refs > 0);
228     e->refs--;
229     if (e->refs == 0) { // Deallocate.
230         assert(!e->in_cache);
231         (*e->deleter)(e->key(), e->value);
232         free(e);
233     } else if (e->in_cache && e->refs == 1) {
234         // No longer in use; move to lru_list.
235         LRU_Remove(e);
236     }
237 }
```

```
assam.dankook.ac.kr - PuTTY
TOTAL TIME : FUNCTION
===== Back-trace =====
2.839 us : (1) leveldb::GLOBAL_N_1::LRUCache::Lookup
2.839 us : (1) leveldb::GLOBAL_N_1::ShardedLRUCache::Lookup
2.839 us : (1) leveldb::TableCache::FindTable
2.839 us : (1) leveldb::TableCache::NewIterator
2.839 us : (1) leveldb::BuildTable
2.839 us : (1) leveldb::DBImpl::WriteLevel0Table
2.839 us : (1) leveldb::DBImpl::RecoverLogFile
2.839 us : (1) leveldb::DBImpl::Recover
2.839 us : (1) leveldb::DB::Open
2.839 us : (1) leveldb::Benchmark::Open
2.839 us : (1) leveldb::Benchmark::Run
2.839 us : (1) main
4.893 ms : (1000) leveldb::GLOBAL_N_1::LRUCache::Lookup
4.893 ms : (1000) leveldb::GLOBAL_N_1::ShardedLRUCache::Lookup
2.519 ms : (1000) leveldb::GLOBAL_N_1::LRUCache::Lookup
2.519 ms : (1000) leveldb::GLOBAL_N_1::ShardedLRUCache::Lookup
===== Call Graph =====
7.415 ms : (2001) leveldb::GLOBAL_N_1::LRUCache::Lookup
1.939 ms : (2001) leveldb::MutexLock::MutexLock
1.643 ms : (2001) leveldb::port::Mutex::Lock
1.363 ms : (2001) std::mutex::lock
1.076 ms : (2001) __pthread_mutex_lock
108.791 us : (2001) __pthread_active_p
165.988 us : (2001) pthread_mutex_lock
2.302 ms : (2001) leveldb::GLOBAL_N_1::HandleTable::Lookup
1.963 ms : (2001) leveldb::GLOBAL_N_1::HandleTable::FindPointer
293.787 us : (1000) leveldb::GLOBAL_N_1::LRUHandle::key
57.983 us : (1000) leveldb::Slice::Slice
1.312 ms : (1000) leveldb::operator!=
1.169 ms : (1000) leveldb::operator==
164.391 us : (3000) leveldb::Slice::size
109.200 us : (2000) leveldb::Slice::data
```

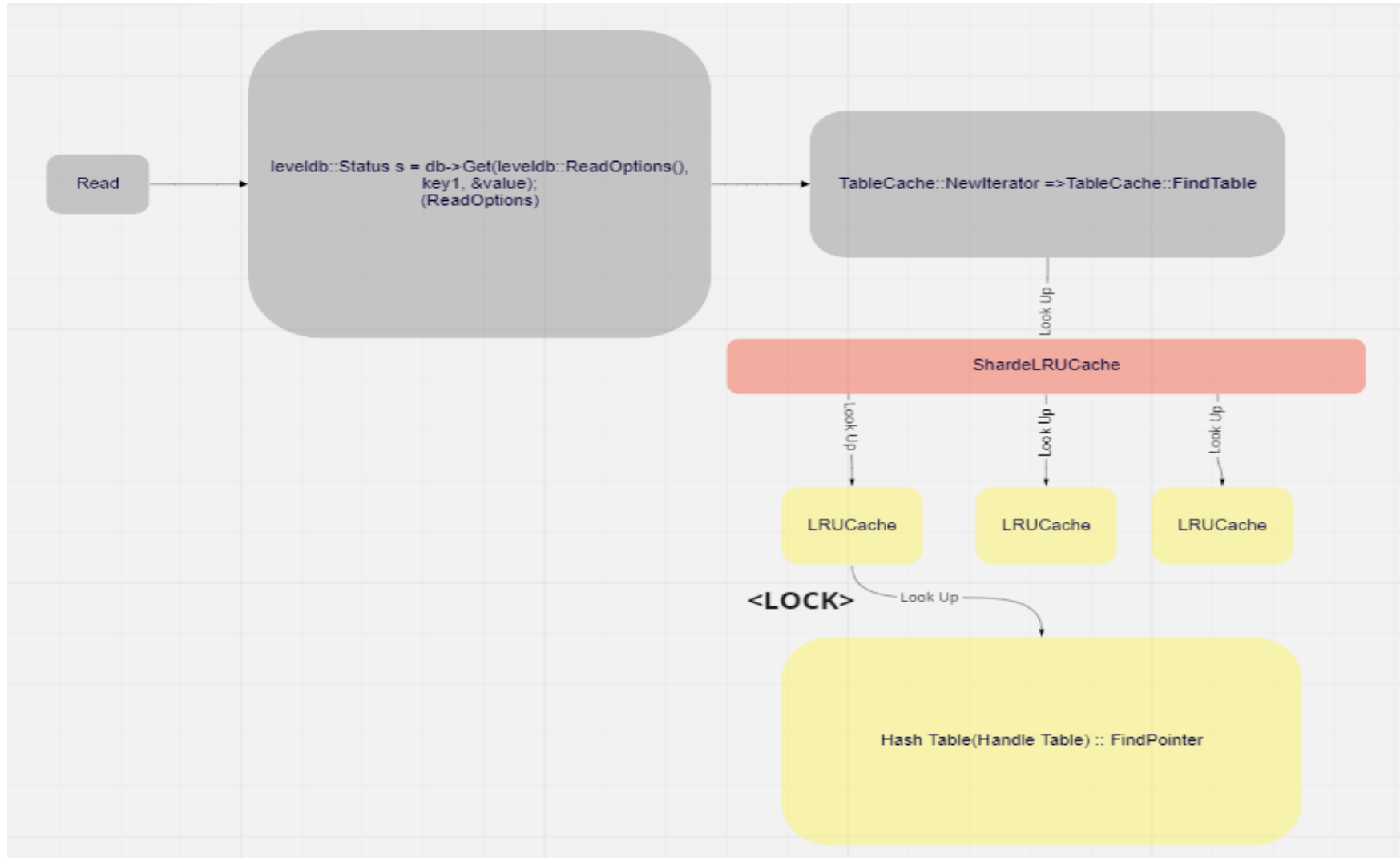
- Use ufrace tui to understand flow of the functions
- Look up the code to understand the function's function

# Overall cache flow chart



# Code flow

## - (1)Read flow



# Code flow

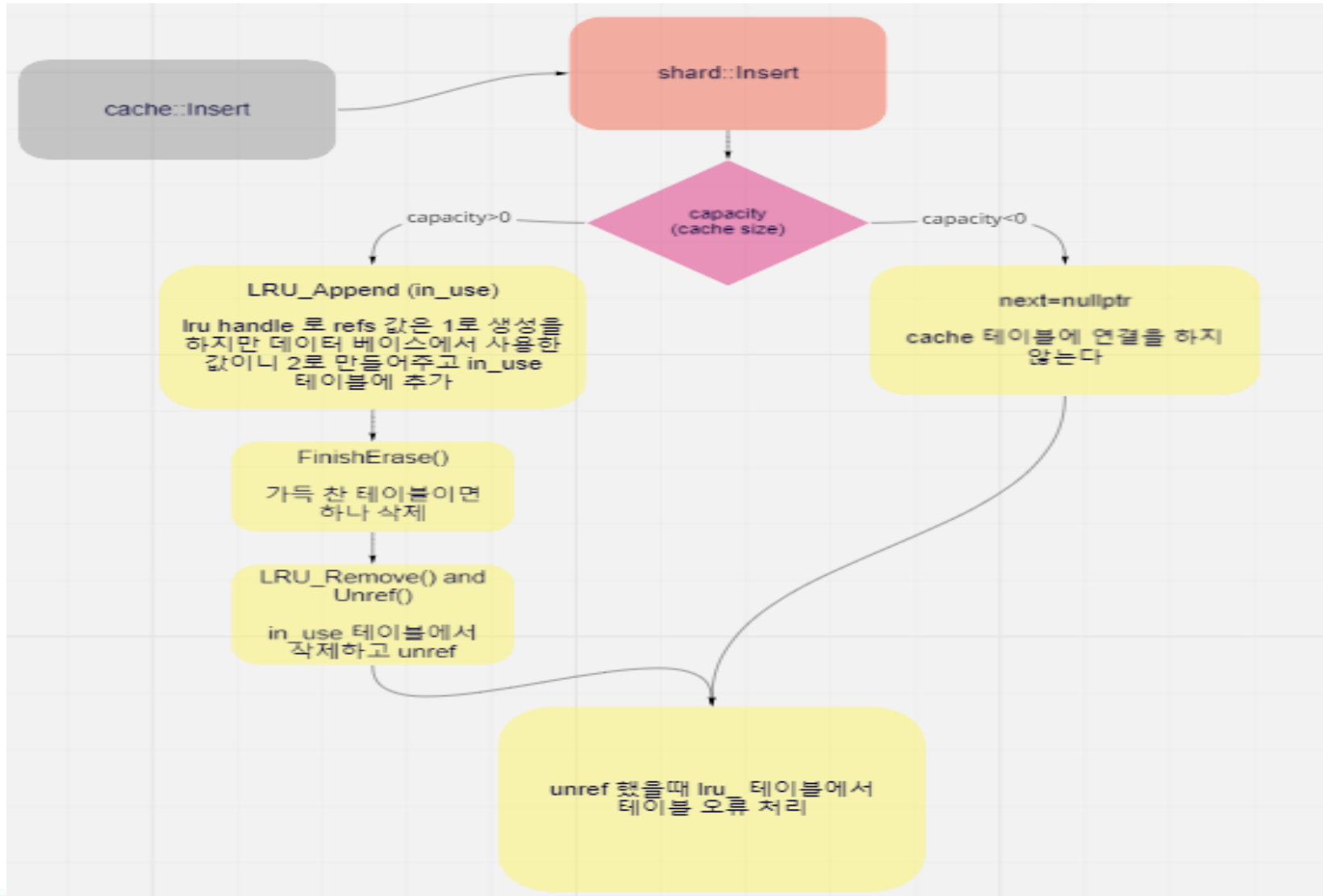
- (1)Read flow

-uftrace tui

6.214	ms	:	— (1000) leveldb:: GLOBAL N 1::ShardedLRUCache::Lookup
6.214	ms	:	(1000) leveldb::TableCache :FindTable
6.214	ms	:	(1000) leveldb::TableCache::Get
6.214	ms	:	(1000) leveldb::Version::Get::State::Match
6.214	ms	:	(1000) leveldb::Version::ForEachOverlapping
6.214	ms	:	(1000) leveldb::Version :Get
6.214	ms	:	(1000) leveldb::DBImpl::Get
6.214	ms	:	(1000) leveldb::Benchmark::ReadRandom

# Code flow

## - (2) Insert flow



# Code flow

## - (2) Insert flow

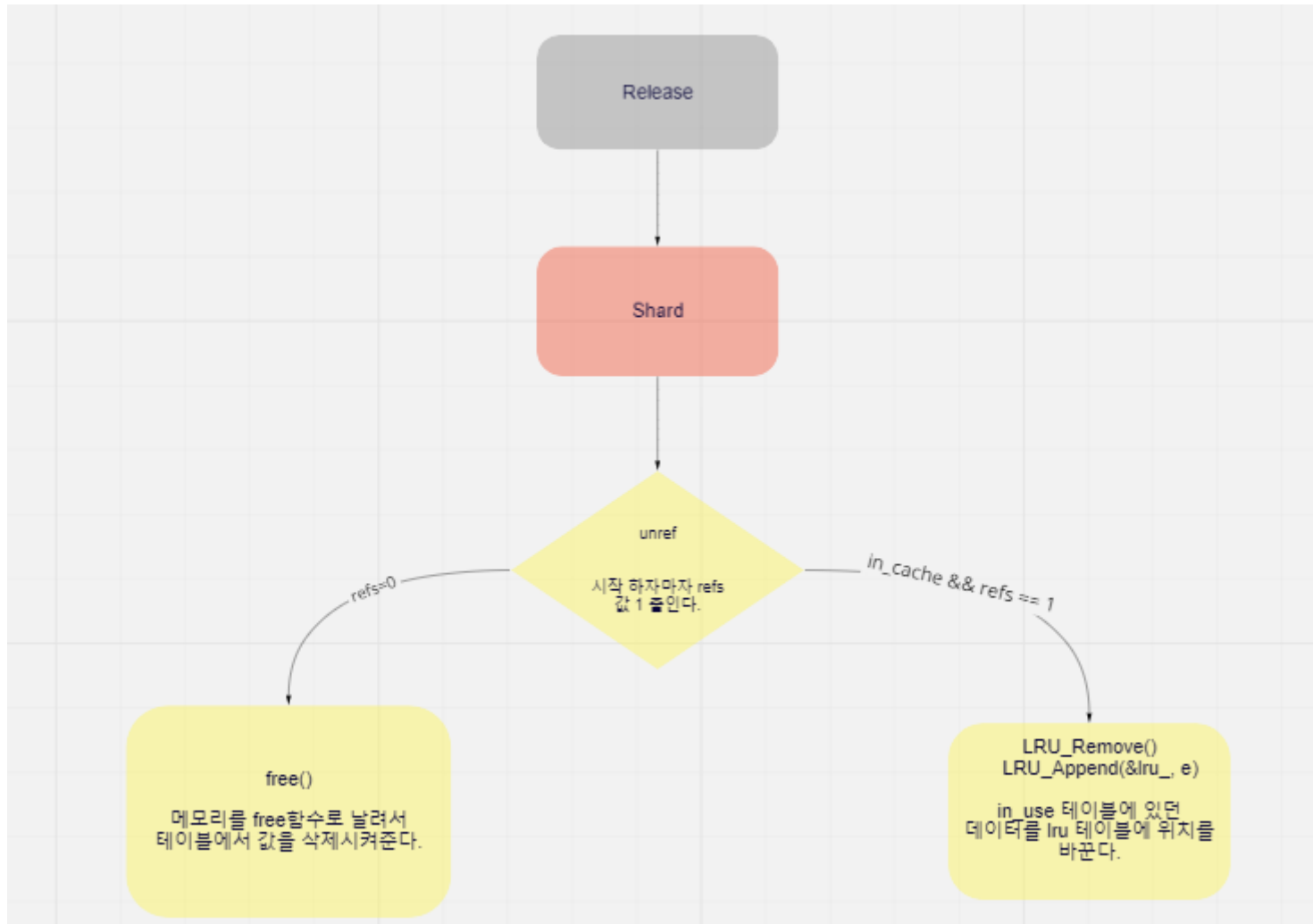
### -uftrace tui

```
===== Call Graph =====
7.496 us : (1) leveldb::_GLOBAL__N_1::ShardedLRUCache::Insert
0.839 us :   (1) leveldb::_GLOBAL__N_1::ShardedLRUCache::HashSlice
0.047 us :     (1) leveldb::Slice::size
:
0.046 us :     (1) leveldb::Slice::data
:
0.409 us :     (1) leveldb::Hash
0.117 us :       (2) leveldb::DecodeFixed32
:
0.047 us :   (1) leveldb::_GLOBAL__N_1::ShardedLRUCache::Shard
:
6.306 us :   (1) leveldb::_GLOBAL__N_1::LRUCache::Insert
0.064 us :     (1) leveldb::_GLOBAL__N_1::LRUCache::LRU_Append
:
0.588 us :     (1) leveldb::_GLOBAL__N_1::HandleTable::Insert
0.251 us :       (1) leveldb::_GLOBAL__N_1::LRUHandle::key
0.052 us :       (1) leveldb::Slice::Slice
:
0.086 us :     (1) leveldb::_GLOBAL__N_1::HandleTable::FindPointer
:
0.059 us :   (1) leveldb::_GLOBAL__N_1::LRUCache::FinishErase
```



# Code flow

## - (3) Release(call by=unref entry, Get()) flow



# Code flow

- (3) Release(call by=unref entry, Get()) flow

-uftrace tui

```
===== Call Graph =====
3.040 ms : (1001) leveldb::_GLOBAL__N_1::ShardedLRUCache::Release
52.702 us : |-(1001) leveldb::_GLOBAL__N_1::ShardedLRUCache: Shard
:
2.704 ms : |-(1001) leveldb::_GLOBAL__N_1::LRUCache::Release
979.511 us : |-(1001) leveldb::MutexLock::MutexLock
837.634 us : |-(1001) leveldb::port::Mutex::Lock
699.408 us : |-(1001) std::mutex::lock
557.612 us : |-(1001) __gthread_mutex_lock
53.313 us : |-(1001) __gthread_active_p
:
80.146 us : |-(1001) pthread_mutex_lock
:
438.387 us : |-(1001) leveldb::_GLOBAL__N_1::LRUCache::Unref
60.208 us : |-(1001) leveldb::_GLOBAL__N_1::LRUCache::LRU_Remove
:
59.521 us : |-(1001) leveldb::_GLOBAL__N_1::LRUCache::LRU_Append
:
945.972 us : |-(1001) leveldb::MutexLock::~~MutexLock
812.058 us : |-(1001) leveldb::port::Mutex::Unlock
675.718 us : |-(1001) std::mutex::unlock
533.670 us : |-(1001) __gthread_mutex_unlock
53.276 us : |-(1001) __gthread_active_p
:
78.413 us : |-(1001) pthread_mutex_unlock
```

# Q&A

---

