



Firebird Database Cache Buffer

Norman Dunbar

20. Juli 2013 – Version 1.3.1-de - deutsche Version

Übersetzung ins Deutsche: Martin Köditz

Inhaltsverzeichnis

Einleitung	3
Der Firebird-Cache	3
MON\$IO_STATS verwenden	4
Anhang A: Dokumenthistorie	8
Anhang B: Lizenzhinweis	9

Einleitung

Firebird nutzt einen Seiten-Cache (engl. page cache), um die Seiten im Speicher vorzuhalten. Diese sind deutlich schneller aus dem RAM zu laden als jedesmal von der Festplatte zu lesen. Die folgende Beschreibung wie Firebird den eigenen Cache verwendet, stammt aus einem Posting von Ann Harrison aus der Firebird-Support-Mailing-List.

Das Posting war eine Antwort auf die Frage ob es einen Weg gibt, den Speicher-Cache zu reduzieren, ohne zuviel Performance aufzugeben (engl. *if there is a way to reduce the memory cache without giving up too much performance?*) Dieses Posting bezog sich auf ein System, das sehr lange benötigte, um aus dem Schlafmodus zu erwachen. Dies wurde als Grund vermutet, dass einige Zeit benötigt wurde, um alle Cache-Seiten vor dem Ausführen der ersten Abfrage von der Festplatte zu lesen. Dieser Grund wurde bestätigt und der verantwortliche DBA fragte über die Liste, wie er den Puffer soweit wie möglich reduzieren könne, zeitgleich aber noch ein vernünftig laufendes System erhalte.

Ann gab ihr Einverständnis das Posting formal als Teil des Firebird-Dokumentations-Projektes aufzunehmen.

Der Firebird-Cache

Alles in der Datenbank ist in Seiten mit fester Größe vorgegebener Struktur abgelegt - es gibt neun verschiedene Seitentypen. Der Seiten-Cache ist ein Zwischenzustand zwischen dem "Arbeits"-Bereich und der Festplattenspeicher.

Beim Start liest Firebird die Datenbank-Header-Seite (engl. database header page), dann die erste Zeiger-Seite (pointer page) der Systemtabelle RDB\$PAGES. Hiermit erfährt Firebird, wo die Zeiger-Seiten, neben anderen Dingen, für die System- und Benutzertabellen zu finden sind.

Sobald die *Anwendung* auf die Datenbank zugreift, beginnt eine Transaktion. Firebird liest die Zeiger-Seiten, die erzählen wo die Daten-Seiten für die innerhalb der Transaktion betroffenen Tabellen zu finden sind. Alle Seiten wandern in den Cache und bleiben dort bis der Cache voll ist. Wenn kein Platz für eine neue Seite vorhanden ist, wird Firebird die am wenigsten genutzte Seite entfernt - *nicht* die erste gelesene, sondern die wenigst genutzte, die nicht geändert wurde.

Wenn ein Commit für eine Transaktion durchgeführt wird - und bei einigen anderen Dingen - schreibt Firebird die durch diese Transaktion geänderten Seiten auf die Festplatte, entfernt sie aber nicht aus dem Cache - dies verhindert, dass die Seiten nochmals eingelesen werden müssen, wenn sie baldmöglichst benötigt werden.

Mit der Zeit und etwas Glück, werden zum Schluss die am häufigsten geänderten und genutzten Seiten im Cache verbleiben - das wären Transaktions-Inventar-Seiten, Zeiger-Seiten für aktive Tabellen, die Header-Seite, die Top-Level-Indexes, etc.. Daten und Lower-Level-Index-Seiten werden rein- und rausgeladen, je nach Gebrauch - aber der Cache wird immer gefüllt sein.

Sie können die MON\$-Tabellen (seit Firebird 2.1, insbesondere MON\$IO_STATS) nutzen, um herauszufinden, wie gut Ihr Cache ausgelastet ist. Sie zeigen die Anzahl der Fetches gegenüber den Reads, wie oft auf Seiten zugegriffen wurde und wie oft diese von der Festplatte gelesen wurden. Wenn die Anzahl der Reads dramatisch in die Höhe geht, haben Sie den Cache zu sehr reduziert.

Die MON\$-Tabellen zeigen Ihnen außerdem die Anzahl der Marks gegen die Anzahl der Writes, welche angeben, wie oft Seiten geändert wurden gegenüber der Anzahl von Seiten, die auf die Festplatte geschrieben wurden. Wenn Sie sehen, dass die Writes in die Höhe schnellen, haben Sie vermutlich den Cache zu sehr reduziert.

MON\$IO_STATS verwenden

Wie oben bereits angegeben, kann die Tabelle MON\$IO_STATS genutzt werden, um zu bestimmen, wie gut Ihr Buffer-Cache arbeitet. Die Tabelle hat folgende Struktur:

MONS\$STAT_ID: Die Statistik-ID.

MONS\$STAT_GROUP: Die statistische Gruppe. Statistiken werden in folgende Gruppen unterteilt:

- 0: Die Datenbank als Ganzes.
- 1: Attachments.
- 2: Transaktionen.
- 3: Statements.
- 4: Aufrufe (Calls).

MON\$PAGE_READS: Die Anzahl der gelesenen Seiten. Dies sind die Seiten, die von der Festplatte und nicht aus dem Arbeitsspeicher gelesen wurden.

MON\$PAGE_WRITES: Die Anzahl der auf die Festplatte zurückgeschriebenen Seiten.

MON\$PAGE_FETCHES: Die Anzahl der Seiten, die aus dem Cache gelesen wurden, Gegensatz zu denen von der Festplatte.

MON\$PAGE_MARKS: Die Anzahl der im Cache geänderten Seiten. Es ist möglich, dass nicht alle zurück auf die Festplatte geschrieben wurden.

Um die derzeitigen Statistiken für die gesamte Datenbank zu untersuchen, verwenden wir folgende Abfrage 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
```

Das erzeugte Resultat zeigt, dass:

- 134 Seiten von der Festplatte in den Cache gelesen werden mussten.
- 13.851 Seiten, auf der anderen Seite, wurden direkt aus dem Cache gelesen.
- 529 Seiten im Cache, wurden auf irgendeinem Wege geändert.
- 526 geänderte Seiten wurden vom Cache auf die Festplatte kopiert.

Hieraus können wir erkennen, dass obwohl eine kleine Anzahl Seiten in den Cache gelesen wurde, wir nichts tun können, um dies zu vermeiden. Sobald die Datenbank gestartet wird, ist der Cache leer, wenn aber eine Anwendung eine Verbindung herstellt, werden verschiedene Seiten gelesen und in der Cache gefüllt, wodurch physikalische Leseoperationen durchgeführt werden. In diesem Beispiel scheint es so zu sein, dass sobald Seiten im Cache vorhanden sind, diese auch relativ häufig verwendet werden. Für jeden physikalischen Read wurden ungefähr 103 Reads aus dem Cache durchgeführt.

Von den 529 aktualisierten Seiten - und dies sind sowohl System- wie auch Benutzerseiten - wurden 526 zurück auf die Festplatte geschrieben, während 3 Seiten immer noch im Cache verbleiben, noch nicht geschrieben.

Das obige Beispiel zeigt die Performance des Caches während der Datenbanklaufzeit. Wir können dies auf unsere aktuelles Attachment herunterbrechen, indem wir die Abfrage etwas verändern. Wir holen nur die Datensätze, wo MON\$STAT_GROUP gleich 1 ist.

```
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

Die Interpretation der neuen Statistiken ist die gleiche wie für die gesamte Datenbank.

Wir können außerdem eine Statistikdiagnose für Transaktionen durchführen:

```
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

Und auch für 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

Schlussendlich ist es möglich - und vermutlich äußerst nützlich - die Statistiken für Ihre eigene Sitzung zu bestimmen. Sie finden Ihre Attachment-ID mittels CURRENT_CONNECTION unter der Verwendung eines Joins mit MON\$IO_STATS und der Spalte MON\$STAT_ID.

```
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

Anhang A: Dokumenthistorie

Die exakte Dateihistorie ist im manual-Module unseres CVS-Baumes aufgenommen; Siehe http://sourceforge.net/cvs/?group_id=9028 Die vollständige URL des CVS-Logs dieser Datei kann unter dem Link <http://firebird.cvs.sourceforge.net/viewvc/firebird/manual/src/docs/firebirddocs/fbcache.xml?view=log> gefunden werden.

Versionsgeschichte

1.0	05. Januar 2010	ND	Neues Dokument auf Basis eines Postings von Ann Harrison an den Firebird-support.
1.1	21. Juni 2010	ND	Ergänzt, dass es möglich <i>ist</i> , die Statistiken für die aktuelle Verbindung zu erhalten. Im Gegensatz zur vorigen Behauptung.
1.1-de	02. September 2013	MK	Deutsche Übersetzung.

Anhang B: Lizenzhinweis

Der Inhalt dieser Dokumentation unterliegt der "Public Documentation License Version 1.0" (der „Lizenz“); die Dokumentation darf nur unter Respektierung dieser Lizenz genutzt werden. Kopien der Lizenz sind verfügbar unter <http://www.firebirdsql.org/pdfmanual/pdl.pdf> (PDF) und <http://www.firebirdsql.org/manual/pdl.html> (HTML).

Die Original-Dokumentation trägt den Titel *Firebird Database Cache Buffer*.

Der ursprünglich Autor der Original-Dokumentation ist: Norman Dunbar unter Verwendung der durch Ann Harrison zur Verfügung gestellten Daten.

Copyright (C) 2010. Alle Rechte vorbehalten. Kontakt zum Original-Autor: NormanDunbar at users dot sourceforge dot net.