WHITE PAPER

Accelerating Big Data: Using SanDisk® SSDs for Apache HBase Workloads

December 2014

SanDisk
a Western Digital brand

Western Digital Technologies, Inc.
951 SanDisk Drive, Milpitas, CA 95035

www.SanDisk.com

Table of Contents

Executive Summary

In today's hyper-connected world, there is a significant amount of data being collected, and later analyzed, to make business decisions. This explosion of data has led to various analytics technologies that can operate on this "Big Data", which stretches into the petabyte (PB) range within organizations, and into the zettabyte (ZB) range worldwide.

Traditional database systems and data warehouses are being augmented with newer scale-out technologies like Apache Hadoop and with NoSQL databases like Apache HBase, Cassandra and MongoDB to manage the massive scale of data being processed today, and analyzed by the organizations that use these databases .

This technical paper describes the benchmark testing conducted by SanDisk, using SanDisk SSDs in conjunction with an Apache HBase deployment. The testing uses the "Yahoo! Cloud Serving Benchmark" (YCSB), and 64GB and 256GB data sets running on a single-node Apache HBase deployment . This testing had a focus on update-heavy and read-heavy YCSB workloads. The primary goal of this paper is to show the benefits of using SanDisk solid-state drives (SSDs) within an HBase environment .

As the testing demonstrated, SanDisk SSDs provided significant performance improvements when running I/O-intensive workloads, especially with a mixed workload having random read and write data accesses over mechanically driven hard disk drives (HDDs). For more information, see www.sandisk.com.

SanDisk SSDs help boost the performance in HBase environments by providing 30x to 40x more transactions/sec (TPS), when compared to HDD configurations, and very low latencies of less than 50 seconds compared to > 1,000 seconds with HDDs.

These performance improvements result in significantly improved efficiency of data-analytics operations by providing faster time to results, and thus providing cost savings related to reduced operational expenses for business organizations.

Apache HBase

Apache HBase is an open-source, distributed, scalable, big-data database . It is a non-relational database (also called a NoSQL database), unlike many of the traditional relational database management systems (RDBMS). Apache HBase uses the Hadoop Distributed File System (HDFS) in distributed mode.

HBase can also be deployed in standalone mode, and in this case it uses the local file system. It is used for random, real-time read/write access to large quantities of sparse data . Apache HBase uses Log Structured Merge trees (LSM trees) to store and query the data . It features compression, in-memory caching, and very fast scans . Importantly, HBase tables can serve as both the input and output for MapReducejobs.

Some major features of HBase that have proven effective in supporting analytics workloads are:

- Linear and modular scalability
- Strictly consistent reads and writes
- Automatic and configurable sharding of tables, with shards being smaller and more manageable parts of large database tables

- Support for fault tolerant and highly available processing, through using an automatic failover support between RegionServers
- Convenient base classes for backing Hadoop MapReduce jobs with Apache HBase tables
- Uses the Java API for client access, as well as Thrift or REST gateway APIs, thus providing applications with a variety of access methods to the data
- Near-real-time query support
- Support for exporting metrics via the Hadoop metrics subsystem to files or the Ganglia open source cluster management and administration tool

The HBase architecture defines two different storage layers, the MemStore and the StoreFile. An object is written into MemStore first, and when the MemStore is filled to capacity, a flush-to-disk operation is requested . This flush-to-disk operation results in the MemStore contents being written to disk . HBase performs compaction/compression during the flush-to-disk operations (which are more frequent for heavy write workloads), merging different StoreFiles together .

HBase read requests will try to read an object first from the MemStore. In case of a read miss, the read operation is fulfilled from the StoreFile . With a heavy read workload, the number of concurrent read operations from the disk increases accordingly.

The HBase storage architecture for a distributed HBase instance is shown in Figure 1 . The figure shows the various components within HBase, including the HBase Master and the HBase Region Servers and it also shows how the various components access the underlying files residing in HDFS . Please visit some of the references in the References section for detailed information about the HBase Architecture.

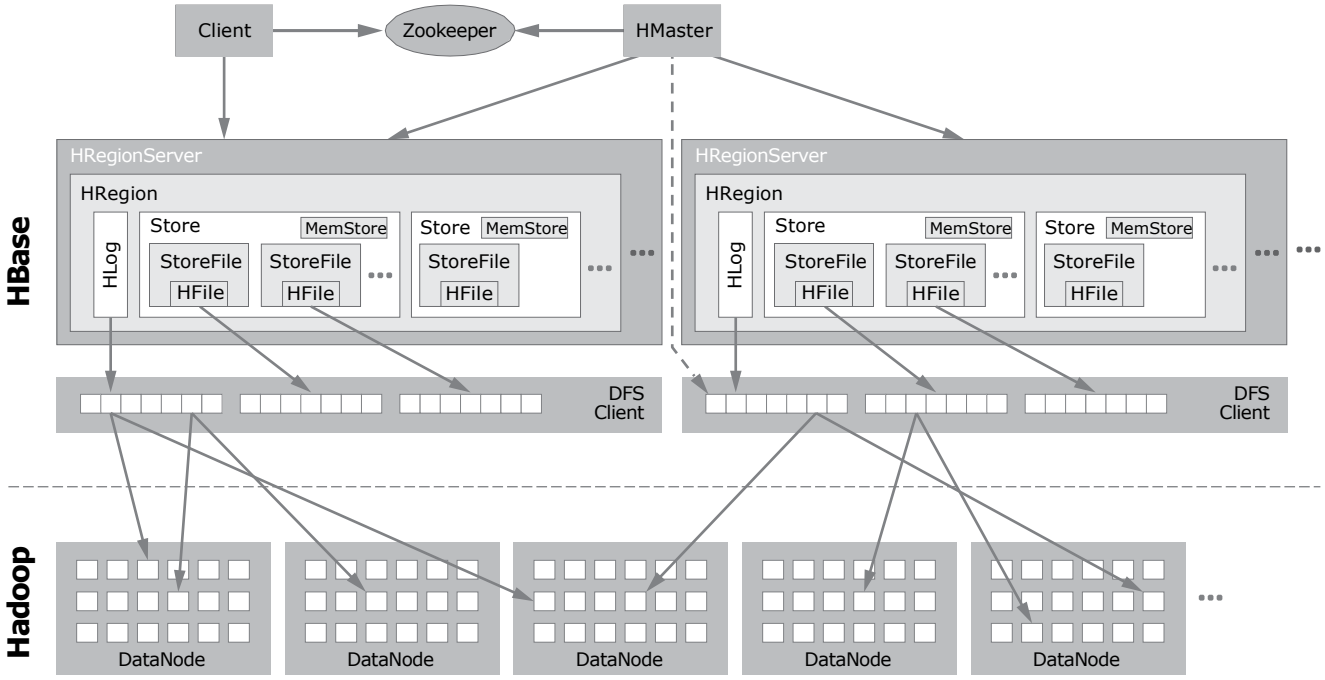

Figure 1: HBase Components and HDFS

YCSB Benchmark

The Yahoo! Cloud Serving Benchmark (YCSB) consists of two components:

- The client, which generates the load according to a workload type and records the latency and throughput associated with that workload .
- The workload files, which define the workload type by describing the size of the data set, the total number of requests, the ratio of read and write queries.

There are six major workload types in YCSB:

1. **Update-Heavy Workload A:** 50/50 update/read ratio
2. **Read-Heavy Workload B:** 5/95 update/read ratio
3. **Read-Only Workload C:** 100% read-only, maps closely to Workload B.
4. **Read-Latest Workload D:** 5/95 insert/read ratio, the read load is skewed towards the end of the key range, has similarities to Workload B.
5. **Short-Ranges Workload E:** 5/95 insert/read ratio, over a range of up to 100 records
6. **Read/modify/write Workload F:** 50/50 write/read ratio, similar to Workload A, but the writes are actual updates/modifications rather than just blind writes like in Workload A.

For additional information about YCSB and workload types, please visit the official YCSB page, as listed in the References section of this paper.

This technical paper describes the first three types of Workloads, A, B and C, and the latency and transactions-per-second results for these workloads using different storage configurations . These workloads provide a good mix of update-heavy and read-heavy test scenarios.

Test Design

A standalone HBase instance using the local file system was set up for the purpose of determining the benefits of using solid-state disks within an HBase environment . The testing was conducted with the YCSB benchmark, using different workloads (Workloads A, B and C) and different workload dataset sizes (64GB and 256GB) . The testing also included HBase, as it was deployed on different storage configurations . The transactions per second (TPS) and latency results for each of these tests were collected and analyzed. The results are summarized in the Test Validation and Results Summary section of this paper.

Test Environment

The test environment consisted of one Dell PowerEdge™ R720 server with two Intel® Xeon® CPUs (processors), with 12 cores per CPU) and 96GB of DRAM, which served as the HBase server, and one Dell PowerEdge R720, which served as a client to the test HBase server (a client to the test HBase server) running the YCSB workloads . For testing purposes, a 1GbE network interconnect was used between the server and the client . The local storage configuration on the HBase server varied between an all-HDD and an all-SSD configuration (more details are provided in the Test Methodology section of this paper) . The dataset sizes used by the YCSB workloads on the clients were 64GB and 256GB .

Technical Component Specifications

Hardware	Software if applicable	Purpose	Quantity
Dell Inc. PowerEdge R720 • Two Intel® Xeon® CPU E5-2620 0 @ 2GHz • 96GB memory	• CentOS 5.10, 64-bit • Apache HBase 2.4.9	HBase server	1
Dell Inc. PowerEdge R720 • Two Intel Xeon CPU E5-2620 0 @ 2GHz • 96GB memory	• CentOS 5.10, 64-bit • YCSB 0.1.4	YCSB client	1
Dell PowerConnect 2824 24-port switch	1GbE network switch	Data Network	1
500GB 7.2K RPM Dell SATA HDDs	Configured as a single RAID0 volume	HBase local file system	6
480GB CloudSpeed Ascend™ SATA SSDs	Configured as a single RAID0 volume	HBase local file system	6

Table 1: Hardware components

Software	Version	Purpose
CentOS Linux	5.10	Operating system for server and client
Apache HBase	2.4.9	HBase database server
YCSB	0.1.4	Client workload benchmark

Table 2: Software components

Compute Infrastructure

The Apache HBase server is a Dell PowerEdge R720 with two Intel Xeon 2 .00GHz CPU E5-2620 and 96GB of memory . The client's compute infrastructure is the same as the server .

Network Infrastructure

The client and the HBase server are connected to a 1GbE network via the onboard 1GbE NICs using a Dell PowerConnect 2824 24-port switch. This network was used as the primary means of communication between the client server and the HBase server.

Storage Infrastructure

The Apache HBase server uses six 500GB 7.2K RPM Dell SATA HDDs in a RAID0 configuration for the all-HDD configuration . The HDDs are replaced by six 480GB CloudSpeed Ascend SATA SSDs for the all-SSD test configuration .

The RAID0 volume is formatted with the XFS file system and mounted for use within HBase .

HBase configuration

Most of the default HBase configuration parameters were used for testing purposes across all of the workloads and storage configurations under test. A few HBase configuration parameters were modified during the testing, and these are listed in the tables, below:

Name of parameter	Default	Modified
hbase.hregion.memstore.chunkpool.maxsize	0	0.5
hbase.hregion.max.filesize	1073741824	53687091200
hbase.regionserver.handler.count	10	150

Table 3: HBase configuration parameter for hbase-env.sh

Name of parameter	Default	Modified
hbase.hregion.memstore.chunkpool.maxsize	0	0.5
hbase.hregion.max.filesize	1073741824	53687091200
hbase.regionserver.handler.count	10	150
hbase.block.cache.size	0.2	0.4
hbase.hregion.memstore.mslab.enabled	False	True
habse.hregion.memstore.mslab.maxallocation	262144	262144

Table 4: HBase configuration parameter for hbase-site.xml

Test Validation and Results Summary

Test Methodology

The primary goal of this technical paper is to showcase the benefits of using SSDs within an HBase environment. To achieve this goal, SanDisk tested three main YCSB workloads (Workloads A, B and C) against multiple tests configurations . The distinct test configurations were set up by varying the underlying storage configuration for the standalone HBase server (using either an all-HDD or an all-SSD configuration), and the size of the data set (both 64GB and 256GB datasets were used in the testing). These storage configuration and data size variations are described below:

Storage Configuration

Storage configuration for the standalone HBase instance was varied between an all-HDD and an all-SSD configuration:

1 . **All-HDD configuration:** The HBase server uses a single RAID0 volume with 6 HDDs for the single instance HBase .

2. **All-SSD configuration:** In this configuration, the HDDs of the first configuration are replaced with SSDs and a corresponding RAID0 volume for use with the HBase instance.

YCSB Workload Dataset Size

Two YCSB dataset sizes were used:

1. **64GB dataset size:** This particular dataset is smaller than the size of the available memory. As a result, all of the data is stored in-memory, thus avoiding disk I/O as much as possible.
2. **256GB dataset size:** This dataset size is larger than the size of the available memory, therefore requiring more disk I/O operations as compared to the smaller 64GB dataset size .

Update-Heavy Workload (Workload A)

The YCSB Workload A is referred to as an update-heavy workload . It is a workload with a 50/50 ratio of updates and reads. The distribution of the update and read operations can be uniformly spread out across the entire data set—"uniform" distribution—or can be of type named "Zipfian", in which case a portion of the dataset is accessed more frequently than the rest of the dataset. A Zipfian distribution is named after Zipf's Law, an empirical law of mathematical statistics, regarding statistical distribution of data.

Workload A models the behavior of applications like a session store, where recent user actions are recorded .

Throughput TPS Results

The throughput results for YCSB Workload A are summarized in Figure 2 . The X axis on the graph shows the different dataset sizes (64GB and 256GB) . The Y axis shows the throughput in terms of transactions per second. For each dataset size, the results for the uniform and Zipfian distributions are shown for the all-HDD and all-SSD configuration via different colored bars (see legend in figures for more details) . These same results are summarized in Table 5.

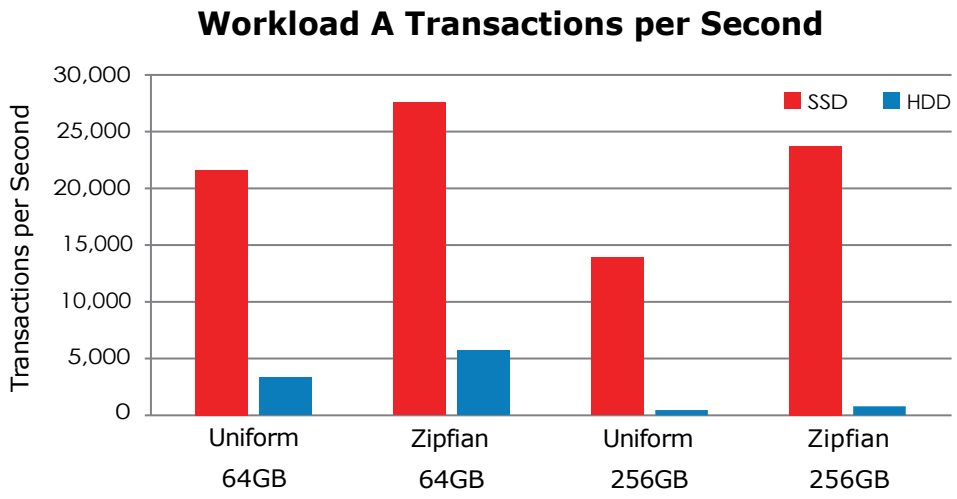

Figure 2: Throughput comparison for Workload A

YCSB Workload Types	Dataset Size	YCSB Workload Distribution	TPS All-HDD	TPS All-SSD
Workload A (50r/50w)	64GB	Uniform	3,263	21,681
Workload A (50r/50w)	64GB	Zipfian	5,705	27,561
Workload A (50r/50w)	256GB	Uniform	411	13,925
Workload A (50r/50w)	256GB	Zipfian	748	23,769

Table 5: Throughput results summary for Workload A

Latency Results

The read latency results for YCSB Workload A are summarized in Figure 3 . The X axis on the graph shows the read latencies for the different dataset sizes (64GB and 256GB), for both uniform and Zipfian distributions, and the Y axis shows the latency in seconds. These same results are summarized in Table 6.

The write latency results for YCSB Workload A are not shown in the figures . HBase uses a write-ahead log, which writes any updates/edits to memory, and these updates are later flushed to disk when the flush interval is reached. Due to the use of memory for updates/edits, the write latencies are between 0-1 milliseconds (ms), which get recorded as 0 seconds . As a result, the write latencies are not shown in the figures . They are, however, recorded in Table 6 .

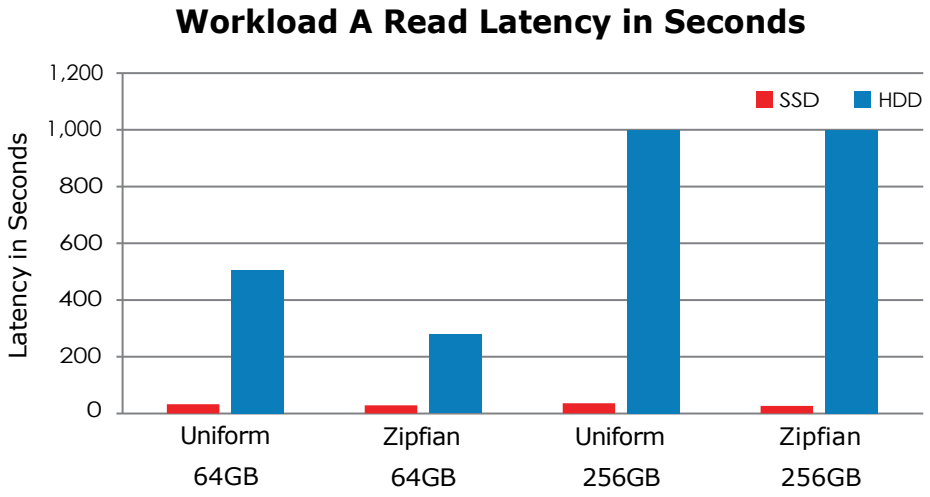

Figure 3: Latency comparisons for Workload A

YCSB Workload Types	Storage Configuration	YCSB Workload Distribution	64GB		256GB	
			Read	Write	Read	Write
Workload A (50r/50w)	HDD	Uniform	505	0	>1,000	0
Workload A (50r/50w)	SSD	Uniform	36	0	38	0
Workload A (50r/50w)	HDD	Zipfian	278	9	>1,000	0
Workload A (50r/50w)	SSD	Zipfian	33	0	30	0

Table 6: Latency results summary for Workload A

Read-Intensive Workload (Workload B)

YCSB Workloads B and C are read-intensive workloads . Workload B generates 95% read operations, with just 5% update operations in that workload . Workload C is a 100% read workload . Again, for both these workloads, the frequency of access across all parts of the dataset can be uniform, or a Zipfian distribution of access can be used where part of the data is being accessed more frequently .

These workloads simulate applications like photo-tagging, where most operations involve reading tags—or user profile caches, and 100% of the workload is reading from the user profile caches . These types of workloads are being increasingly adopted today, especially for Web-enabled cloud-computing workloads and social media workloads .

Throughput Results

The throughput results for YCSB Workload B are summarized in Figure 4 . The X axis on the graph shows the different dataset sizes (64GB and 256GB) . The Y axis shows the throughput in terms of transactions per second . For each dataset size, the results for the uniform and Zipfian distributions are shown for the all-HDD and all-SSD configuration via different colored bars . These same results are summarized in Table 7.

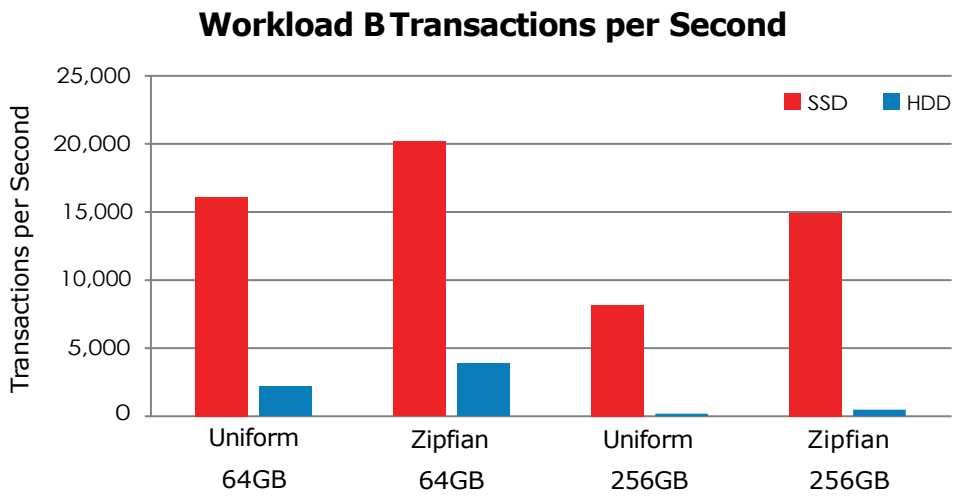

Figure 4: Throughput comparison for Workload B

YCSB Workload Types	Dataset Size	YCSB Workload Distribution	TPS All-HDD	TPS All-SSD
Workload B (95r/5w)	64GB	Uniform	2,186	16,051
Workload B (95r/5w)	64GB	Zipfian	3,870	20,169
Workload B (95r/5w)	256GB	Uniform	219	8,193
Workload B (95r/5w)	256GB	Zipfian	410	14,943

Table 7: Throughput results summary for Workload B

Latency results

The read latency results for YCSB Workload B are summarized in Figure 5 . The X axis on the graph shows the read latencies for the different dataset sizes (64GB and 256GB), for both uniform and Zipfian distributions, and the Y axis shows the latency in seconds. These same results are summarized in Table 8.

The write latency results for YCSB Workload B are not shown in the figures . Since this particular workload is a read-heavy workload, there is a very small proportion of writes for Workload B . As a result, write latencies are not significant for this workload . For the small proportion of writes, HBase writes any updates/edits to memory, with latencies of 0-1 milliseconds (ms) which get recorded by YCSB as 0 seconds .

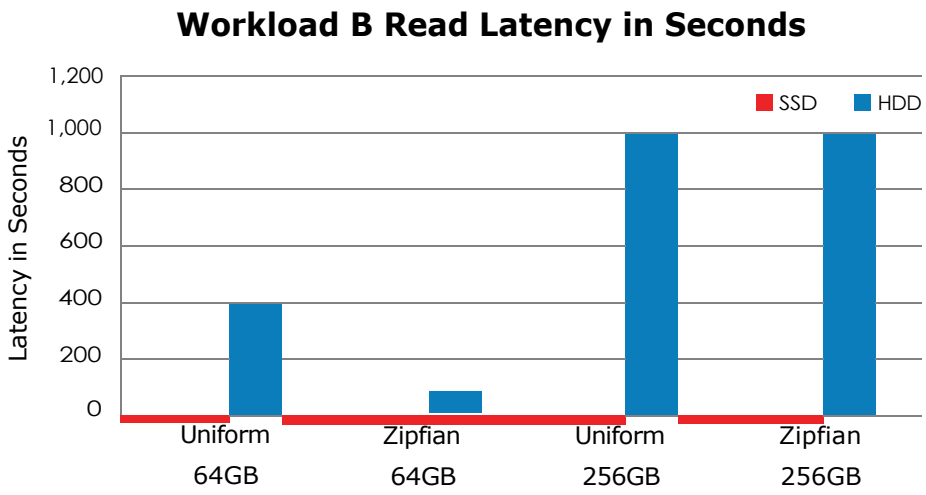

Figure 5: Latency comparisons for Workload B

YCSB Workload Types	Storage Configuration	YCSB Workload Distribution	64GB		256GB	
			Read	Write	Read	Write
Workload B (50r/50w)	HDD	Uniform	427	0	>1,000	0
Workload B (50r/50w)	SSD	Uniform	29	0	37	0
Workload B (50r/50w)	HDD	Zipfian	76	0	>1,000	0
Workload B (50r/50w)	SSD	Zipfian	23	0	31	0

Table 8: Latency results summary for Workload B

Read-Only Workload (Workload C)

Throughput Results

The throughput results for YCSB Workload C are summarized in Figure 6 . The X axis on the graph shows the different dataset sizes (64GB and 256GB) . The Y axis shows the throughput in terms of transactions per second . For each dataset size, the results for the uniform and Zipfian distributions are shown for the all-HDD and all-SSD configuration via different colored bars (see the legend in the figures for more details) . These same results are summarized in Table 9.

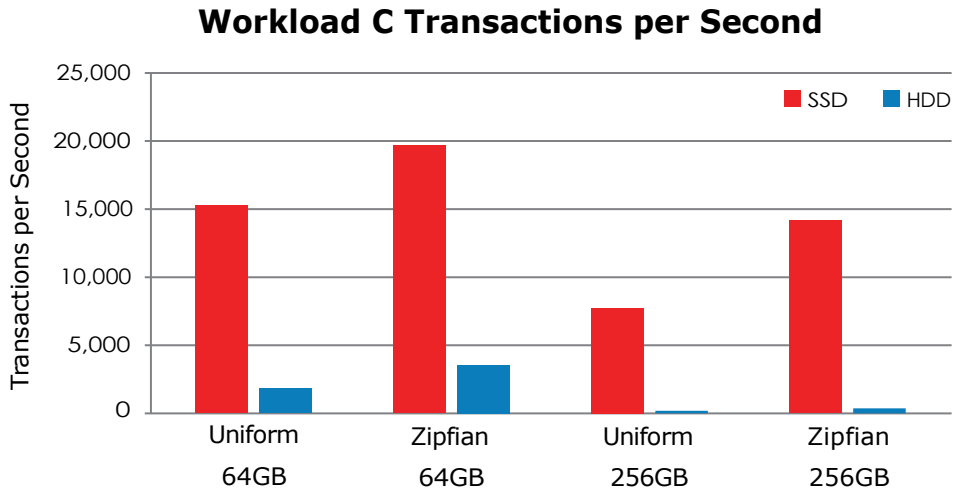

Figure 6: Throughput comparison for Workload C

YCSB Workload Types	Dataset Size	YCSB Workload Distribution	TPS All-HDD	TPS All-SSD
Workload C (100r/0w)	64GB	Uniform	1,865	15,334
Workload C (100r/0w)	64GB	Zipfian	3,582	19,756
Workload C (100r/0w)	256GB	Uniform	209	7,799
Workload C (100r/0w)	256GB	Zipfian	373	14,214

Table 9: Throughput results summary for Workload C

Latency Results

The read latency results for YCSB Workload C are summarized in Figure 7. The X axis on the graph shows the read latencies for the different dataset sizes (64GB and 256GB), for both uniform and Zipfian distributions, and the Y axis shows the latency in seconds. These same results are summarized in Table 10.

The write latency results for YCSB Workload C are not shown in the figures . Since this particular workload is a read-only workload, there are no write operations in Workload C. As a result, write latencies are not applicable for this workload.

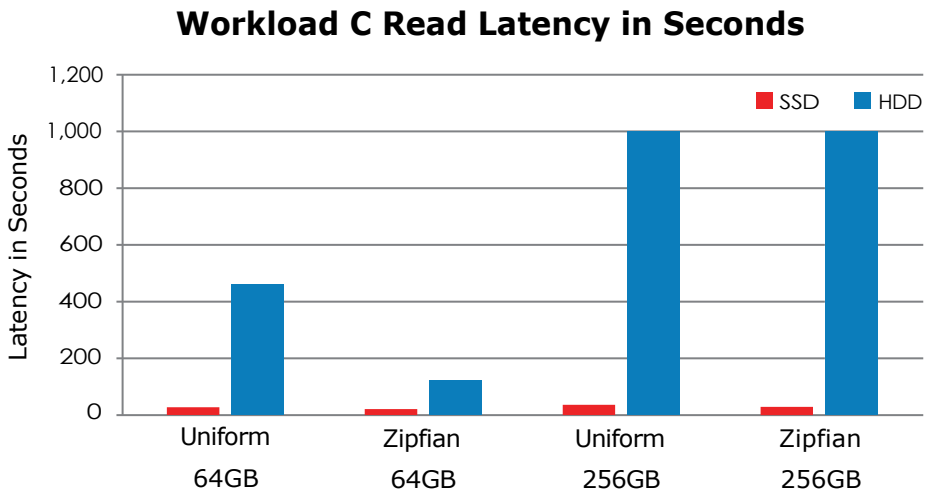

Figure 7: Latency comparisons for Workload C

YCSB Workload Types	Storage Configuration	YCSB Workload Distribution	64GB		256GB	
			Read	Write	Read	Write
Workload C (50r/50w)	HDD	Uniform	463	0	>1,000	0
Workload C (50r/50w)	SSD	Uniform	28	0	37	0
Workload C (50r/50w)	HDD	Zipfian	124	0	>1,000	0
Workload C (50r/50w)	SSD	Zipfian	22	0	31	0

Table 10: Latency results summary for Workload C

Conclusion

The key takeaways from the YCSB tests for Apache HBase, comparing all-HDD and all-SSD storage configurations, are as follows:

1 . SanDisk SSDs show a dramatic increase (30-40x) in number of transactions per second for update-heavy and read-heavy workloads for different types of data access, whether uniform across the dataset or targeted to only a portion of the dataset . This translates to a higher rate of query processing in a variety of different data analytics applications,

2 . SanDisk SSDs also benefit HBase workloads with significantly lower latencies than mechanically driven HDDs . Lower latencies allow quicker query-response time, therefore increasing the productivity and efficiency of analytics operations .

3 . Following is a graphical summation of the key takeaways for this SanDisk test of the server/storage platform supporting the Apache HBase workload:

30-40x Transactions Per Second with SanDisk SSDs

SanDisk SSDs within the HBase environment provide dramatic improvements (30x – 40x) in transactions/second when compared to HDDs .

Improved performance seen for update-heavy workloads as well as read-heavy workloads, for uniform and zipfian distribution .

Significantly lower latencies (**< 50 seconds**) with SanDisk SSDs

SanDisk SSDs consistently provide read latencies of < 50 seconds, whereas with HDDs, the latency is consistently > 1000 seconds (for the 256GB data sets).

Lower latencies translate to much quicker response times for queries .

Summary

The higher transactions/second and lower latency results seen with CloudSpeed® SSDs directly translate to faster query processing, thus reducing the time-to-results . In an organization's IT department, this results in significant improvements in business process efficiency and therefore cost-savings.

The tests described in this paper clearly demonstrate the immense benefits of deploying SanDisk's CloudSpeed SATA solid-state drives (SSDs) drives within HBase environments . Using CloudSpeed drives will help deploy a cost-effective and process-efficient data analytics environment by providing increased throughput and reduced latency for data analytics queries.

It is important to understand that workloads will vary significantly in terms of their I/O access patterns . Therefore, all workloads many not benefit equally from the use of SSDs for an HBase deployment. For the most effective deployments, customers will need to develop a proof of concept for their deployment and workloads, so that they can evaluate the benefits that SSDs can provide to their organization.

References

SanDisk website: www.sandisk.com

Apache Hadoop Distributed File System: http://hadoop.apache.org/docs/r1.2.1/hdfs_design.html Apache HBase: http://HBase.apache.org

HortonWorks HBase: http://hortonworks.com/hadoop/hbase/

Yahoo LABS Yahoo! Cloud Serving Benchmark: http://research.yahoo.com/Web_Information_Management/YCSB/

HBase Storage Architecture: http://files.meetup.com/1228907/NYC%20Hadoop%20Meetup%20-%20Introduction%20to%20HBase.pdf HBase performance writes: http://hbase.apache.org/book/perf.writing.html