facebook

MyRocks: MySQL on RocksDB

Herman Lee
Facebook
Sept 2015

Agenda

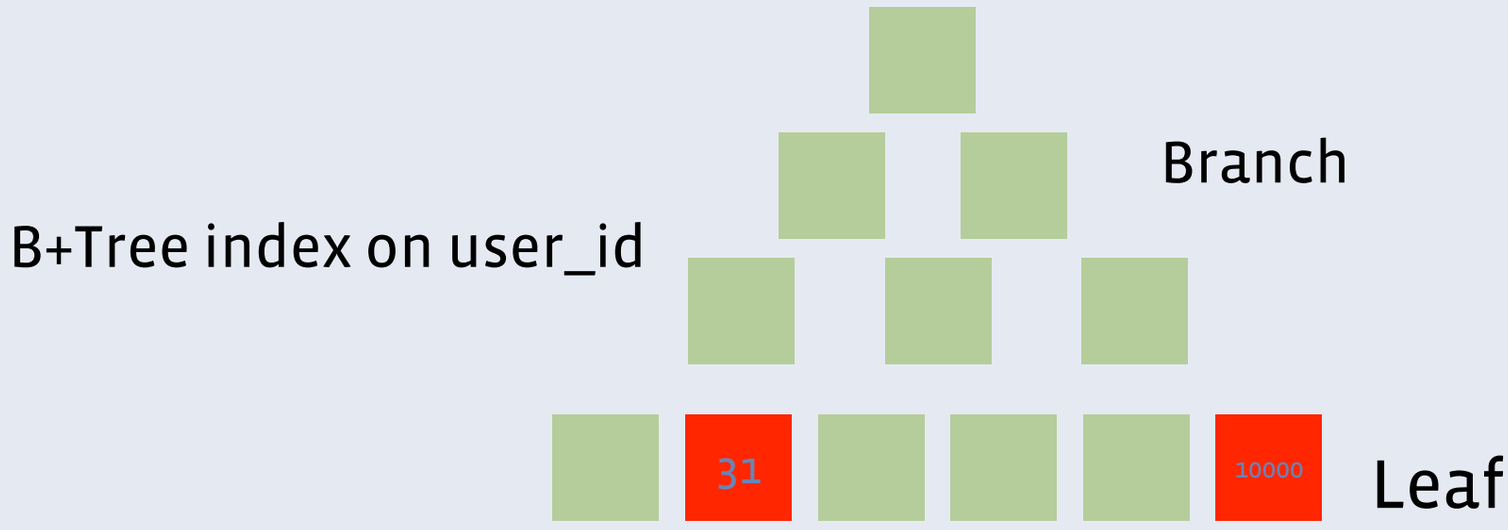
- B+Tree and InnoDB, and their issues on Disk/Flash
- What is a Log Structured Merge Tree; LSM pros and cons
- RocksDB and MyRocks
- Features
- Performance

H/W trends and limitations

- SSD/Flash is getting affordable, but is still expensive
- HDD: Large enough capacity but very limited IOPS
  - Reducing read/write IOPS is very important -- Reducing write is harder
- SSD/Flash: Great read iops but limited space and write endurance
  - Reducing space is higher priority

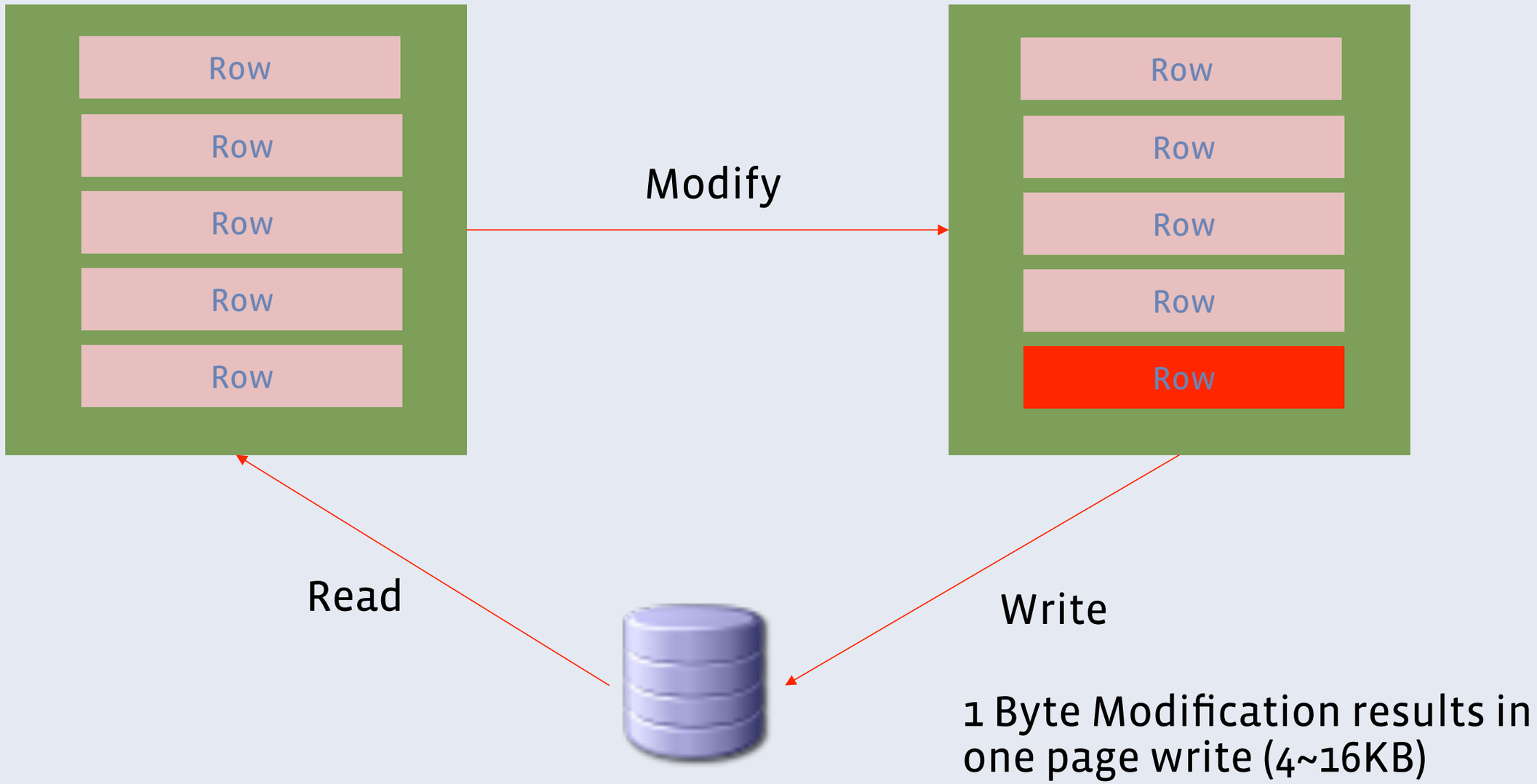
Random Write on B+Tree

```
INSERT INTO message (user_id) VALUES (31);
INSERT INTO message (user_id) VALUES (10000);
.....
```



- B+Tree index leaf page size is small (16KB in InnoDB)
- Modifications in random order => Random Writes
- N rows modification => In the worst case N different random page writes per index

Write Amplification on B+Tree



B+Tree Fragmentation increases Space

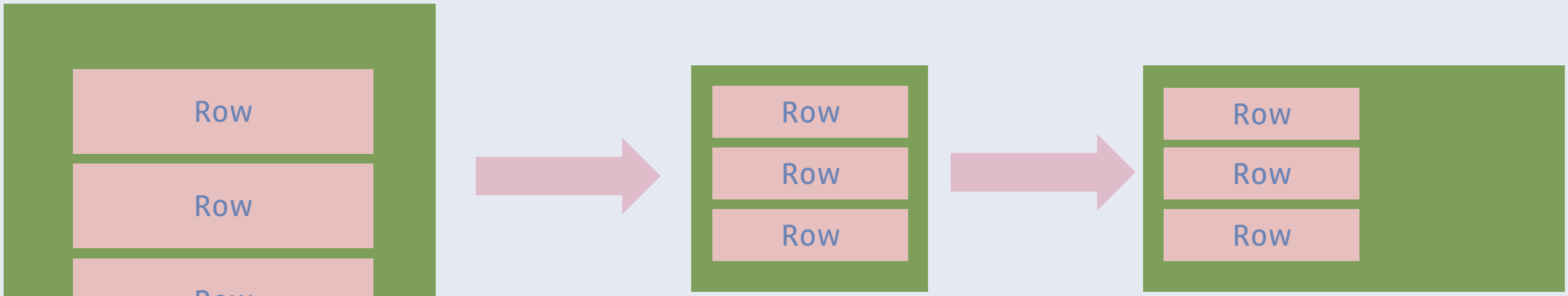
```
INSERT INTO message_table (user_id) VALUES (31)
```

Leaf Block 1	
user_id	RowID
1	10000
2	5
3	15321
…	
60	431

Leaf Block 1	
user_id	RowID
1	10000
…	
30	333
Empty	

Leaf Block 2	
user_id	RowID
31	345
…	
60	431
Empty	

Compression issues in InnoDB



Uncompressed
16KB page

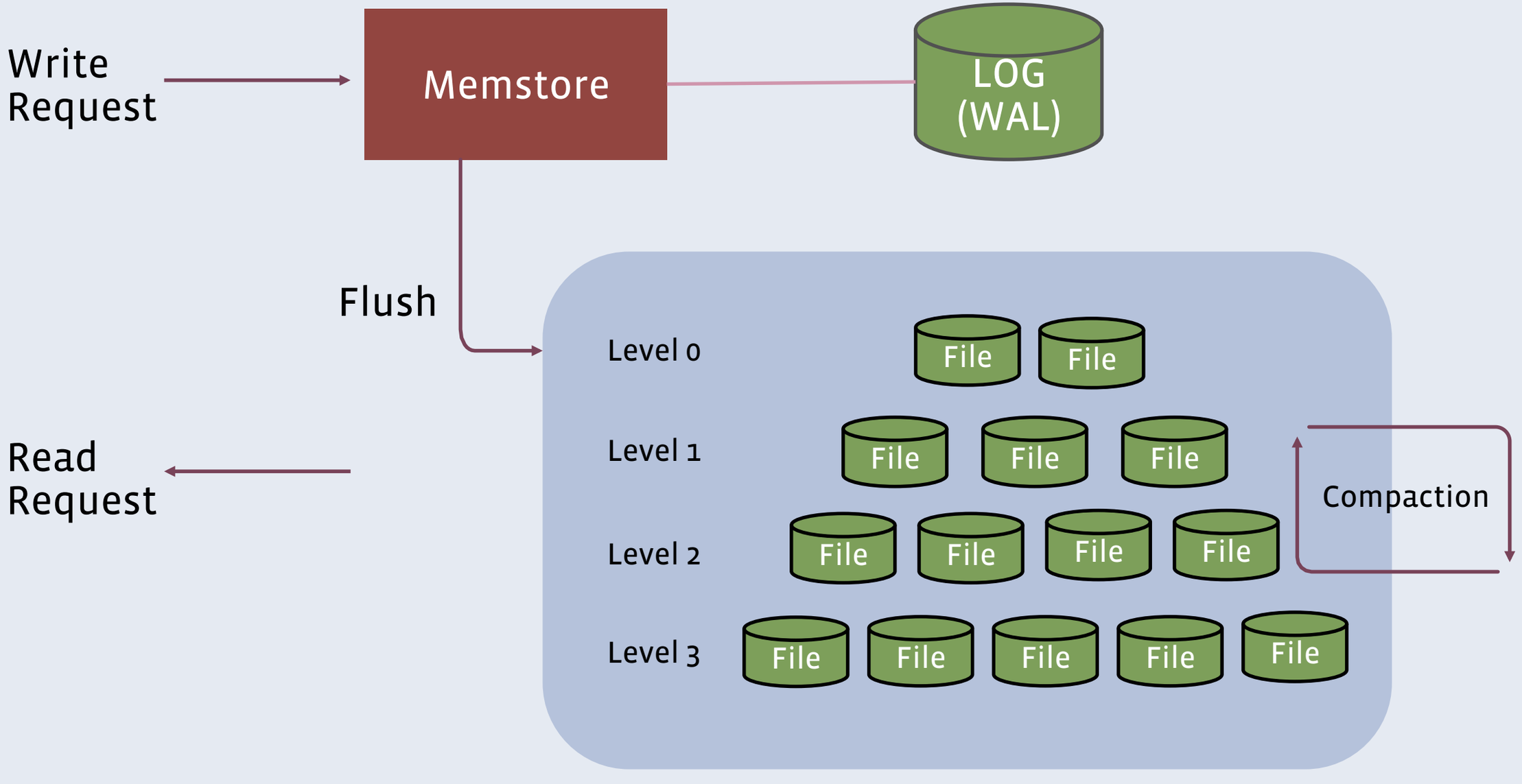
Compressed
to 5KB

Using 8KB space
on storage

0~4KB => 4KB
4~8KB => 8KB
8~16KB => 16KB

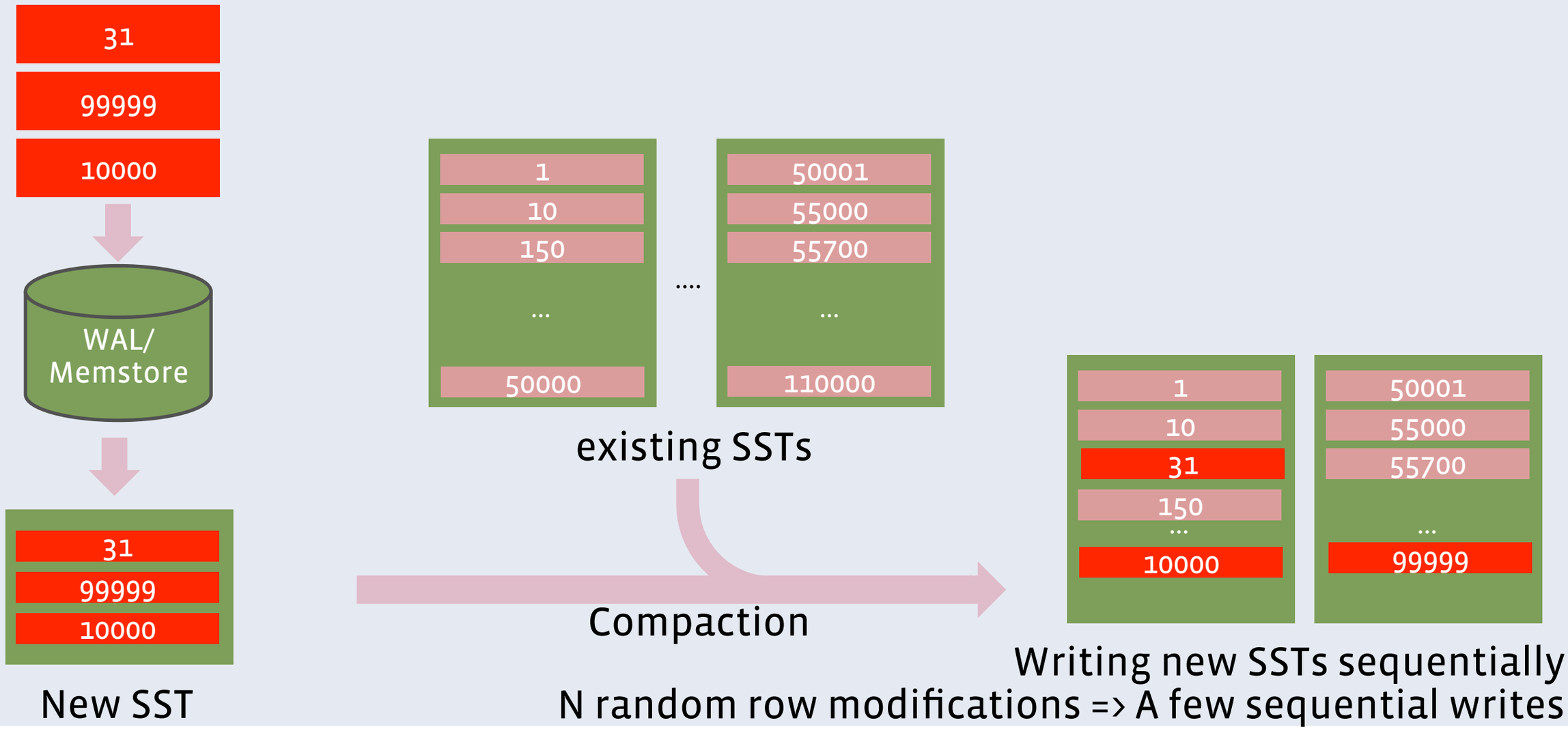
New (5.7~) punch-hole compression has similar issue

LSM Database

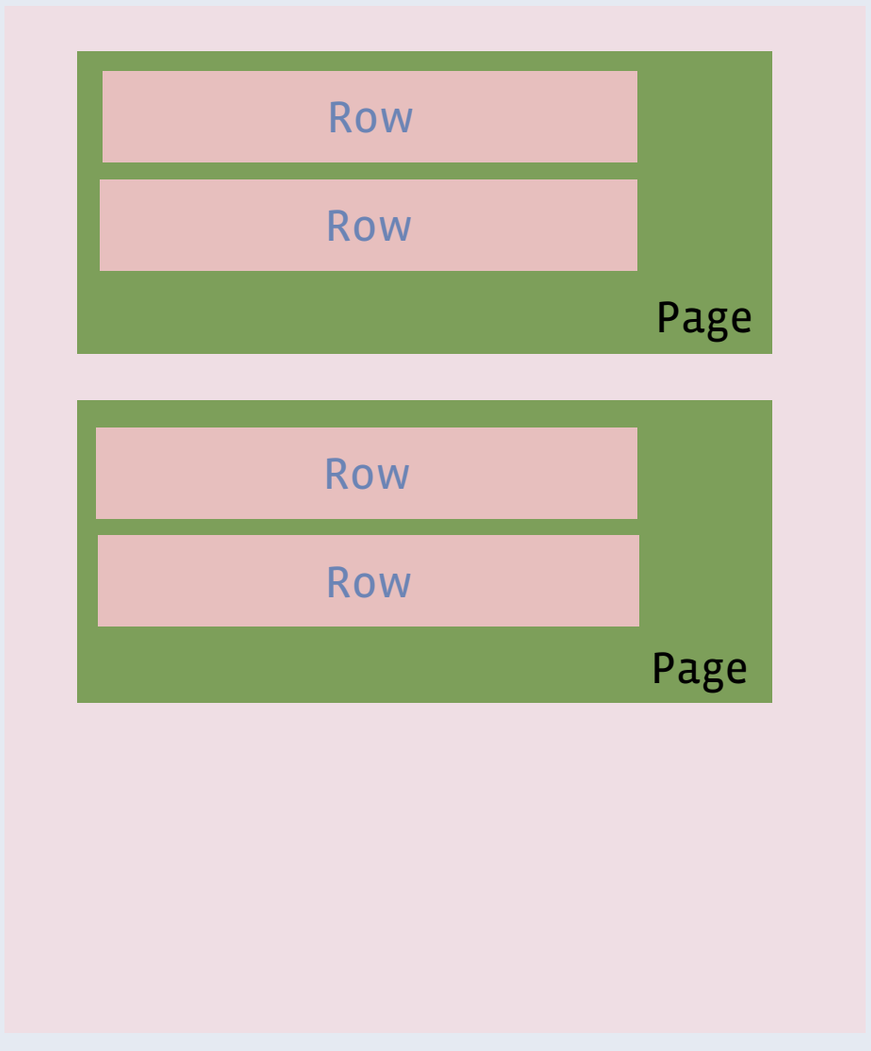


How LSM works

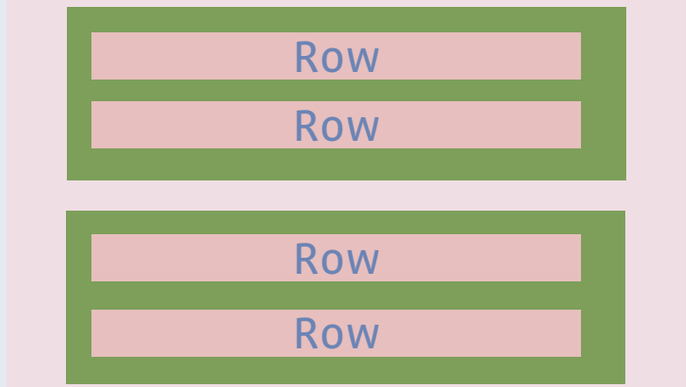
```
INSERT INTO message (user_id) VALUES (31);
INSERT INTO message (user_id) VALUES (99999);
INSERT INTO message (user_id) VALUES (10000);
```



LSM handles compression better



16MB SST File

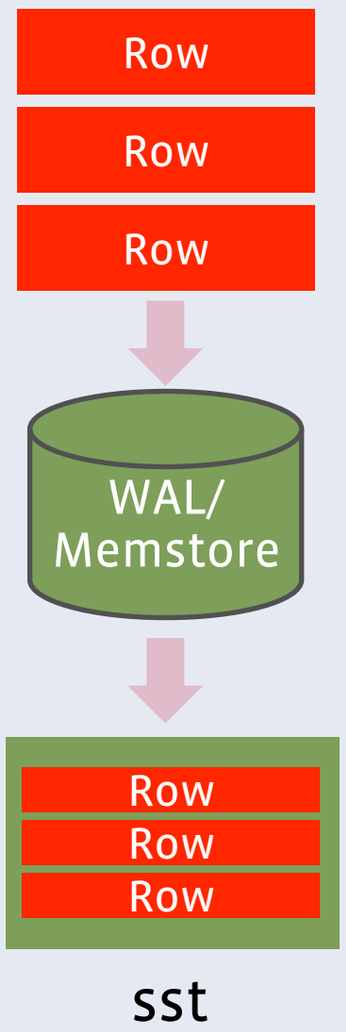


5MB SST File

=> Aligned to OS sector (4KB unit)
=> Negligible overhead

Reducing Space/Write Amplification

Append Only



Prefix Key Encoding

id1	id2	id3
100	200	1
100	200	2
100	200	3
100	200	4

id1	id2	id3
100	200	1
		2
		3
		4

Zero-Filling metadata

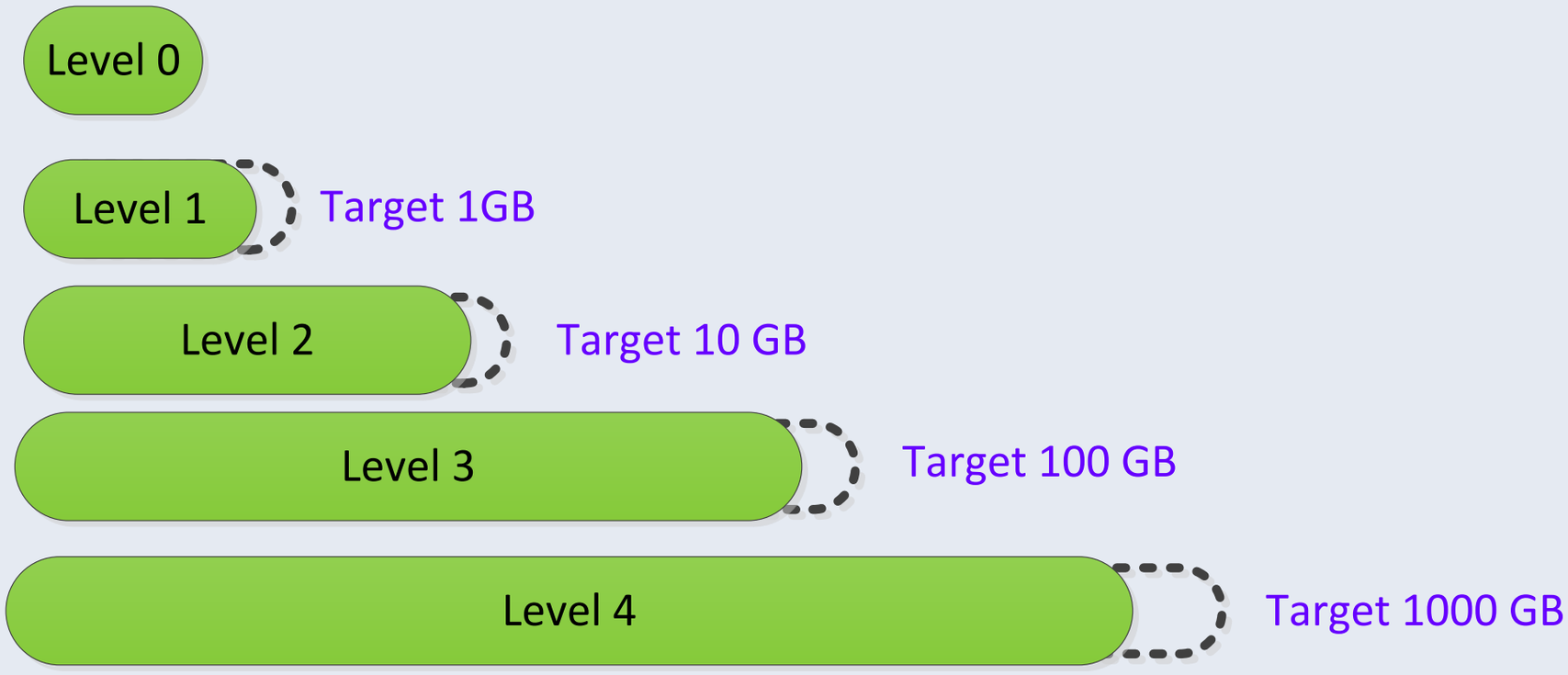
seq id	flag	key	value
1234561	W	k1	v1
1234562	W	k2	v2
1234563	W	k3	v3
1234564	W	k4	v4

seq id	flag	key	value
0	W	k1	v1
0	W	k2	v2
0	W	k3	V3
0	W	k4	V4

LSM Algorithm – Universal

- Read Amplification: number of files
- Write Amplification: 2~
- Space Amplification: depends

LSM Compaction Algorithm -- Level

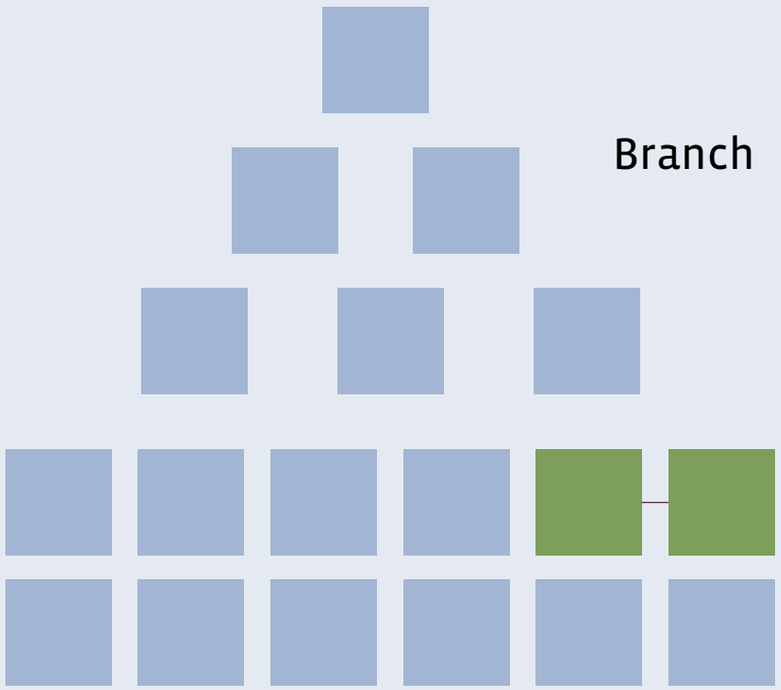


- Read Amplification: 1 ~ number of levels
- Write Amplification: 1 + 1 + fanout * (number of levels – 2)
- Space Amplification: 1.11

Read Penalty on LSM

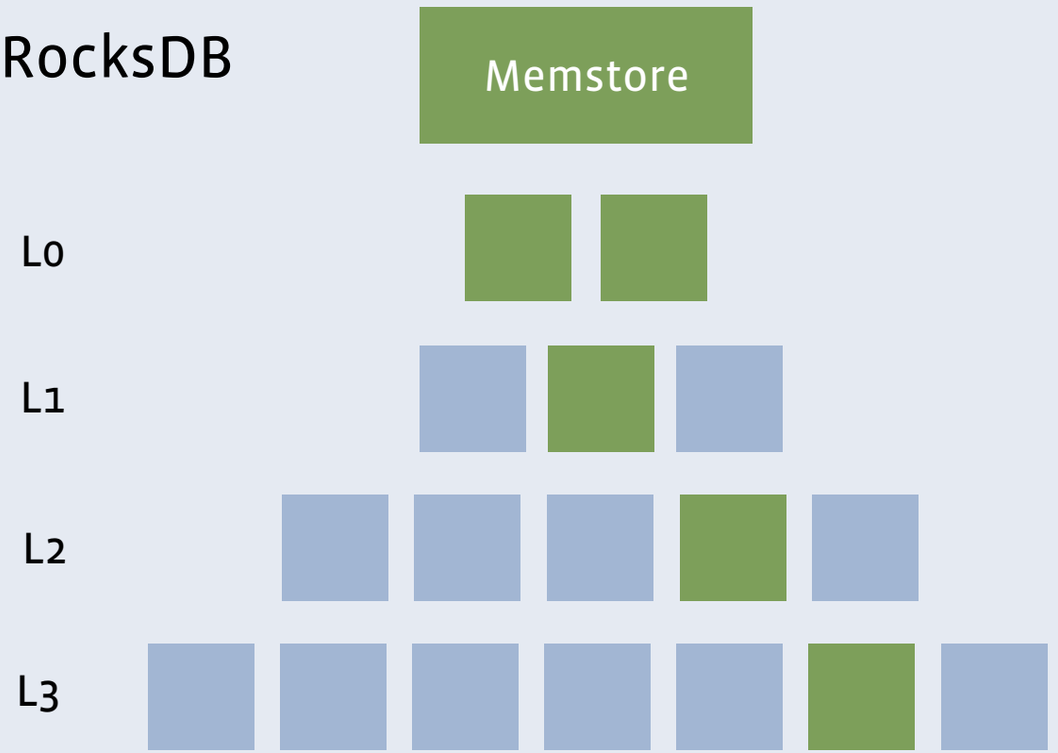
SELECT id1, id2, time FROM t WHERE id1=100 AND id2=100 ORDER BY time DESC LIMIT 1000;
Index on (id1, id2, time)

InnoDB



Range Scan with covering index is done by just reading leaves sequentially, and ordering is guaranteed (very efficient)

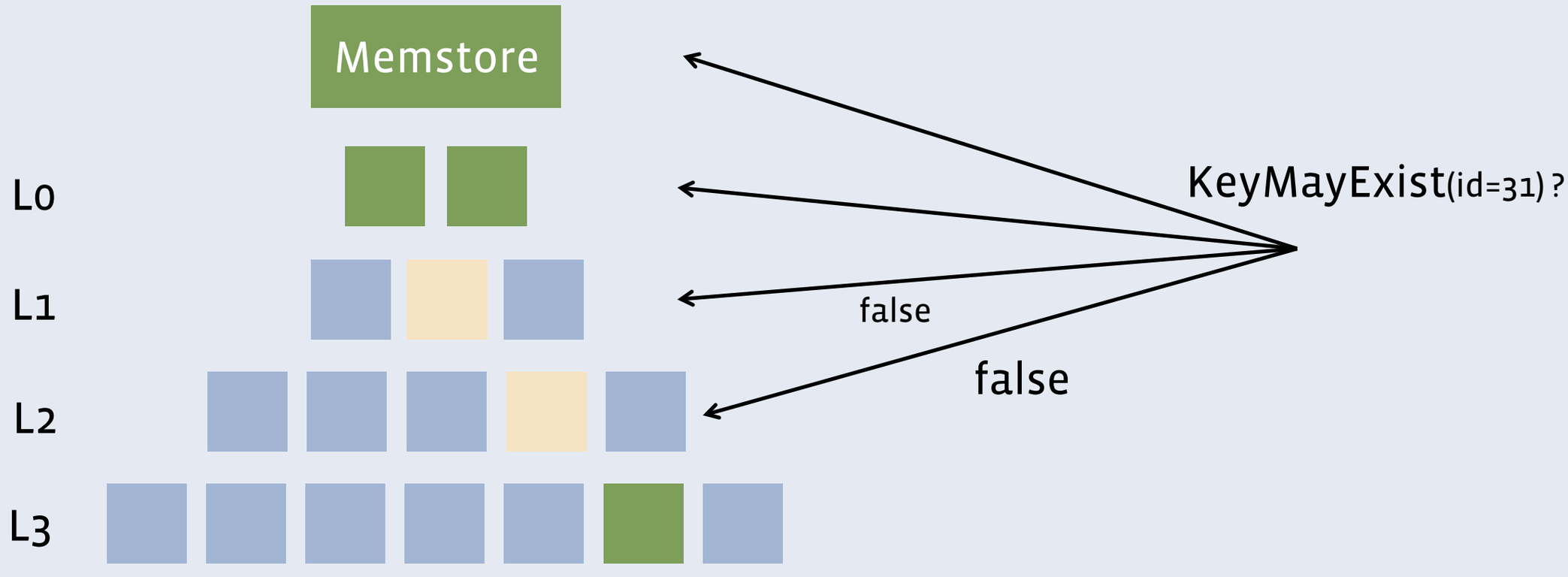
RocksDB



Merge is needed to do range scan with ORDER BY

Bloom Filter

Checking key may exist or not, and skipping read i/o if it definitely does not exist



RocksDB

- Forked from LevelDB
  - Key-Value persistent store
  - Embedded
  - Data stored locally
  - Optimized for fast storage

- LevelDB was created by Google
- Facebook forked and developed RocksDB

MyRocks (RocksDB storage engine for MySQL)

- Taking both LSM advantages and MySQL features
  - LSM advantage: Smaller space and lower write amplification
  - MySQL features: SQL, Replication, Connectors and many tools
- MariaDB and Facebook are collaborating together on MyRocks
  - Open Source
  - Currently alpha stage
  - https://github.com/facebook/mysql-5.6/

Interesting Features & Issues

- Reverse column families
- Online copy using snapshots
- Delete Tombstones
- Tracking stats for optimizer

Reverse column families

- RocksDB is great at scanning forward
- But ORDER BY DESC queries are slow
- Reverse column families added

id1	id2	id3
200	200	4
		3
		2
		1

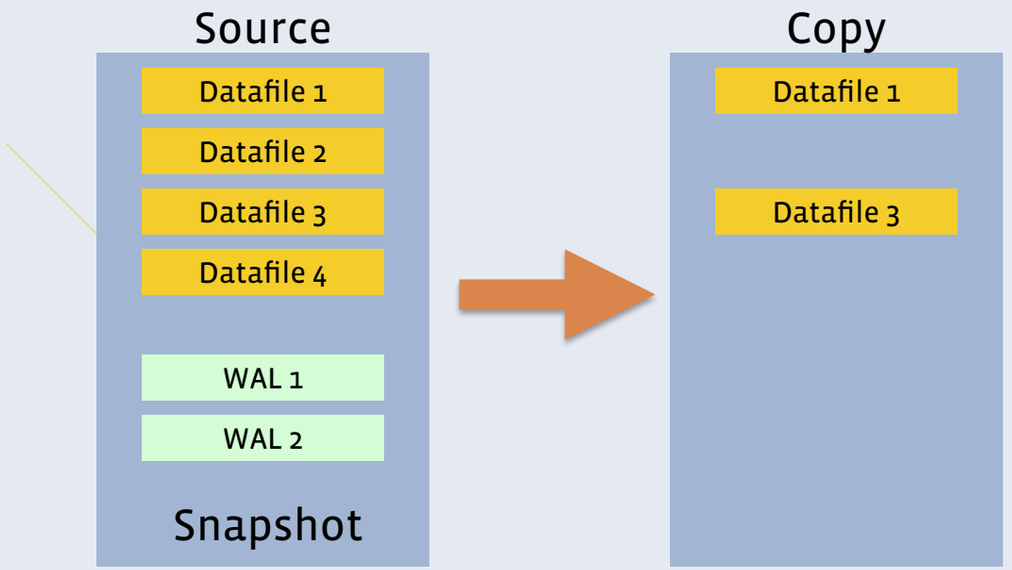
Prefix Key Encoding

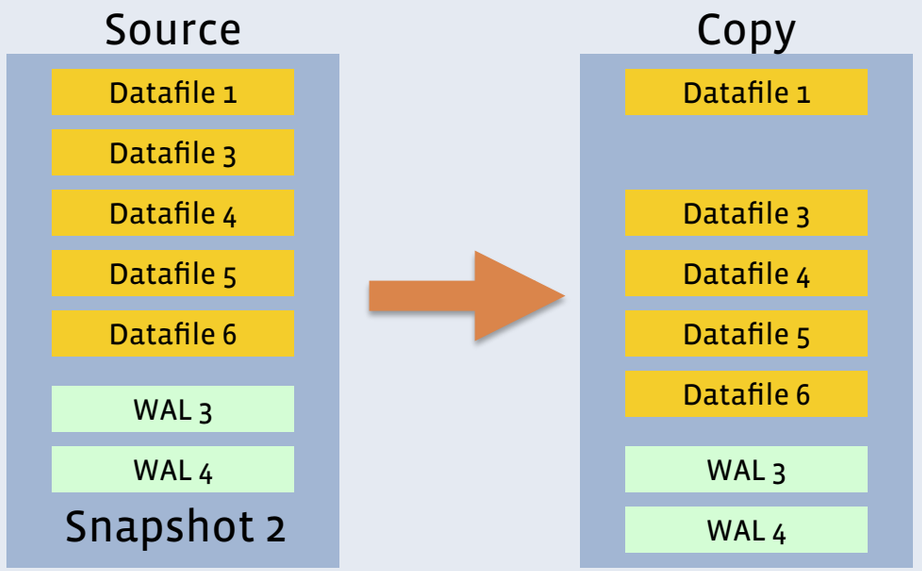
id1	id2	id3
100	200	1
100	200	2
100	200	3
100	200	4

id1	id2	id3
100	200	1
		2
		3
		4

Online Copy Utility

- Clones a MyRocks instance while the server is active
- WAL/data files are immutable
- Create snapshot using filesystem link call
- Copy files, start server
- Optimization: snapshot deltas





Delete Tombstones

- Delete markers needed
- Slows scans by 2.5x
- More frequent compactions increases write-amp

Optimizer Stats

- Query plans need table statistics
- Track statistics per data file
- Performed during compaction and memtable flush
- Updated periodically

Performance

- LinkBench
- Space Usage
- Write Rates
- Read Rates
- CPU utilization
- Query Latencies
- SysBench

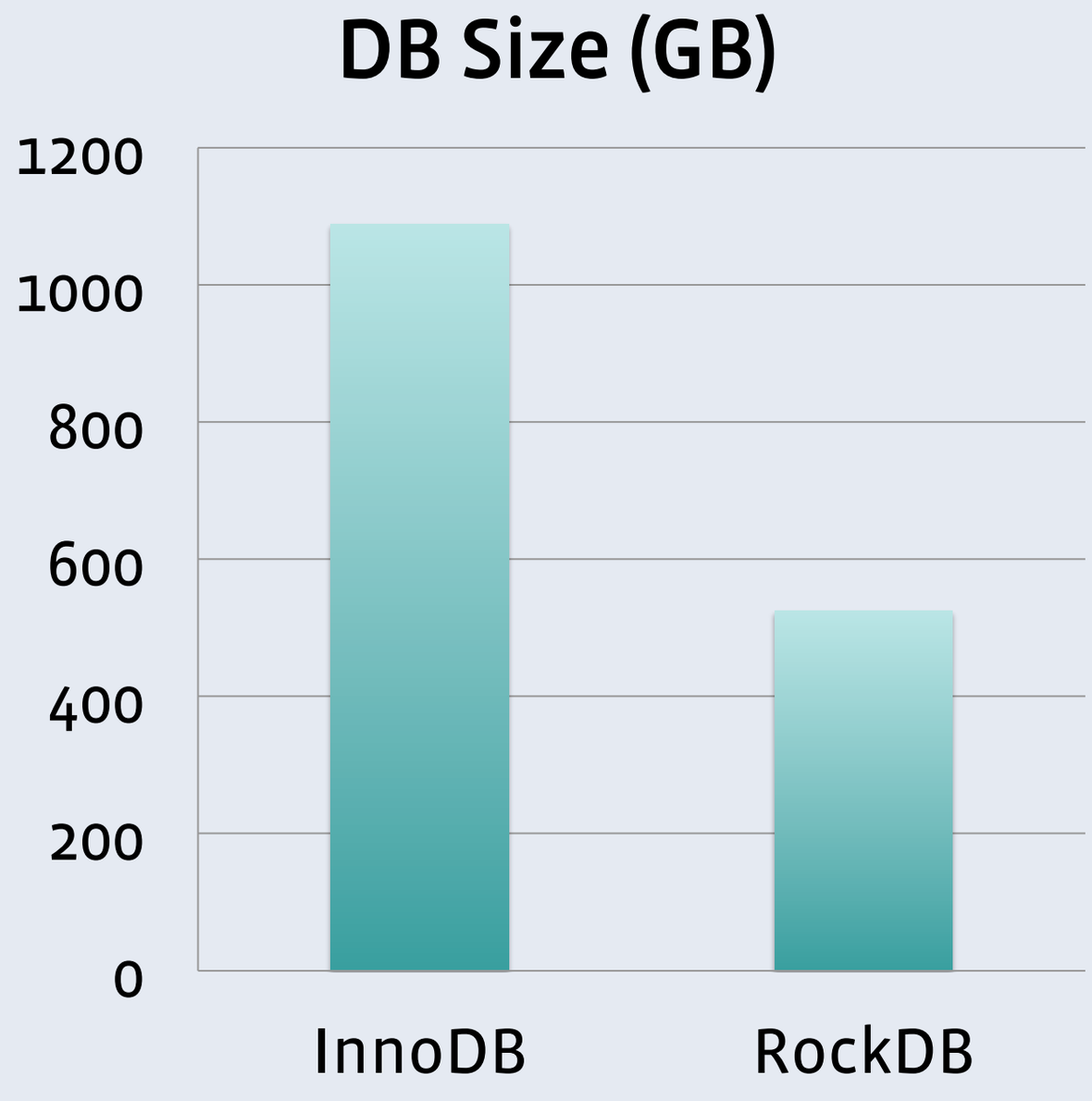
Linkbench

- 1.5B IDs, 32 query threads, 48 hour run, flash storage

	InnoDB	RocksDB
Space	1172GB	574GB (49%)
QPS	20227	33094
Read (kB/s)	249001	155236 (62%)
Write (kB/s)	152422	66931 (44%)
CPU utilization	50%	85%

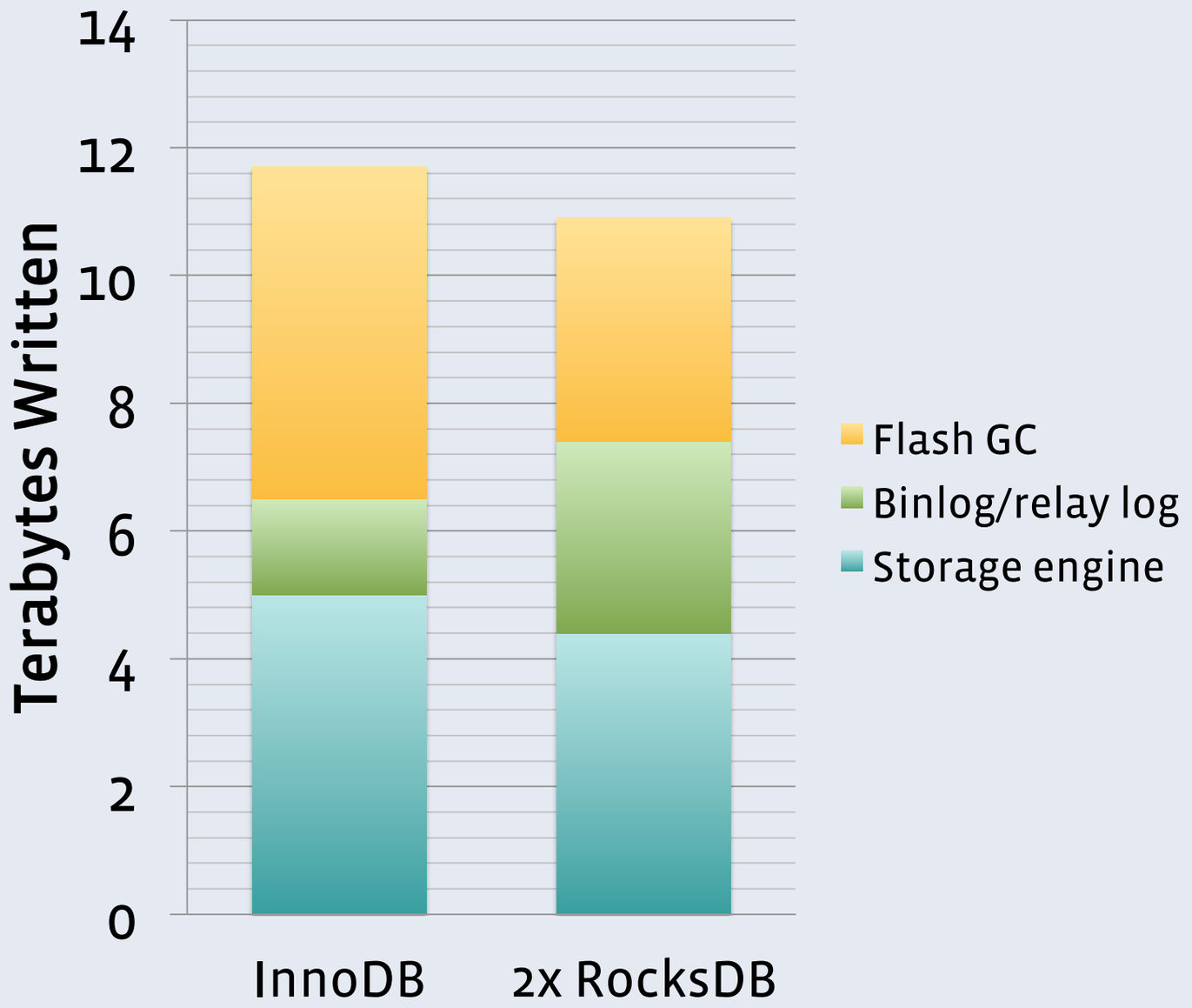
Space usage (production setup)

- Uses zlib compression
- Experimenting with higher instances/server ratios



Write rates

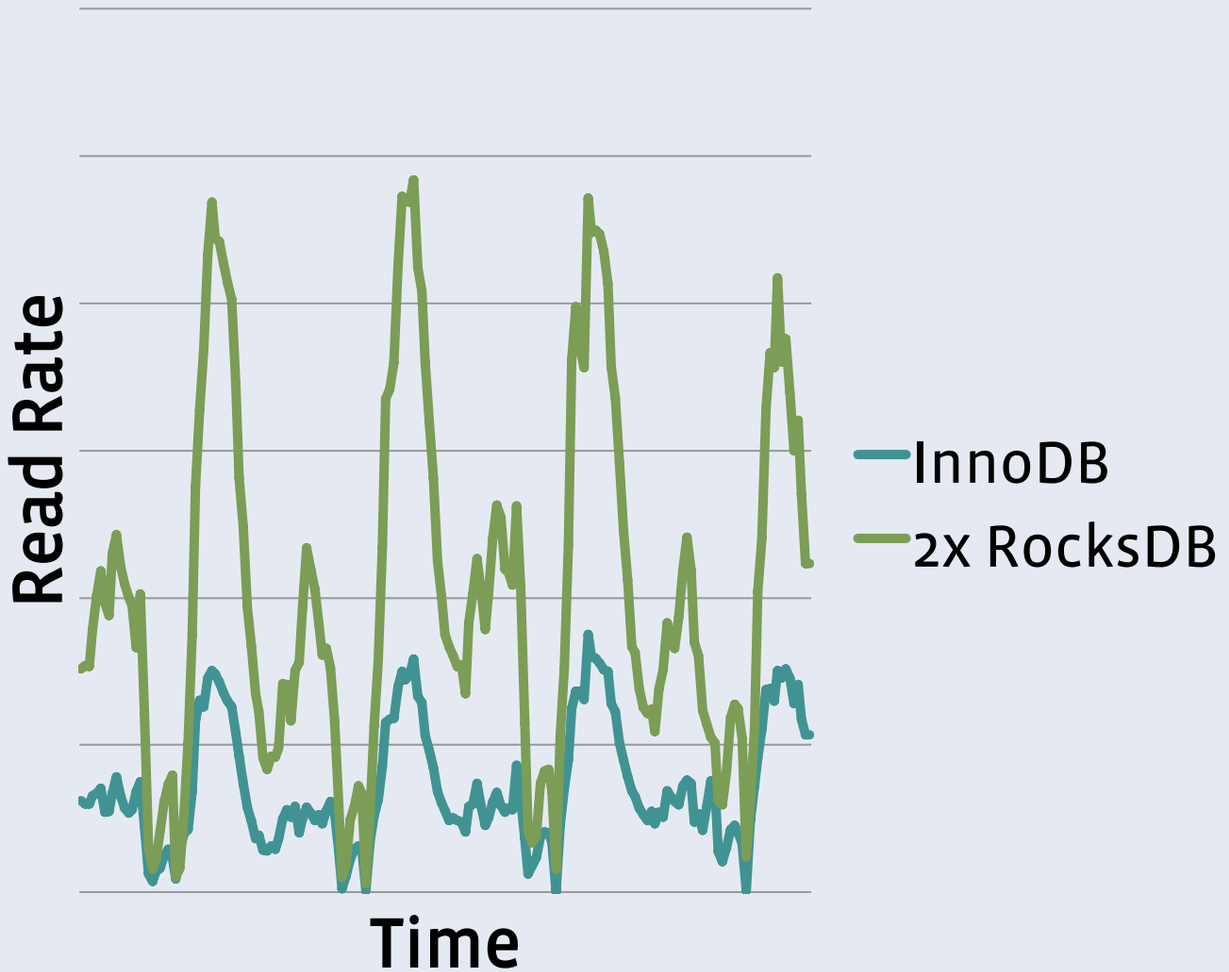
- Important for extending flash duration
- Double instances (2x) on RocksDB, same read/write traffic per instance



Read rates

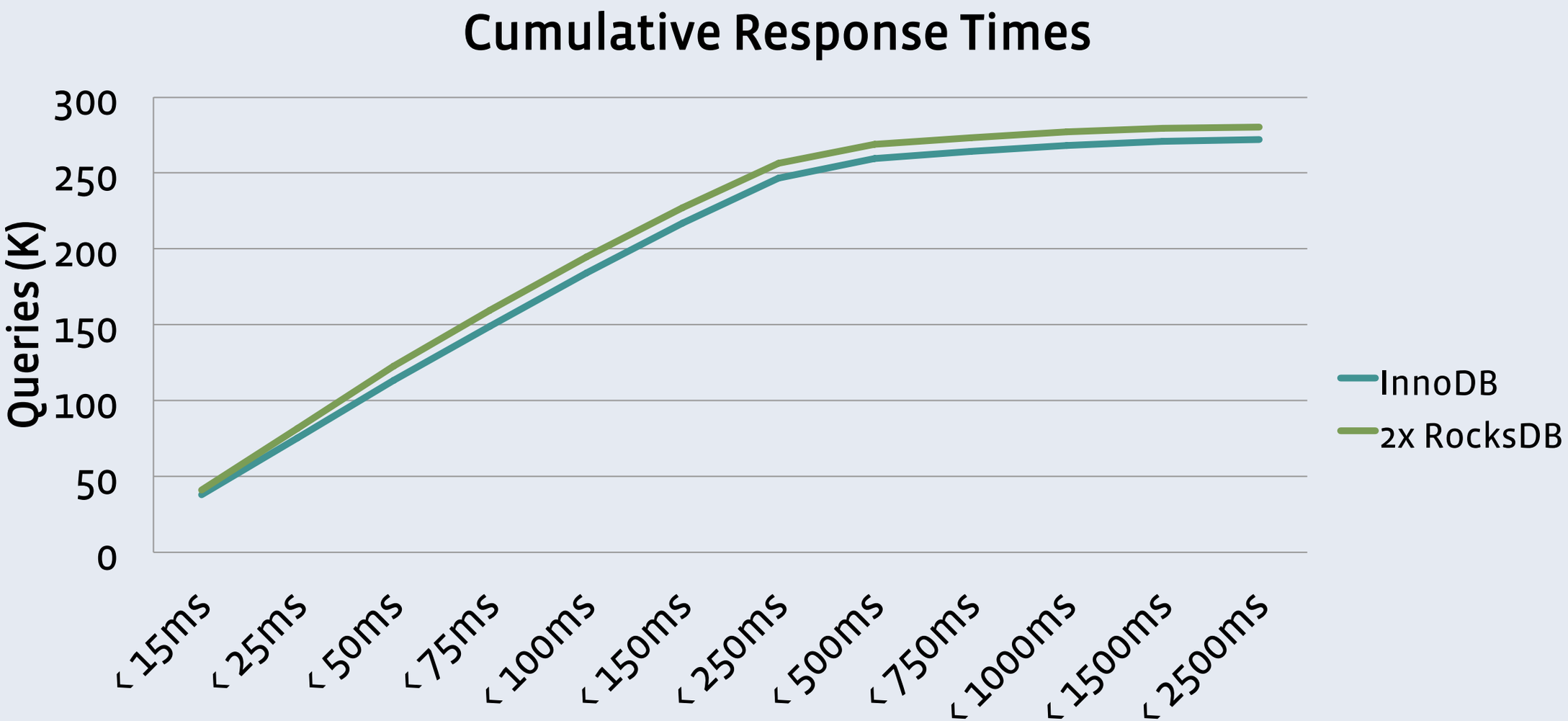
- 2x RocksDB read 300% more than InnoDB

	InnoDB	2xRocksDB
Flash bytes read	12TB	37TB (~300%)



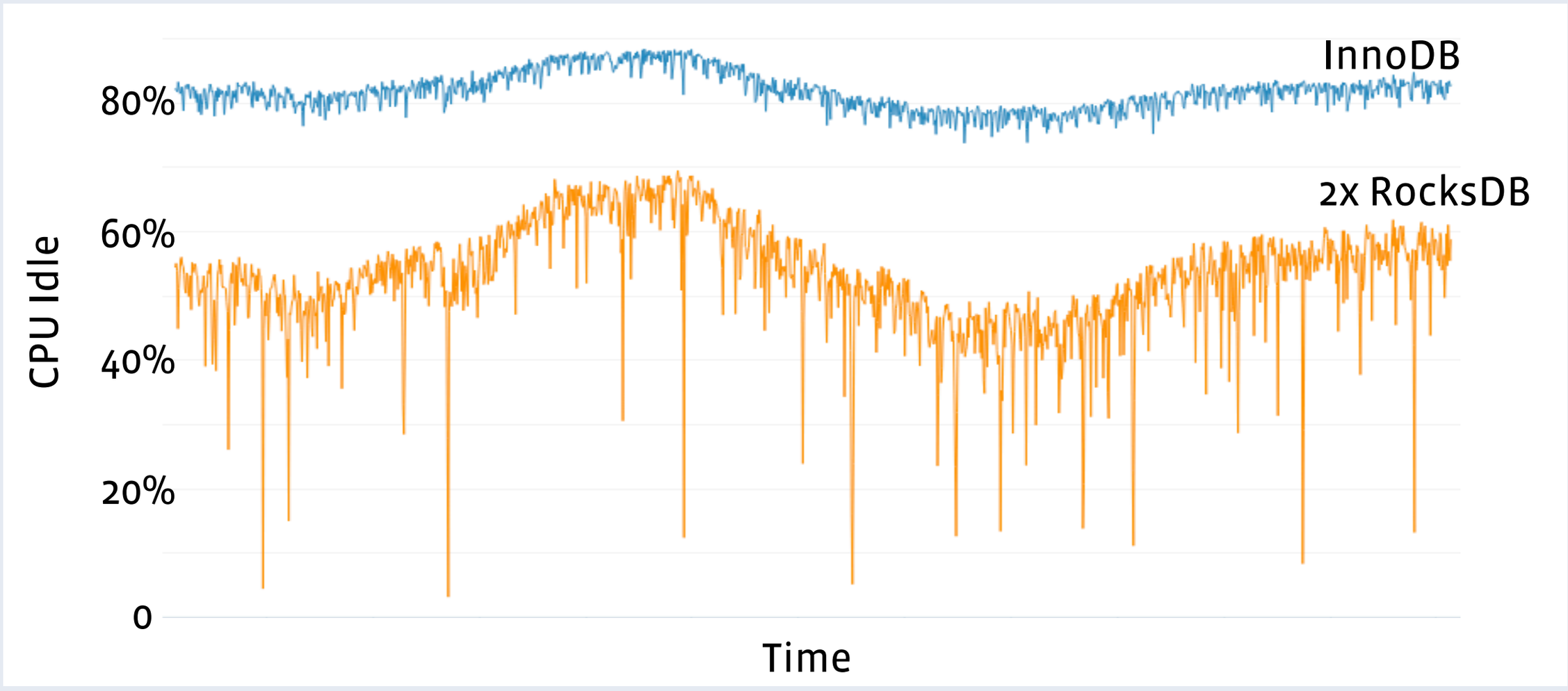
Latency

- Slow query log to capture query times



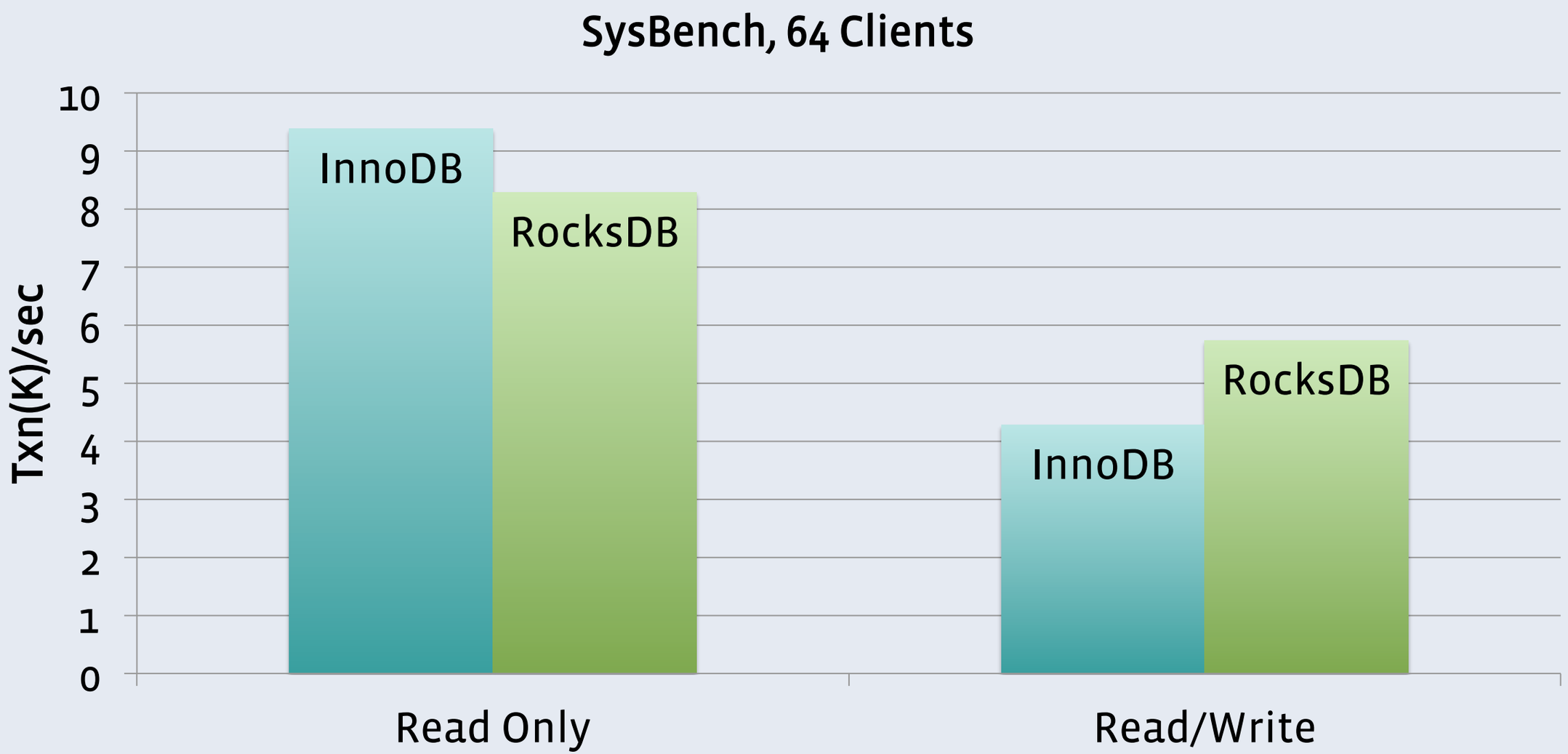
CPU

- InnoDB around 80% cpu idle.
- 2xRocksDB around 50% to 60%



SysBench

- 1.4B rows, uncompressed data (InnoDB: 318GB, RocksDB: 287GB)



MyRocks features

- Much lower write and space amplification
- Clustered Index
- Mem comparable keys (SELECT can be even faster with *_bin collation)
- Crash Safe slave/master
- Online backup
- Reverse-order index so that ORDER BY DESC can be faster
- Bloom filter to make lookups faster

Current MyRocks limitations

- No support for Online DDL, Foreign Key, Spatial Index, and Fulltext Index yet
- All tables must have Primary Key
- No next key locking support, ROW based binary logging must be used
- Full durability with XA (binlog and MyRocks) is not supported
- Either ORDER BY DESC or ASC is slow
- Bulk deletes slow Select/Update/Delete

Summary

- There are trade-offs across Space, Write and Read amplification
- LSM reduces Space and Write, but increases Read
- Don’t expect LSM to replace B+Tree
- We started the MyRocks project to take advantage of LSM properties

facebook