ABSTRACT

Most of systems using a large DBMS have been using disk-based storages consisting of hard disk drives. Recently, flash memory storages have appeared, which have great advantages like low access latency, low power consumption, light weight and shock resistance. These flash memory storages have gradually been successes as an alternative storage of traditional hard disk drives. However, because it is still a big price gap between the hard disk and flash memory device, they’re still using hard disk-based storages. Also systems to perform online transactions and batch jobs at the same time have performed batch jobs on a small online transaction time because of poor performance. Therefore, if we use flash memory devices and hard disk drivers with, it will be very efficient in terms of price, performance and reliability. In addition, it is possible to shorten the recovery time of DBMS. This paper proposes to move the REDO logs and ARCHIVE logs on flash memory storage device appropriately depending on the operating and working environments. By utilizing a small amount of flash memory devices, the performance of batch jobs has been increased to 40-80%, DBMS recovery time was reduced to 5-8%.

KEYWORDS

Flash memory, OLTP, Batch

1 INTRODUCTION

In the last few years, while the solid state drive (SSD) technology advances, the SSD has become an alternative to disk storage in a variety of fields, including the mass of an enterprise storage system [1, 2]. Nevertheless, the price per capacity of the SSD is much higher than the price per capacity of the hard disk driver (HDD), this trend is expected to continue for some time (shown in Table 1). Therefore, in most cases of the application dealing with the data of large size, to complement the performance of the hard disk using the SSD is more reasonable to compensate the performance rather than to completely replace the HDD to the SSD in consideration the of the economic factors. Also, methods using an SSD as a cache between RAM buffer and the hard disk have been recently studied in commercial DBMS such as Oracle, IBM DB2 and MS SQL Server, have been shown a very high level of satisfaction in price and performance [3-5]. However, all studies have been made around OnLine Transaction Processing (OLTP).

Table 1. The price of HDD and SSD

<table>
<thead>
<tr>
<th>Storage Media</th>
<th>Capacity of GB</th>
<th>List Price of $ (/$GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDD</td>
<td>1,024</td>
<td>54.99 (0.05)</td>
</tr>
<tr>
<td>SSD</td>
<td>128</td>
<td>125.90 (0.98)</td>
</tr>
</tbody>
</table>

* HDD : WD Blue 1 TB, 3.5", 7200 RPM, SATA 6Gb/s, 64 MB Cache - WD10EZEX
** SSD : Samsung Electronics MZ-7PD128BW 840 Pro 2.5" 128GB SATA 6Gbps

In fact, if we consider the situation mainly conducted in OLTP and batch jobs is being made simultaneously in one system and batch operations are conducted primarily at night, the least OLTP time in order to prevent the OLTP performance degradation, load balancing of...
performance is needed in simple OLTP performance, as well as a batch jobs. In this paper, we describe the implementation and experiment on commercial ORACLE DBMS in case of proceeding simultaneously for OLTP and batch jobs with using the SSD.

Random access speed of the SSD (random access latency) is much faster than speed of the HDD. Also, random write speed of the SSD is faster than speed of the HDD. However, random write speed of the SSD is still slow compared with the random read of the SSD. To take full advantage of these features, if we move a part of DBMS areas to take advantage of random read to the SSD we can see the improvement in the performance of the batch. In addition, we showed when using the SSD than the HDD DBMS recovery time can be shortened.

The key contributions of this work can be summarized as follows.
First, by using the features of the SSD, we have shown that the performance of the batch can be improved significantly.

Second, if we store the REDO log and Archive log on the SSD, the tasks of bulk data upload batch jobs can be significantly accelerated.

The contents of this paper were implemented in ORACLE 11GR2, performance evaluations were measured the working hours to perform batch upload large amounts of data in a realistic OLTP environment. Results of experiments showed performance increase of 40-80% compared to the conventional HDD-based ORACLE, showed the DBMS failure recovery time is reduced by 5-8%.

The rest of this paper is organized as follows.
Section 2 presents a few key features of flash memory SSD, redo log and archive log of ORACLE. Section 3 describes the experimental settings that will be used in the following sections. In Section 4, we analyze the performance gain that can be obtained by adopting the SSD as stable storage for log. Lastly, Section 5 summarizes the contributions of this paper.

2 Background and Related Work

The SSD is a non-volatile storage device of based on NAND-type flash memory, which has several advantages including low access latency, low power consumption, light weight and shock resistance. So the SSD has been successes as an alternative storage of traditional magnetic disk drives for a wide range of computing platforms. In this section, the characteristics of the SSD as a storage medium for DBMS are briefly summarized. We then present REDO and ARCHIVE of ORACLE DBMS.

2.1 Characteristics of the SSD

The SSD is a pure electronic device, and has not machinery parts like disk arms of the HDD. Therefore, unlike the HDD, the SSD can provide constant speed of random data access. In other words, the time of data access in the SSD is almost linearly proportional to the amount of data regardless of their physical locations in the SSD. This is implemented as dual-channel architecture. The ability of the SSD to increase bandwidth through parallel read/write operations is one of the key characteristics (shown in Table 2).

<table>
<thead>
<tr>
<th>Storage Media</th>
<th>RAND(IOPS)</th>
<th>SEQ(MB/s)</th>
<th>Capacity of GB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Read</td>
<td>Write</td>
<td>Read</td>
</tr>
<tr>
<td>HDD*</td>
<td>409</td>
<td>343</td>
<td>156</td>
</tr>
<tr>
<td>SSD**</td>
<td>65,803</td>
<td>13,979</td>
<td>534</td>
</tr>
</tbody>
</table>

* HDD : Segate Cheetah 15.6K 146.8GB (SAS 6Gb/s)
** SSD : Samsung 840 Pro 256GB (SATA 3)

On the other hand, no data in the SSD can simply be updated by overwriting it on the spot. In order to update a data stored in the SSD, a pre-erase operation must be conducted, this operation takes a relatively long time. In addition, the pre-erase operation cannot be conducted partially on a particular data or disk sector, which can only be done for an entire block containing the data of the SSD. There are many gaps to the read and write speed of the
SSD, the speed of random read is almost 4 times faster than that of random write. This is simply because it takes longer to write a block until reaching a stable status than to read the status from a block.

All these operations of the SSD are controlled by a firmware layer known as flash translation layer (FTL) [6, 7]. The FTL is at the heart of several essential functions of the SSD, which have address mapping, wear leveling and so on. Though the SSD is a pure electronic device, the latency of accessing data is not entirely free. If a read or write request is occurred from a system, the SSD controller interpret and process the I/O command. At that time, logical addresses are mapped to physical addresses, and if mapping information is altered by a write operation, then the mapping table in the SSD is updated [1, 2]. In addition, to the minimum energy consumption, a small amount of SRAM is used in the one-chip controller for program code, data and buffer memory.

2.2 REDO and ARCHIVE of ORACLE

REDO log is the most crucial structure for recovery operations, which consists of two or more preallocated files that store all changes made to the database. Every instance of an ORACLE DBMS has an associated redo log to protect the database in case of failures. REDO log files are filled with redo records. A redo record is made up of a group of change vectors, each of which is a description of a change made to a single block in the database. REDO logs record data that we can use to reconstruct all changes made to the database, including the undo segments. Therefore, the REDO log also protects rollback data. When you recover the database using redo data, the database reads the change vectors in the redo records and applies the changes to the relevant blocks. Redo records can also be written to a REDO log file before the corresponding transaction is committed. If the redo log buffer fills, or another transaction commits, LGWR flushes all of the redo log entries in the redo log buffer to a redo log file, even though some redo records may not be committed. If necessary, the database can roll back these changes. The database requires a minimum of two REDO log files to guarantee that one is always available for writing while the other is being archived.

REDO log files that are required for instance recovery are called active redo log files, the opposite is called inactive redo log files. If we have enabled archiving for recovery, then the DB cannot reuse or overwrite an active redo log file until its contents have archived. So if you proceed with BATCH jobs for OLTP concentrated time, it becomes the cause of performance degradation.

When the database archives redo log files, the archived log retains its log sequence number which is assigned to each redo log file by ORACLE. Each online or archived redo log file is uniquely identified by its log sequence number. During crash, instance, or media recovery, the database properly applies redo log files in ascending order by using the log sequence number of the necessary archived and redo log files (shown in Figure 2).

![Figure 1. REDO and ARCHIVE of ORACLE](image)

3 Our Approach

This chapter describes how to configure the description of the environment for OLTP and batch operations and explains the reasons of
performance improvements in the case of using SSD.

3.1 OLTP and BATCH

Typically online processing requires fast response times, and with the growth in computing power, companies have been able to put more emphasis on online transaction processing. In order to implement the OLTP environments for experiments, we ran a background program to generate infinitely processes that perform simple queries consisting of SELECT, INSERT and COMMIT after connecting to ORACLE DBMS, then end connection.

In addition, batch processing typically involves the transfer of some hardcopy record into a computer readable record, and putting a number of these records into a "BATCH". These "BATCHES" of records were supplied into the nightly process which read them in one at a time and performed the appropriate processing against jobs of the company. So the batch programs had to contain edits for keying errors, bad information, or other types of problems. In order to implement the batch environments for experiments, SQL loader of ORACLE was used to upload large amounts of data. Table for the experiment was composed of a range partition table, but the data of 48,000 tuples was uploaded in the partition of table that is accessed most frequently OLTP process using SQL loader.

For the objectivity of each time of experiment, partition tables were truncated to reset the OLTP and batch environments. For keeping the same experiment environment, REDO log was switched and ARCHIVE log files in the ARCHIVE log directory was moved before each experiment.

3.2 Analysis of performance improvement on SSD

Features of the ARCHIVE logs are always "write only" except in DBMS recovery. And features of the REDO logs are always "sequential write" and "a full deletion once" except in DBMS recovery. At time of DBMS recovery, features of REDO logs and ARCHIVE logs are "sequential read". When batch jobs of a large amount of data are conducted writing consistently occurs in REDO and ARCHIVE logs, therefore performance improvement can be achieved by utilizing the characteristics of SSD.

1. The SSD has low access latency than the HDD. Accordingly, the jobs generating continuous write/delete of REDO log and new/write of ARCHIVE log are carried out efficiently when going on the SSD. However, many REDO log deletion may result in system performance degradation.

2. The SSD are faster than the HDD to access random data. Actually, a flash memory has low reading and writing performance. So, multichannel architecture is implemented on the SSD that makes parallel reading from several flash memories. Because the SSD read data from separated flash memories, the SSD has higher performance on random reading than the HDD. So if we increase the number of ARCHIVE log files and reduce the size of ARCHIVE log files, the performance of the batch jobs will be greatly improved.

3. An IO/s of the SSD is much higher than the HDD. Compared to its random IO performance, the HDD has 120 IO/s of access speed whereas, the SSD has 11,200 IO/s access speed [8]. This is tremendous difference about 100-fold speed. As a result, if we take advantage of this feature it is possible to reduce the DBMS recovery time.

4 Experiment and Results

In the experiment, we used ORACLE enterprise DBMS 11GR2 and ran it on a Windows 7 Enterprise K 64 bit with a 3.30 GHz Intel i5-2500K processor and 4 GB RAM. This system was equipped with a magnetic disk drive and a flash memory SSD drive. The magnetic disk drive model was Western Digital with 1TB
capacity, 7200rpm, 64MB cache and SATA interface. The flash memory SSD model was Samsung Standard Type 830Series with 128 GB capacity and 1.8 inch PATA interface. These storage devices were connected to the computer systems via a SATA or PATA interface.

To configure an environment similar to the real mass DBMS, the main table was to create a partition table with monthly basis, partition key of the table is the date. All indexes and sub-partitions were made up of separate datafiles and tablespaces. The file size of REDO log was 5MB for frequent log switch and many ARCHIVE log. For measuring the performance of large data upload batch, SQL loader of ORACLE was used, performance was determined by taken time in the loader log.

In the first experiment, 3 REDO log files were consisted and in each of 4 cases to put ARCHIVE log and REDO log to the HDD and the SSD measured the execution time of a batch job during 20 times. As a result, when all of the REDO log and ARCHIVE log was moved to SSD and only REDO log was moved to the SSD, approximately 10% performance improvement was achieved. When only ARCHIVE log was moved to the SSD with the REDO log on the HDD, average performance improvement of 27% was achieved (shown in Figure 2).

The experimental results show that a lot of performance improvement will be achieved when number of REDO log, REDO log switching time and the size of ARCHIVE log are determined appropriately with exact consideration of the characteristics of batch job DATA and the SSD.

In the second experiment, the number of REDO log files has increased to six for plenty of times to write the REDO log in ARCHIVE log. The first and the second experiment result are not much difference on the HDD. However, when only REDO log was moved to the SSD, average performance improvement of 34% was made, when all of the REDO log and ARCHIVE log was moved to the SSD, average performance improvement was led to 40%. Also when only ARCHIVE log was moved to the SSD with the REDO log on the HDD, performance improvement of up to 290% was achieved and average performance improvement was led to 80% (shown in Figure 3).

The experimental results show that a lot of performance improvement will be achieved when number of REDO log, REDO log switching time and the size of ARCHIVE log are determined appropriately with exact consideration of the characteristics of batch job DATA and the SSD.

In the final experiment, we were measured restart time after force abort of DBMS during the second experiment. Depending on the location of the REDO log and ARCHIVE log DBMS recovery time was also some difference. In this case, when REDO log was moved to the SSD, recovery time of DBMS was shortened by
about 7% (shown in Table 3). This is considered to be due to features that the data required for recovery is mainly in the REDO log and the SSD has extremely fast read rate.

Table 3. Average Recovery time (6 REDO log, 20 times)

<table>
<thead>
<tr>
<th>Recovery Time (Sec)</th>
<th>Redo Log</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>HDD</td>
</tr>
<tr>
<td>Archive Log</td>
<td>18.25</td>
</tr>
<tr>
<td></td>
<td>17.40</td>
</tr>
</tbody>
</table>

However, if you decrease the REDO log switch time to improve performance, the number of ARCHIVE log files may increase, then, it may be occurred issues of the storage size and backup time. However, the experiment results showed that number of files is increased from 1,390 to approximately twice but the capacity of files is only increased about 20% from 1.9GB to 2.3GB. Furthermore, ARCHIVE log files should not be significantly considered a burden as they are just a simple backup for recovery.

5 Conclusion

If the SSD is used to the storage of systems using a large DBMS, performance will become excellent in most of cases. But it is more effective to utilize the SSD as appropriate because of the large price gap between the HDD and the SDD. As shown in the experiment result, when ARCHIVE log files are on the SSD and REDO log files on the HDD, it is most efficient from the aspect of database performance and the cost for building and maintaining a database. In this paper, we are sure that companies are able to improve the performance of the DBMS performing bulk data upload batch operations in utilizing a small amount of flash memory storage devices. In the future, if hot-date areas of the DBMS are on the SSD, example table or datafile, performance of BATCH, as well as OLTP seems to be much more improved.

REFERENCES


