



## Tech Note: SchoonerSQL Optimizes InnoDB for Top Performance

InnoDB is the standard MySQL transactional storage engine. While very widely used, InnoDB was not designed to take full advantage of the power of modern commodity hardware. Many MySQL installations see inefficient use of their servers because they're "out of balance": they hit a bottleneck on at least one hardware resource (disk I/O, DRAM, CPU cycles, or network capacity).

SchoonerSQL™ eliminates the chokepoints in traditional MySQL and InnoDB, while retaining 100% compatibility. SchoonerSQL is tuned for extensive multi-threading to exploit the parallelism inherent in commodity servers using multi-core x86 processors. When used with flash memory, SchoonerSQL goes even further to efficiently balance the use of cores, flash, and DRAM to minimize the chance that any one hardware resource will limit throughput.

### InnoDB Optimizations in SchoonerSQL

To get the most from InnoDB, SchoonerSQL optimizes it in several ways that include:

- Exploiting multi-core concurrency
- Re-architecting the buffer pool
- Adding scan-resistance
- Optimizing the I/O system to exploit flash memory
- Fine-tuning performance through specialized InnoDB formats

A brief technical summary of each of these optimizations follows.

**Exploiting Multi-Core Concurrency for Higher Performance:** Multi-core x86 processors support parallel execution of multiple instruction streams (threads). Multiple cores can make some applications run much faster than they would by simply increasing the CPU clock speed. However most software has not been written with the concurrency and parallelism needed to take advantage of multiple cores.

This is certainly true with MySQL and InnoDB: they were written with coarse locking which serializes access to resources which are not inter-dependent. They also serialize access to very large data structures which could otherwise be accessed in parallel in independent regions. The result is that threads queue up and block under heavy workloads, leading to low utilization of multi-core processors and a plateau in transaction throughput.

SchoonerSQL fixes these problems with tuning that includes:

- Splitting unrelated activities under one lock into separate locks;
- Converting serial data structures to be concurrent; and
- Reducing the granularity of locks to exploit the faster response time of SSDs compared to HDDs.

**Re-Architecting the Buffer Pool for Higher Throughput:** The database buffer pool is at the heart of database performance for online transaction processing (OLTP). The buffer pool's locality-of-reference, efficiency, and parallelism have a huge impact on transaction throughput. InnoDB traditionally uses a

buffer pool organized as a Least Recently Used (LRU) list. In practical use, strict LRU ordering of 2 million blocks (in a typical 32GB InnoDB buffer pool with 16kB block size) does not have big effect on the hit rate of the buffer pool.

SchoonerSQL optimizes InnoDB to use a buffer pool organized as a Clock, an algorithm that approximates LRU behavior but with less overhead. Schooner also tuned InnoDB to employ very intelligent memory management between DRAM and flash memories, effectively making flash memory an extension of DRAM. To maintain parallelism in SSD access and to eliminate replacement serialization, SchoonerSQL ensures that DRAM-based database buffers are flushed in the background and space allocation occurs concurrently rather than using a serial LRU approach. In addition, Schooner's patent-pending techniques greatly accelerate logging of writes and commits. These memory and log management techniques maximize the concurrency of DRAM / flash access and minimize the time a transaction thread waits to read data or hold a lock, dramatically speeding up index access and transaction throughput.

**Adding Scan Resistance to Improve Response Time:** Most workloads use indexes efficiently, and queries are written to leverage the index attribute. In the case where the attribute is not indexed (primary or secondary), the query triggers a scan which results in a time-consuming sequential read of all the data stored in a table. In traditional MySQL with InnoDB, the sequential read of large amounts of data that are not later reused tends to replace more frequently-used data in the buffer pool. This dramatically degrades the response time and throughput of indexed transactions. SchoonerSQL is tuned to minimize the effects of scans by detecting and limiting their impact on buffer pool usage.

**Optimizing the I/O System to Exploit Flash Memory:** Hard disk drives (HDDs) are characterized by high read or write IOPS (I/O per second) for sequential access and modest IOPS for random access, due to mechanical disk arm movement. Since solid state drives have no moving parts they're incredibly fast compared to HDDs and have about the same sequential and random IOPS. Thus many MySQL users are moving to SSDs to speed up access and save on power consumption.

But writes to flash memory are slower than reads, and flash drives undergo predictable wear based on the volume and pattern of writes. Using flash instead of HDDs reduces the random access I/O time by up to 100X. But InnoDB, initially written with HDD access times in mind, doesn't take advantage of these much faster flash access times. For example, a lock may be set for 1/1,000 of a second when it could be set for 1/100,000 of a second. To fully exploit flash, InnoDB needs algorithms, data-structures and I/O mechanisms tuned for flash. SchoonerSQL is specifically tuned for maximum parallelism in read, write, purge, flush, checkpoint and foreground threads to exploit the increased I/O capacity of flash memory.

**Fine-Tuning Performance through Specialized InnoDB Formats:** SchoonerSQL supports further performance improvements with flexible block sizes and enhanced checksum algorithms. These options reduce CPU overhead and further improve memory management.

## Discover What SchoonerSQL Can Do for Your Datacenter

SchoonerSQL takes MySQL and InnoDB where it's never been before. We package deep analysis and innovation in SchoonerSQL so you don't have to figure it out on your own. See for yourself with a download and free trial at [http://www.schoonerinfotech.com/free\\_trial](http://www.schoonerinfotech.com/free_trial).