# Firebird Database Cache Buffer

Norman Dunbar

21 June 2010 – Document version 1.1

# Table of Contents

# Introduction

Firebird uses a page cache to hold pages in memory. It is much quicker to retrieve pages from RAM than to go out onto the disc system and read them physically each time they are required. The following description of how Firebird uses its cache is taken from a posting by Ann Harrison on the Firebird Support mailing list.

The posting was in response to a query asking *if there is a way to reduce the memory cache without giving up too much performance?* This was itself in respect of a system that was taking a long time to wake up from hibernation and the cause was thought to be the time required to load back all the cache pages from disc before the first query (after hibernation) could be processed. The cause was confirmed and the DBA responsible for the database asked, on the list, how he could reduce the buffer cache as much as possible but still have a responsive system.

Ann has given her permission for this posting to be documented formally as part of the Firebird documentation project.

# The Firebird Cache

Everything in the database is organised as fixed sized pages of known structure - there are nine different types of pages. The page cache is the intermediate between the "working" parts of the database and the disk.

When initially started, Firebird reads the database header page, then the first pointer page of a system table called RDB$PAGES from which it then knows where to find the pointer pages, amongst other things, for the system and user tables within the database.

As soon as the *application* accesses the database and begins a transaction, Firebird reads the pointer pages telling it where to find the data pages for the tables involved in the application's transactions. All the pages go into the cache and stay there until the cache is completely full. When there's no place to put the next page, Firebird releases the *least recently used* page - *not* the first one read, but the one that has been referenced least recently and has not been modified.

When a transaction commits - and at a few other times - Firebird writes the pages changed by that transaction to disk, but does not release them from the cache - this helps avoid an extra read if the page is required again soon.

Over time and with luck, you end up with the most frequently changed and referenced pages resident in the cache - those would be transaction inventory pages, pointer pages for active tables, the header page, the top levels of indexes, etc. Data and lower level index pages swap in and out as required - but the cache will always be full.

You can use the MON$ tables (in Firebird 2.1 onwards, especially MON$IO_STATS) to figure out how well your cache is being used. They will report on the number of Fetches vs. Reads which is the number of times pages were accessed versus the number of times they had to be read from disk. When the number of reads goes up dramatically, you've reduced the cache too much.

The MON$ tables will also give you the number of Marks vs. Writes, which tells you the number of times pages are changed versus the number of times the pages were written to disk. When you see the disk writes go up, you've probably reduced the cache too much.

# Using MON$IO_STATS

As mentioned above, the table MON$IO_STATS can be used to help you determine how well your buffer cache is performing. The table has the following structure:

**MONS$STAT_ID:** The statistic id.

**MONS$STAT_GROUP:** The statistic group. Statistics are gathered for the following groups:

- 0: The database as a whole.
- 1: Attachments.
- 2: Transactions.
- 3: Statements.
- 4: Calls.

**MON$PAGE_READS:** The number of pages read. These are the pages read from the disc and not from the cache.

**MON$PAGE_WRITES:** The number of pages written back to disc.

**MON$PAGE_FETCHES:** The number of pages read from the cache as opposed to from disc.

**MON$PAGE_MARKS:** The number of pages changed while in the cache. It is possible that not all of these have been written back to disc.

To inspect the current statistics for the database as a whole, we would use the following query in isql:

```
tux> isql employee
Database:  employee

SQL> SELECT MON$PAGE_READS, MON$PAGE_WRITES, MON$PAGE_FETCHES, MON$PAGE_MARKS
CON> FROM MON$IO_STATS
CON> WHERE MON$STAT_GROUP = 0;

     MON$PAGE_READS     MON$PAGE_WRITES    MON$PAGE_FETCHES      MON$PAGE_MARKS
========== ========== ========== ==========
               134                526              13851                529
```

The results of the above show that:

- 134 pages have had to be physically read from the disc into the cache so far.
- 13,851 pages, on the other hand, have been read directly from the cache.
- 529 pages, in the cache, have been changed in some way.
- 526 changed pages have been copied from the cache to disc.

We can assume, therefore, that although a small number of pages have been read into the cache, there is nothing we can do to avoid that. When the database is started up the cache is empty, when applications connect and access the database, various pages must be read and the cache must be filled, so physical reads will be a necessity. In this example, it appears that once pages are in the cache they are being accessed quite frequently given that there have been approximately 103 cache reads for every physical read.

Of the 529 updated pages - and these are system as well as user pages - 526 have been written back to the physical discs but three still remain in cache, as yet, unwritten.

The results shown above show the performance of the cache over the life of the database so far. We can narrow this down to our current attachments by modifying the query to select those rows where the MON$STAT_GROUP is 1.

```
SQL> SELECT MON$PAGE_READS, MON$PAGE_WRITES, MON$PAGE_FETCHES, MON$PAGE_MARKS
CON> FROM MON$IO_STATS
CON> WHERE MON$STAT_GROUP = 1;

     MON$PAGE_READS     MON$PAGE_WRITES    MON$PAGE_FETCHES      MON$PAGE_MARKS
========== ========== ========== ==========
                 0                  4                 87                  5
               134                520              13619                522
```

Interpretation of the above statistics is exactly the same as for the database as a whole.

We can further diagnose the statistics by individual transactions, as follows:

```
SQL> SELECT MON$PAGE_READS, MON$PAGE_WRITES, MON$PAGE_FETCHES, MON$PAGE_MARKS
CON> FROM MON$IO_STATS
CON> WHERE MON$STAT_GROUP = 2;

     MON$PAGE_READS     MON$PAGE_WRITES    MON$PAGE_FETCHES      MON$PAGE_MARKS
========== ========== ========== ==========
                 0                  0                 60                  0
                 0                  0                  1                  0
                 0                  0                  1                  0
                 0                  0                 69                  0
                 0                  0                 93                  0
                 0                  0                 85                  0
                 0                  0                  1                  0
                 0                  0                  1                  0
```

And, by individual statements:

```
SQL> SELECT MON$PAGE_READS, MON$PAGE_WRITES, MON$PAGE_FETCHES, MON$PAGE_MARKS
CON> FROM MON$IO_STATS
CON> WHERE MON$STAT_GROUP = 3;

     MON$PAGE_READS     MON$PAGE_WRITES    MON$PAGE_FETCHES      MON$PAGE_MARKS
========== ========== ========== ==========
                 0                  0                  1                  0
                 0                  0                 38                  0
                 0                  0                  4                  0
                 0                  0                 18                  0
                 0                  0                158                  0
                 0                  0                  1                  0
                 0                  0                  1                  0
                 0                  0                  1                  0
                 0                  0                  1                  0
                 0                  0                  1                  0
                 0                  0                  0                  0
                 0                  0                  1                  0
                 1                  0                 12                  0
                 0                  0                  2                  0
                 3                  0               1436                  0
```

```
                    0                    0                  101                    0
                    7                    0                  613                    0
```

Finally, it is possible - and probably most useful - to determine the statistics for your own session. You can find your attachment id from CURRENT_CONNECTION and use that in a query that joins with MON$IO_STATS using the MON$STAT_ID column.

```
SQL> SET LIST;

SQL> SELECT T.MON$ATTACHMENT_ID, T.MON$TRANSACTION_ID,
CON> IO.MON$PAGE_READS, IO.MON$PAGE_WRITES,
CON> IO.MON$PAGE_FETCHES, IO.MON$PAGE_MARKS
CON> FROM MON$TRANSACTIONS AS T
CON> JOIN MON$IO_STATS as IO
CON> ON (IO.MON$STAT_ID = T.MON$STAT_ID)
CON> WHERE T.MON$ATTACHMENT_ID = CURRENT_CONNECTION;

MON$ATTACHMENT_ID               12
MON$TRANSACTION_ID              218
MON$PAGE_READS                  5
MON$PAGE_WRITES                 0
MON$PAGE_FETCHES                66
MON$PAGE_MARKS                  0

MON$ATTACHMENT_ID               12
MON$TRANSACTION_ID              217
MON$PAGE_READS                  0
MON$PAGE_WRITES                 0
MON$PAGE_FETCHES                1
MON$PAGE_MARKS                  0
```

# Appendix A: Document history

The exact file history is recorded in the `manual` module in our CVS tree; see http://sourceforge.net/cvs/?group_id=9028 . The full URL of the CVS log for this file can be found at http://firebird.cvs.sourceforge.net/viewvc/firebird/ manual/src/docs/firebirddocs/fbcache.xml?view=log

**Revision History**

| | | | |
|---|---|---|---|
| 1.0 | 05 January 2010 | ND | Created a new manual based on a posting to Firebird-support by Ann Harrison. |
| 1.1 | 21 June 2010 | ND | Amended to include the fact that it *is* possible to extract the statistics for the current connection. Contrary to what was said before. |

# Appendix B: License notice