ORACLE

Oracle Database Smart Flash Cache with Oracle Linux

Technical Paper

September, 2021

Introduction

This document explains how to increase Oracle database performance and improve user response time by deploying Oracle's Database Smart Flash Cache feature with Oracle Linux. Oracle's Database Smart Flash Cache feature is supported only on Oracle Linux and Oracle Solaris operating systems.

For transaction-based workloads, Oracle database blocks are normally loaded into a dedicated shared memory area in main memory called the System Global Area (SGA). Database Smart Flash Cache allows the database buffer cache to be expanded beyond the SGA in main memory to a second level cache on flash memory.

Online Transaction Processing (OLTP) and Datawarehouse (DWH) environments are expected to benefit from this technology. It is appropriate for new Oracle Database deployments with IO-intensive workloads as well as existing environments with main memory constraints and IO-bound applications. Benefits can include both increased transaction throughput and improved application response times. It is also supported in Real Application Cluster (RAC) environments – Database Smart Flash Cache can be applied to individual Oracle RAC nodes as required.

Oracle Database environments with the potential to make effective use of this technology include: workloads with repeated short transactions where many users access the same data, storage systems that exhibit intensive disk read activity, and systems under heavy main memory pressure that prevents more memory being allocated to the SGA buffer cache.

Oracle's Database Smart Flash Cache

Over many years the performance of spinning disk-based storage devices has not improved significantly. That situation has changed dramatically thanks to flash-based storage devices. Oracle Database Smart Flash Cache feature leverages this I/O breakthrough offered by flash-based storage devices.

Oracle's Database Smart Flash Cache functions as a secondary cache, which means that it stores clean (unmodified) database blocks that have been evicted from the SGA buffer cache to make room for other blocks. If a block selected for eviction has been modified, the dirty block is first written to disk by one of the database writer (DBWR) processes, then it is written to the Database Smart Flash Cache. Blocks can later be retrieved from the Database Smart Flash Cache and returned to the SGA as required. If a block cannot be found either in the SGA buffer cache or the Database Smart Flash Cache, it will be retrieved from disk.

The lifecycle of a block looks something like this:

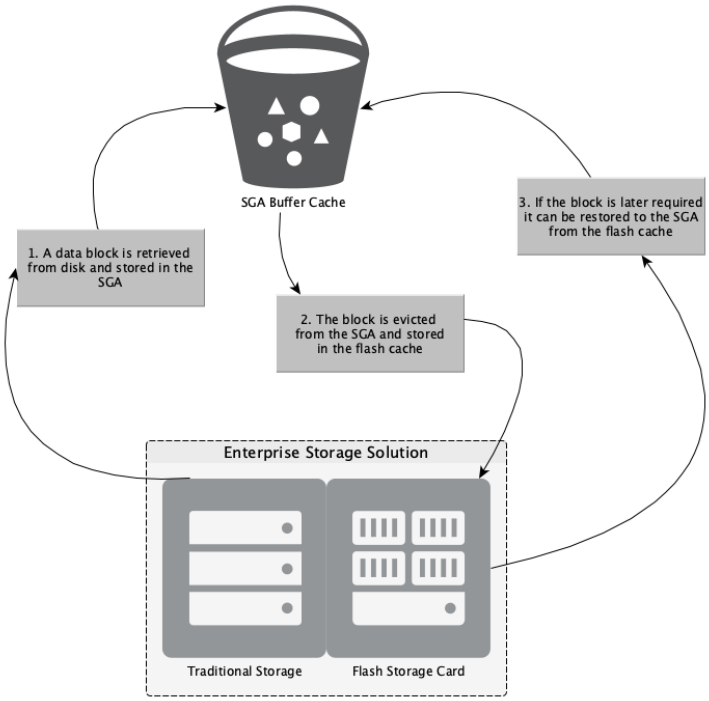

Figure 1. Lifecyle of a data block with Database Smart Flash Cache

When should you consider implementing Smart Flash Cache?

Smart Flash Cache is a suitable solution if/when:

- Oracle Database is suffering bandwidth saturation during high peak period
- Oracle Database needs to increase the number of transactions per second
- You want to **extend the life** of obsolete servers
- Looking for a cost-effective solution in order to optimize your database TCO
- Database is running on Oracle Linux with UEK or Solaris OS
- The *Buffer Pool Advisory* section of your *Automatic Workload Repository (AWR)* report indicates that doubling the size of the buffer cache would be beneficial
- "*db file sequential read*" is a top wait event

Oracle recommends a flash cache size of 4 to 10 times the Oracle Database System Global Area (SGA) size.

Some of the side benefits are:

- Improving physical write performance with less latency to the disk
- increasing server efficiency due to reduced storage I/O
- Saving energy consumption

Implementation

The Database Smart Flash Cache feature introduces two new init.ora parameters:

- DB_FLASH_CACHE_FILE, which identifies the flash device
- DB_FLASH_CACHE_SIZE, which specifies the amount of flash storage available to the Database Smart Flash Cache

The DB_FLASH_CACHE_FILE parameter can point to a raw device (including a logical volume), a file, or an ASM disk group. Note that for raw devices, the partition being used should start at cylinder 1 rather than cylinder 0 (to avoid the disk's volume label).

Here are some examples of these variables in use in an init.ora file:

Identifying the file location of the Database Smart Flash Cache (which could also be a symlink to a raw device):

```
db_flash_cache_file = "/export/home/oracle/lffile_raw"
```

Indicating that the Database Smart Flash Cache is managed by an ASM instance:

```
db_flash_cache_file = "+DG1/asm_llfile"
```

Specifying the size of the Database Smart Flash Cache (in this case 50 Gbytes):

```
db_flash_cache_file_size= 50G
```

A typical sizing of the Database Smart Flash Cache is two to ten times the size of SGA memory buffers. Note that header information is stored in the SGA for each flash cache buffer - 100 bytes per buffer in exclusive mode, 200 bytes per buffer in Oracle RAC mode - so the number of available SGA buffers is reduced as the Database Smart Flash Cache size increases, and the SGA size should be increased accordingly.

The implementation of Database Smart Flash Cache currently only allows a single flash device to be specified. For this reason some form of volume management is necessary when using Flash storage arrays; Oracle Database Automatic Storage Management (ASM) demonstrated to be a better choice than a traditional volume manager.

Note that the Database Smart Flash Cache is not shared across Oracle RAC nodes; it is private to each Oracle RAC instance. For that reason, each Oracle RAC instance must define its own local flash cache file path. When the flash cache is managed by ASM, a separate disk-group is needed for the flash cache of each instance.

Test Results

To characterize the Database Smart Flash Cache feature, a number of tests were run using a synthetic Online Transaction Processing (OLTP) simulated workload where multiple concurrent DB users submitted transactions against a database. Under heavy load, the large number of concurrent transactions generated a significant number of Input/Output operations (I/Os) to the storage sub-system. The workload included only read-only transactions and therefore only issued reads to the database.

The graph in Figure 2 below shows test results when the workload was disk-bound (meaning that application throughput was limited by the number of I/Os that could be processed by the disk subsystem). As Database Smart Flash Cache was introduced and increased, throughput increased steadily up to a point where it flattened off.

In the test environment, 8 gigabytes (GB) had been set aside for the SGA buffer cache, and the dataset size totaled 72 GB. Consequently, approximately 65 GB of flash cache were needed to cache the data entirely (assuming that about 1 GB of the SGA buffer cache was consumed by flash cache headers, and that 7 GB were therefore available in the SGA buffer cache for data blocks).

The graph shows that physical disk reads gradually diminished as the Database Smart Flash Cache size increased. Flash writes (which are needed to populate the Database Smart Flash Cache buffers) also increase with more flash - with 50 GB of flash storage, the majority of heavily-used buffers are already located in either the SGA or the Database Smart Flash Cache.

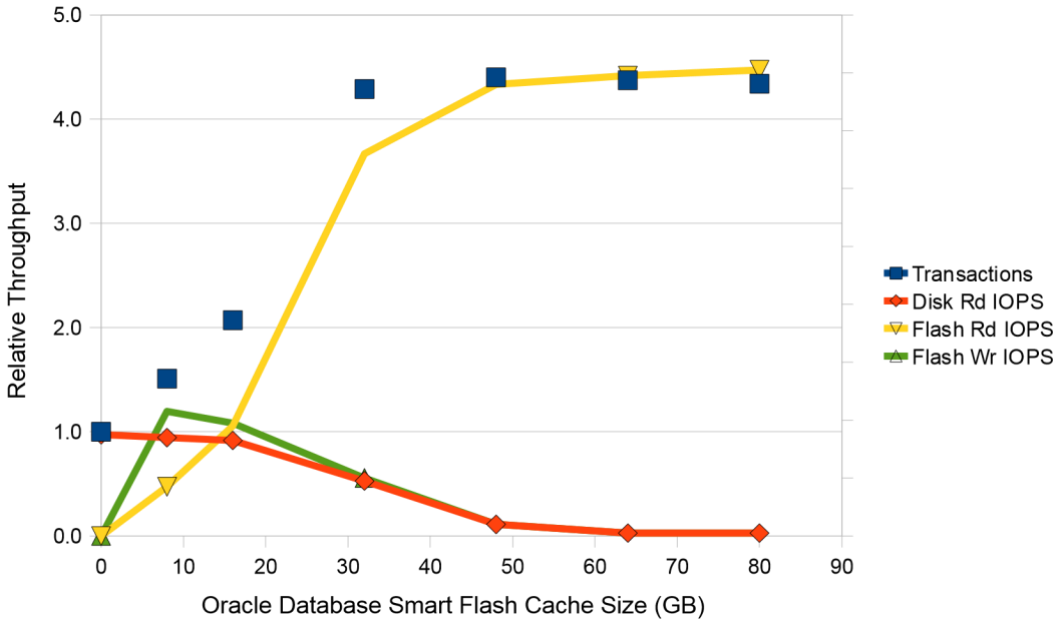

Figure 2: Relationship between Throughput and Disk and Flash I/Os

The final graph in Figure 3 below shows the transaction response times associated with the benchmark tests described above. In each case, the use of Database Smart Flash Cache resulted in a significant response time improvement. Where the workload was significantly constrained by disk I/O and adequate Database Smart Flash Cache was assigned, the gains were very significant with response times improving almost 5 times (x5). These results demonstrate that the vastly superior read and write performance can lead directly to improved transaction response times for disk-constrained applications.

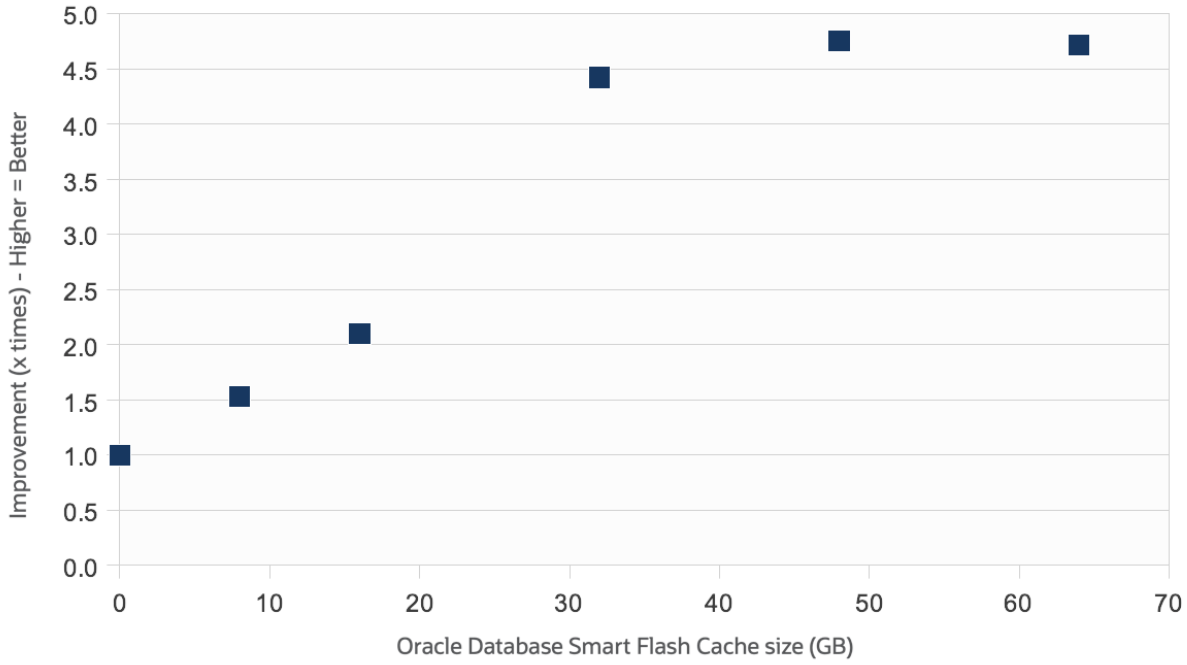

Figure 3: Response Time Improvement with Increasing Database Smart Flash Cache Size

Higher is the size of the Oracle Database Smart Flash Cache and higher is the possibility that blocks are identified into the same (improved performance).

Note that benefits may vary for write-intensive workloads, and improvements are likely to be less significant.

Conclusion

Oracle Database Smart Flash Cache accelerates applications, increases productivity and improves business responsiveness. Based upon the benchmarks that were executed running on Oracle Linux with Unbreakable Enterprise Kernel, large performance gains were realized.
As a side benefit, by implementing Oracle's Database Smart Flash Cache feature, not only will this combination reduce the hard disk IOPS for reads resulting in large performance gains, this reduction in IOPS for reads will result in the capability to perform physical writes with less latency.

Administrators can influence caching priorities by forcing the cache to keep entire tablespace.
SFC can **highly improve the response time** and **overall throughput** for both read-intensive online transaction processing (**OLTP**) workloads, ad hoc queries and bulk data modifications in a data warehouse (**DW**), environment; these are significant benefits for customers running large databases.

Oracle's Database Smart Flash Cache feature with Oracle Linux and Unbreakable Enterprise Kernel provides a platform that can scale and perform to the demanding needs of growing enterprises.

Connect with us

Call **+1.800.ORACLE1** or visit **oracle.com**. Outside North America, find your local office at: **oracle.com/contact**.

blogs.oracle.com facebook.com/oracle twitter.com/oracle