Flash Memory Database (FMDB) Issues and Promising

Nooruldeen Nasih Qader[1], Mohammed Anwar Mohammed[2], Danial Abdulkareem Muhammed[3]

Department of Computer Science, University of Sulaimani, IRAQ.

[1] nuraddin.nasih@gmail.com, [2] hama987@gmail.com, [3] dam987@yahoo.com

ABSTRACT

Due to rapidly growing size of database, storage considers one of the obstacles for database performance. The structure of current hard disk media contains traditional technology with some changes, which cannot maintain the database workloads. On the other hand, flash memory promises faster performance for database applications, flash memory has ended up to be the significant consistent data storage tool for mobile and embedded devices.

Flash memory-based development platform supplies the adaptability to provide a wide range of hardware and software implementations in the cost-performance trade off scale. In this paper we have reviewed flash memory technology, the flash memory database trend. We present the role of flash memory in: importance of small database in small businesses, and database security. We presented advantages and weaknesses, utilizing this technology required an efficient flash memory-based database system; also, this issue counts one of the challenges.

Keywords: Flash memory, business, database, security, storage

INTRODUCTION

Nowadays the crucial aim of enterprises is to reduce costs and improve performance. It is the IT's duty for backing the quick growth of data, besides the real-time processing desires and detailed analytics demands, all with decreasing the operational costs. Although all the demands have increased the heritage technology of spinning disks has not progressed with them.

Due to rapidly growing size of database, storage considers one of the obstacles for database performance. The structure of current hard disk media contains traditional technology with some changes, which cannot maintain the DB workloads. On the other hand, flash memory promises faster performance for database applications (Anciaux, Bouganim, & Pucheral).

There are some advantages of flash memory over hard disks, they are lower energy consumption, smaller size, lower noise and heat, damage resistance and high speed, figure 1 shown table space read performance and illustrated that using flash make the performance much better. Nonetheless, limited wear and high price (relatively) are the disadvantages of flash memory (Batni & Safaei, 2014). Even though, flash memory price has been decreased faster than hard disk ("SSD Hard Drive Upgrade Results » The Online Investing AI Blog," n.d.).

Flash memory-based development platform supplies the adaptability to provide a wide range of hardware and software implementations in the cost-performance trade off scale. Fast prototyping and easy evaluation of flash memory-based storage system designs is supposed to be provided by the proposed platform (Kim, Lee, Nam, & Min, 2008).

NAND (Negated AND or NOT AND) flash memory has ended up to be the significant consistent data storage tool for mobile and embedded devices (e.g. mobile phone, USB disk drive, mp3 player). This paper will discuss limitation of using erase-before-write and displays inadequate write performance of flash memory particularly when write operations are requested in an arbitrary order (NA, Moon, & Lee, 2011).

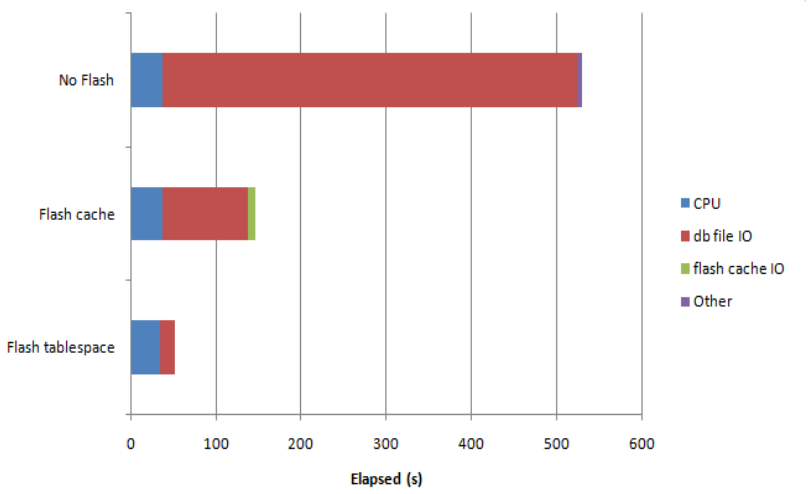

Figure 1. Flash disk read performance ("Guy Harrison - Yet Another Database Blog - Flash tablespace vs. DB Flash Cache," n.d.)

Rapidly growing sizes of DB makes the storage with high performance and significant capacity necessary and crucial; this is after mobile embedded systems evolved into data centric and multimedia-oriented applications. This is made flash memory to be an essential component in mobile embedded systems (Song, Tao, & Gao, 2010).

In addition, flash-based storage devices are now supposed to have incredible ability as an alternative storage medium that can replace magnetic disk drives and achieve much higher performance, this is because, flash memory density has increased and its price has decreased (Nam, Na, & Lee, 2010). Table 1 presents cost of flash memory and disks; the price of flash is coming down more rapidly than the price of spinning disk.

Table 1. Relative costs for (Graefe, 2007)

	NAND Flash	*SATA Disk*
Price and capacity	$999 for 32 GB	$80 for 250 GB
Price per GB	$31.20	$0.32
Time to read a 4-KB page	0.16 ms	12.01 ms
4-KB reads per second	6,200	83
Price per 4-KB read per second	$0.16	$0.96
Time to read a 256-KB page	3.98 ms	12.85 ms
256-KB reads per second	250	78
Price per 256-kB read per second	$3.99	$1.03

Leena and Luna International, Oyama, Japan.
(株) リナアンドルナインターナショナル, 小山市、日本.
ISSN: 2186-8476, ISSN: 2186-8468 Print
www.ajsc. leena-luna.co.jp

Data management on flash memory becomes a new great challenge. This result from, the rapid growth in capacity and availability of flash memories, designing an efficient flash memory-based DB system is one of these challenges (C. Chung & Hsu, 2014). In addition, many papers have been published by researchers, which discuss query processing algorithms and indexing technique on flash memory-based DB systems. Furthermore, they highlighting that these storage devices will replace the existing electronic mechanical hard disks in the near future (T. Chung, Park, Park, & Lee, 2006; Song et al., 2010; Teshome & Chung, 2010).

SYSTEM SOFTWARE FOR FLASH MEMORY: A SURVEY

Special software operations are required for flash memory-based applications during reading/writing data from/to flash memory this is because of its hardware characteristics. An example of its characteristics is erase-before-write architecture. That is, in order to update a location on a flash memory, it has to be first erased before the new data can be written to it. Additionally, the erase unit (block) is larger than the read or write unit (sector) (Nam et al., 2010; Suh, Moon, Efrat, Kim, & Lee, 2014). This is result in declining major performance of the overall flash memory system. As a result, the system software called FTL (Flash Translation Layer) should be introduced (T. Chung et al., 2006; Sang-won Lee, 2007; Na, Moon, & Lee, 2009; Park, Lee, Pyi, Lee, & Cho, 2014).

FLASH MEMORY TECHNOLOGY

There are two main categories of flash memory NOR and NAND. On the one hand, the properties of NOR memory are fast and simple access procedures, however, its storage capacity is slower and it is thus preferred to being program storage. On the other hand, significantly higher storage capacity is offered by NAND flash, and is more appropriate for storing large amounts of data(Lin, Chen, Wang, & Wu, 2014).

Currently there are mainly four kinds of Flash memory DB(Kim et al., 2008; Saxena & Swift, 2010).

FLASH MEMORY DB SYSTEM VS. FTL

FTL performs a number of actions in order to make linear flash memory appear to the system as a disk drive, and these actions are: first, creating “virtual” small blocks of data or sectors out of flash’s large erase block. Second, data will be managed on the flash so that it appears to be “written in place”, while it’s stored in different spots in the flash. Third, clean/erased places to store data are available because of managing the flash by FTL itself (*Understanding the Flash Translation Layer (FTL) Specification*, 1998).

Flash Technology

There are two different states in flash media which are erased and non-erased. While it’s in erase state a byte is either all zeros (0x00) or all ones (0xFF) depends on the flash device. As soon as the media enters erase state a certain bit of data can only be written to, and it’s regarded as unusable. Therefore, a significantly larger block of flash known as erase block should be erased. Additionally, in flash technology toggling single bits or bytes from a non-erased state back to an erased state is not allowed. These details are shielded from the file system by the FTL. In addition, FTL remaps the data passed to it by writing to unused data areas in the flash media. Moreover, it manages reclaiming the discarded data blocks for reuse(*Understanding the Flash Translation Layer (FTL) Specification*, 1998).

PREVIOUS WORKS ON FLASH MEMORY DB

R-tree

Block-oriented access over flash memory may result in introducing an important number of node updates for applications with spatial data management. For instance, Geographic Information Systems (GIS). This update may result in a great number of out-places updates and garbage collection over flash memory and damages its consistency.

R-tree index access of spatial data over flash memory possibly will capably handle fine-grained updates. Additionally, flash translation layer will be used directly for the implementation without any changes current application system (Wu, Chang, & Kuo, 2003).

Advantages of Using R-Tree

I. Improving system performance, overheads on flash-memory management, and energy dissipation.

II. Handling updates efficiently.

In-Page Logging (IPL)

It is a new design for flash memory based DB servers, and some limitations such as write latency has been solved by this new design. Additionally, it provides a high performance for flash-based DB servers comparing to disk-based DB servers. Moreover, access methods and traditional disk-based DB algorithms can perform its task effectively without any changes.

Advantages of Using In-Page Login

I. Despite reducing the modifications made to the overall DB server architecture, IPL supports gaining the best achievable performance from flash memory. Additionally, it helps running a full-fledged DB server on a wide scale of computing platforms with flash memory instead of magnetic disk drives.

II. Computing a platform that uses flash memory as a core storage device can handle large-scale transactional DB applications to run efficiently(S.-W. Lee & Moon, 2011; SW Lee, Moon, Park, Hwang, & Kim, 2010).

IPL B^+-tree

IPL B^+-tree is a new index structure, which based on IPL storage scheme. It overcomes the problems in the current disk-based B^+-tree index system.

Table 2 displays page utility for B-tree nodes on disk and flash memory (Graefe, 2007).

Why using this method:

1. High shock resistance, low access latency and low power consumption are made NAND flash memory to become the main permanent data storage medium for mobile and embedded devices. However, writing performance in flash memory is not good enough particularly when it comes to write operation in a random order.
2. With the current disk-based index structures and algorithms the preferable performance may not be gained because when updating, inserting or deleting data records in the DB the size of changes which made to the index structure is very small, especially in the range of 10s to 100s bytes (Kim et al., 2008).

Leena and Luna International, Oyama, Japan.
(株) リナアンドルナインターナショナル, 小山市、日本.
ISSN: 2186-8476, ISSN: 2186-8468 Print
www.ajsc. leena-luna.co.jp

Advantages of Using IPL B$^+$-Tree

I. By taking the advantages of IPL scheme it can decrease the number of overwrites in B$^+$-tree index. Additionally, it has all the features of B$^+$-tree index.

II. It can work well on various flash memory platforms without depending on a specific FTL; this is because of, using a raw flash device interface as an alternative to a block device interface.

This approach can be concluded by

In-Page logging concept is a useful and feasible solution for flash memory DB systems when implementing IPL based B$^+$-tree. Additionally, the conventional disk-based B$^+$-tree index performance is deeply depends on its FTL algorithms, and it is not appropriate for flash storage devices.

Problem with Conventional Design

First, when updating a single record the entire page that contains the record should be updated, this will cause the record to be overwritten. Additionally, when the accessing a small record in the DB is random and spread the total amount of overwritten data is likely to be much greater than the updated data. Second, assume that in the computing platform that the traditional disk-based DB system works on it, all magnetic disks are changed to flash memory. In case of updating data in place the whole erase unit should be erased after reading its content to the memory, and then filling the erase unit by writing the new content, this is caused by the erase-before-write restriction of flash memory. Another problem is, in case of updating data repeatedly the lifespan of flash memory will be shortened. Since the erase cycles of erase unit is limit before becoming statistically undependable.

Table 2. Page utility for b-tree nodes on disk and flash memory

			Access Time ms		*Utility / Time*	
Page size KB	*Records / page*	*Node Utility*	*HDD*	*Flash*	*HDD*	*Flash*
4	140	7	12	0.16	0.58	43.6
16	560	9	12.1	0.34	0.75	26.3
64	2240	11	12.2	1.07	0.9	10.3

FLASH MEMORY DB IN BUSINESS

Flash memory DB may be more suitable for small business in the following perspective: mobility, preserves privacy, security, less required infrastructure, and reduce cost operation. Therefore, we present a brief comparison between small and big DB in the context of business management, also, we present latest DB security framework suitable for small business.

Small DB vs. Big DB

Whenever we discuss DB protection, we begin by discussing extensive DBs maintained simply by large businesses usually. But it could be argued that biggest DB challenges of most are those confronted by the small businesses, which are struggling to get a basic security set up just. Today the business environment in which smaller businesses are operating, dictates

that new needs have to be applied in small sized businesses to improve their DB protection. They ought to adopt a new group of security equipment, which require simple installation, maintenance and use. Security equipment's protect the small company DBs from DB protection breaches.

In both cases, instead of going for the costly DB servers which additionally requires extra hardware as well as the extra expenses in training and handling, the flat file may be considered as a candidate due to its easy handling nature, fast accessing, and of course free of cost, but the main hurdle is the security aspect which are not up to the optimum level.

Although large companies are the targets of data breaches by hackers often, small businesses have to be more worried about their DB security also. In fact, small businesses also have much more to lose because of data breaches since they generally have fewer infrastructures set up with less IT personnel, so the threat of data loss is a lot higher in comparison to large companies. Although small businesses do not utilize the same security steps that large companies do, there are many other security measures by which your small business can reinforce its DB security (K. K.-Y. Lee, Tang, & Choi, 2013).

DB Security

DB management systems are increasingly being used to store information about all aspects of an enterprise. The data stored in a DBMS is often vital to the business interests of the organization and is regarded as a corporate asset. In addition to protecting the intrinsic value of the data, corporations must consider ways to ensure privacy and to control access to data that must not be revealed to certain groups of users for various reasons.

New framework has proposed by (Qader, 2014) called Multilayer Checkpoints for DB Security (MLC-DBS). It discussed factors influenced required layered to ensure DB security and what specific layer is more necessary; there are many techniques of security, which differs from multi sides, reliability, requirement, cost, speed, policy... etc. That is, a technique may be the best for some place, whereas the same technique is totally insufficient for another place.

The strategy framework proposed by this paper will undoubtedly be highly beneficial for small DB and big DB because both sizes may contain valuable data. All they need to implement the framework for any size of the DB is deciding how many layers is required and what specific layer is more necessary. MLC-DBS is the flexible solution for big and small DB, and systems that protected by MLC-DBS are more secure than others, because, if one or some of the checkpoints are cracked, there are other that remains the system in secure.

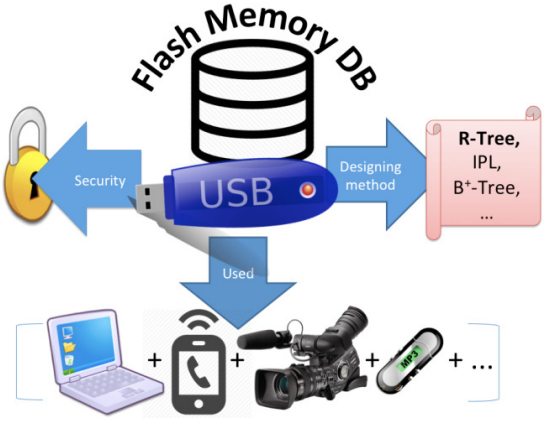

Figure 2. Flash memory DB issues and promises

CONCLUSION AND FUTURE WORK

Modern life need more DB, DB store sensitive and private data therefore securing DB consider crucial. Flash memory DB may be more suitable for small business in the following perspective: mobility, preserves privacy, security, less required infrastructure, and reduce cost operation. Therefore, we present a brief comparison between small and big DB in the context of business management, also, we present latest DB security framework suitable for small business. These features encourage researchers to: more utilizing flash memory DB, develop this technology, and overcome weakness (Anciaux, Bouganim, & Pucheral, 2014; Li, Wang, Wang, & Li, 2013). In this paper, as shown in Figure 2, we reviewed the works of integrating flash memory and DB, also we discuss the importance of small DB in small businesses, and we present latest DB security framework. Small businesses have much more to lose because of data breaches since they generally have fewer infrastructures set up with less IT personnel, so the threat of data loss is a lot higher in comparison to large companies.

In the next work we will implement varies techniques of information security with flash memory DB using different DB management system.

REFERENCES

[1] Anciaux et al. (2014). MILo-DB: a personal, secure and portable database machine. *Distributed and Parallel Databases*, *32*(1), 37–63. Retrieved from http://link.springer.com/article/10.1007/s10619-012-7119-x

[2] Batni, A., & Safaei, F. (2014). A New Multi-Tiered Solid State Disk Using Slc/Mlc Combined Flash Memory. *International Journal of Computer Science, Engineering and Information Technology (IJCSEIT)*, *4*(2), 11–23. Retrieved from http://arxiv.org/abs/1405.2157

[3] Chung, C., & Hsu, H. (2014). Partial Parity Cache and Data Cache Management Method to Improve the Performance of an SSD-Based RAID. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, *22*(7), 1470–1480.

[4] Chung et al. (2006). System software for flash memory: a survey. In *Embedded and Ubiquitous Computing* (pp. 394–404). Berlin: Springer. **doi:10.1007/11802167_41**

[5] Graefe, G. (2007). The five-minute rule twenty years later, and how flash memory changes the rules. *Proceedings of the Third International Workshop on Data Management on New Hardware (DaMoN 2007)*, (August). Retrieved from http://dl.acm.org/citation.cfm?id=1363198

[6] Harrison, G. (2010, January 25). Flash tablespace vs. DB Flash Cache [Blog]. Retrieved July 31, 2014, from http://guyharrison.squarespace.com/blog/2010/1/24/flash-tablespace-vs-db-flash-cache.html

[7] Kim et al. (2008). Flash memory-based development platform for homecare devices. *2008 IEEE International Conference on Systems, Man and Cybernetics*, 2259–2263. **doi:10.1109/ICSMC.2008.4811629**

[8] Lee et al. (2013). Alternatives to relational database: comparison of NoSQL and XML approaches for clinical data storage. *Computer Methods and Programs in Biomedicine*, *110*(1), 99–109. **doi:10.1016/j.cmpb.2012.10.018**

[9] Lee, S. (2007). Design of Flash-Based DBMS : An In-Page Logging Approach. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data* (pp. 55–66). doi:10.1145/1247480.1247488

[10] Lee et al. (2010). Accelerating In-Page Logging with Non-Volatile Memory. *IEEE Data Eng. Bull.*, 41–47. Retrieved from http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Accelerating+In-Page+Logging+with+Non-Volatile+Memory#0

[11] Lee, S. W., & Moon, B. (2011). Transactional In-Page Logging for multiversion read consistency and recovery. In *Data Engineering (ICDE), 2011 IEEE 27th International Conference on* (pp. 876–887). doi:10.1109/ICDE.2011.5767889

[12] Li et al. (2013). FBSA: A Flash Memory-based Scheduling Algorithm for Optimistic Replication. *Information Technology Journal*, *12*(1), 71–79.

[13] Lin et al. (2014). HDC: An adaptive buffer replacement algorithm for NAND flash memory-based databases. *Optik - International Journal for Light and Electron Optics*, *125*(3), 1167–1173. **doi:10.1016/j.ijleo.2013.07.162**

Leena and Luna International, Oyama, Japan.
(株) リナアンドルナインターナショナル, 小山市、日本.
ISSN: 2186-8476, ISSN: 2186-8468 Print
www.ajsc. leena-luna.co.jp

[14] Na et al. (2009). In-Page Logging B-Tree for Flash Memory. DASFAA '09. In *Proceedings of the 14th International Conference on Database Systems for Advanced Applications* (pp. 755–758). doi:10.1007/978-3-642-00887-0_66

[15] Na et al. (2011). IPL B-tree for Flash Memory Database Systems. *Journal of Information Science and Engineering*, *127*(2010), 111–127. Retrieved from http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:IPL+B-tree+for+Flash+Memory+Database+Systems#0

[16] Nam et al. (2010). A hybrid flash memory SSD Scheme for Enterprise Database applications. In *12th International Asia-Pacific Web Conference* (pp. 39–44). doi:10.1109/APWeb.2010.70

[17] Park et al. (2014). Improvement of the multi-level cell performance by a soft program method in flash memory devices. *Solid State Electronics*, *94*, 86–90.

[18] Qader, N. N. (2014). Strategic Framework of Multilayer Checkpoint For DB Security. *International Journal of Computer Engineering and Applications*, *V*(III), 53–60.

[19] Saxena, M., & Swift, M. M. (2010). Revisiting Database Storage Optimizations on Flash. Online Available at www.minds.wisconsin.edu/bitstream/handle/1793/60702/TR1671.pdf?sequence=1

[20] Song et al. (2010). Performance Optimization for Flash Memory Database in Mobile Embedded System. In *Education Technology and Computer Science, International Worksho*p (pp. 35-39). doi:10.1109/ETCS.2010.109

[21] SSD Hard Drive Upgrade Results. The Online Investing AI Blog. (n.d.). Retrieved August 01, 2014, from http://www.onlineinvestingai.com/blog/ssd-hard-drive-upgrade-results/

[22] Suh et al. (2014). Memory efficient and scalable address mapping for flash storage devices. *Journal of Systems Architecture*, *60*(4), 357–371.

[23] Teshome, S., & Chung, T. (2010). Query cost estimation for read intensive flash memory based database systems. In *International Conference on Electronics and Information Engineering (ICEIE 2010)*, *1*(Iceie), V1–496–V1–498. doi:10.1109/ICEIE.2010.5559682

[24] Intel Corporation. (1998). *Understanding the Flash Translation Layer (FTL) Specification*. (p. AP–684).

[25] Wu et al. (2003). An Efficient R-tree Implementation over Flash-memory Storage Systems. In *Proceedings of the 11th ACM International Symposium on Advances in Geographic Information Systems* (pp. 17–24). New York, NY, USA: ACM. doi:10.1145/956676.956679