УДК 681.3.07

ИСПОЛЬЗОВАНИЕ ТИПОВ ХРАНИЛИЩ MYSQL: MYISAM И INNODB

Т.Т.КАРИМБАЕВ, А.М.МАМБЕТАЛИЕВА

E.mail. ksucta@elcat.kg

Бул макалада MySQLде: MyISAM жана InnoDB сактоо типтерин пайдалануу көрсөтүлгөн. Алардын негизги касиеттери, параметрлери, алар аткаруучу функциялар жана алардын ортосундагы айырмачылыктар чагылдырылган.

В статье описано использование типов хранилищ MySQL: MyISAM и InnoDB. Отражены основные их свойства, параметры, выполняемые ими функции и различия между ними.

This article describes how to use MySQL storage types such as MyISAM and InnoDB. Reflected the main properties, parameters, their functions and the differences between them.

MySQL ("Язык структурированных запросов") представляет собой:

- систему управления базами данных. База данных представляет собой структурированный набор данных;
- систему управления реляционными базами данных. Реляционная база данных хранит информацию в отдельных таблицах, а не в одном большом хранилище, благодаря чему достигается высокая производительность и гибкость;
- систему с открытым исходным кодом. Открытость исходного кода означает, что любой желающий имеет возможность использовать и модифицировать это программное обеспечение по своему усмотрению;
- сервер баз данных MySQL – очень быстрый, надежный и простой в эксплуатации сервер;
- сервер MySQL работает в клиент-серверных и встроенных системах. СУБД MySQL является клиент-серверной системой, включающей многопоточный SQL-сервер, поддерживающий различные платформы, несколько клиентских программ и библиотек, инструменты администрирования и широкий диапазон программных интерфейсов приложений (API-интерфейсов).
- сервер MySQL существует также и в форме встраиваемой многопоточной библиотеки, которую можно связывать с разрабатываемыми приложениями, чтобы получить более компактные, быстрые и легкоуправляемые продукты /1/.

База данных MySQL работает с несколькими видами хранилищ данных. Хранилища отличаются способом хранения данных, набором возможностей.

Хранилище MyISAM поддерживается сервером MySQL начиная с третьей версии. Оно организовано достаточно просто, тем самым с его помощью можно добиться очень высокой производительности, особенно при выборе таблиц из данных. С другой стороны, добавление данных в таблицу влечет за собой блокировку всей таблицы, поэтому одновременное добавление и выбор данных из одной и той же таблицы ведет к задержке выбора данных.

Хранилище InnoDB устроено гораздо более сложным образом. Производительность при выборе данных из таблиц в нем ниже, чем в MyISAM, зато таблицы не блокируются при изменении данных и одновременные добавления в таблицу не блокируют операции чтения из нее. Более того, InnoDB полностью поддерживает транзакции со всеми четырьмя уровнями изоляции и ссылочную целостность.

Таблицы MyISAM прекрасно подходят для использования в WWW и других средах, где преобладают запросы на чтение, а также показывают хорошие результаты при выборках в SELECT. Во многом это связано с отсутствием поддержки транзакций и внешних ключей. Однако при модификации и добавлении записей вся таблица кратковременно блокируется, это может привести к серьезным задержкам при большой загрузке. Для таблиц этого типа создан ряд специализированных утилит, позволяющих манипулировать табличными файлами. Это утилита myisamchk для проверки и восстановления таблиц и индексов, она требует полной остановки процесса MySQL и создает время неработоспособности системы, исполнение заключается в создании с нуля нового целостного файла таблицы и перезаписи данных в него. Также есть утилита myisampack для создания сжатых таблиц. Таблицы MyISAM являются платформенно-независимыми.

Новшества, которыми обладает тип MyISAM:

- Флаг в файле MyISAM, указывающий, правильно была закрыта таблица или нет. В случае запуска mysqld с параметром --myisam-recover таблицы MyISAM будут автоматически проверяться и/или восстанавливаться при открытии, если таблица была закрыта неправильно.
- При помощи команды INSERT можно вставлять новые строки в таблицу, в середине файла данных которой нет свободных блоков, в то время как другие потоки считывают из таблицы информацию (совмещенная вставка). Свободный блок может быть получен при обновлении строки с динамической длиной, когда большее количество данных заменяется меньшим количеством или при удалении строк. Когда свободных блоков не остается, все последующие блоки снова будут вставляться как совмещенные.
- Поддержка больших файлов (63 бита) в файловых/операционных системах, которые поддерживают большие файлы.
- Хранение всех данных осуществляется с первым младшим байтом. Это делает данные независимыми от операционной системы. Единственное требование – в компьютере должны применяться дополненные до двух байтов целые числа со знаком (как и во всех компьютерах в последние 20 лет) и формат с плавающей единичной запятой IEEE (также использующийся в подавляющем большинстве серийных компьютеров). Единственными компьютерами, которые могут не поддерживать бинарную совместимость, являются встроенные системы (поскольку в них иногда применяются специальные процессоры). При хранении данных с первым младшим байтом не происходит снижения скорости. Обычно байты в строке таблицы не выровнены и нет большой разницы в том, как прочитать невыровненный байт – в прямой последовательности или в обратной. Фактическое время извлечения значения столбца также не критично по сравнению со временем выполнения остального кода.
- Все ключи номеров хранятся с первым старшим байтом, чтобы сжатие индексов было более эффективным.
- Внутренняя обработка столбца AUTO_INCREMENT. MyISAM автоматически обновляет его при выполнении команд INSERT/UPDATE. Значение AUTO_INCREMENT может быть обнулено оператором myisamchk. После этого столбец AUTO_INCREMENT будет быстрее (по крайней мере на 10 %) и старые номера не будут повторно использоваться, как со старым ISAM. Обратите внимание: когда AUTO_INCREMENT задан в конце составного ключа, старое поведение все еще сохраняется.
- При вставке в отсортированном порядке (как при использовании столбца AUTO_INCREMENT) дерево ключей будет разделено таким образом, чтобы верхний узел содержал только один ключ. При этом сокращается расход пространства памяти в дереве ключей.
- Столбцы BLOB и TEXT могут быть проиндексированы.
- В индексных столбцах разрешены значения NULL. Они занимают 0-1 байта на ключ.

- По умолчанию максимальная длина ключа составляет 500 байтов (это значение может быть изменено при повторной компиляции). В случаях, когда ключи больше 250 байтов, для них используются большие размеры блока ключа, чем предусмотренные по умолчанию 1024 байта.
- По умолчанию в таблице может быть не более 32 ключей. Это значение можно увеличить до 64 без повторной компиляции myisamchk.
- myisamchk будет отмечать таблицы как проверенные, если они запускаются с параметром --update-state. myisamchk --fast будет проверять только те таблицы, в которых отсутствует данная пометка.
- myisamchk -a сохраняет статистические данные по частям ключа (не только для ключей целиком, как в ISAM).
- Строки с динамическим размером будут менее фрагментированными, чем при смешивании удалений с обновлениями и вставками. Это осуществляется путем автоматического сочетания удаленных смежных блоков и расширением блоков, если следующий блок удален.
- myisampack может упаковывать столбцы BLOB и VARCHAR.
- Можно поместить файл данных и файл индексов в разные каталоги, чтобы увеличить скорость (с параметром DATA/INDEX DIRECTORY="path" для CREATE TABLE) /7/.

MyISAM также поддерживает следующие функции, которые можно будет использовать в MySQL в ближайшем будущем:

- Поддержка типа VARCHAR; столбец VARCHAR начинается с длины, которая хранится в 2 байтах.
- Таблицы с VARCHAR могут иметь фиксированную или динамическую длину записей.
- VARCHAR и CHAR могут быть до 64 Кб длиной. У всех ключевых сегментов есть свои собственные определения языка. Это позволяет задавать в MySQL различные определения языка для каждого столбца.
- Для UNIQUE может использоваться вычисленный хэш-индекс. Это позволяет использовать UNIQUE с любым сочетанием столбцов в таблице (тем не менее, нельзя производить поиск по вычисленному UNIQUE индексу) /7/.

Если создаем базу данных по обстоятельствам и не уверены, как в дальнейшем база будет использоваться, нужно выбрать InnoDB. Его следует использовать, когда взаимодействие с базой данных имеет характер OLTP[1], когда нужна высокая надежность хранения и быстрое восстановление после сбоя /2/.

Преимуществом InnoDB является то, что хорошо справляется с нагрузкой (select/update/delete/insert).

Недостатком InnoDB является то, что могут возникать deadlock[2], не свойственные MyISAM. Также медленнее выполняется insert операции и работа с блогами, не поддерживается полнотекстовой поиск, иногда возникают проблемы с производительностью Count(*) и нет поддержки mysqlhotcopy (копирование баз данных и таблиц MySQL) /2/.

[1]OLTP(Online Transaction Processing),транзакционная система-обработка транзакций в реальном времени. Способ организации базы данных, при котором система работает с небольшими транзакциями.

[2] deadlock – ситуация в системе управления базами данных, при которой несколько процессов находятся в состоянии бесконечного ожидания ресурсов, захваченных самими этими процессами.

В MyISAM без особых причин таблица может отказаться работать до выполнения REPAIR TABLE. Для избежания этой проблемы рекомендуется организовать периодический запуск mysqlcheck (для сопровождения и аварийного восстановления таблиц) через cron.

InnoDB обладает следующими преимуществами перед MyISAM:

1. для программиста есть возможность объединить операции с базой в транзакцию;

2. в отличие от MyISAM, где блокировка идет на уровне таблицы, в InnoDB блокировка осуществляется на уровне строки;
3. InnoDB более устойчивая к сбоям, намного лучше восстанавливается после сбоев и практически не теряет данные;
4. качественная работа с Input/Output (IO). InnoDB имеет свой собственный Buffer Pool в памяти, где держит таблицы. Для InnoDB можно отключить системную буферизацию IO при работе с таблицами InnoDB, следовательно, оперативная память разумно расходуется. А в MyISAM –двойная буферизация /6/.

Очень важно понимать, как происходят процессы в InnoDB, чтобы правильно настроить этот тип хранилища и способствовать корректной работе на высоких нагрузках.

Конфигурации существенно влияющие на производительность:

innodb_file_per_table

По умолчанию, InnoDB использует общее хранилище для всех таблиц и индексов. Данная опция позволяет создавать на каждую таблицу свой .ibd файл. Часто применяется эта опция - когда необходимо раскидать отдельные таблицы по отдельным физическим устройствам. Когда часто дополняются данными или удаляются данные из таблицы, чтобы не пострадала производительность других таблиц, также применяют разбиение общего хранилища на отдельные части для каждой таблицы.

Если таблицы уже созданы в общем хранилище, то при перезагрузке сервера MySQL с этой опцией, автоматически старые таблицы останутся в общем хранилище, а новые будут создаваться в отдельных хранилищах. /6/

Определенной особенностью InnoDB является то, что можно использовать целые партиции[3] или физические устройства вместо файлов общего хранилища InnoDB. Это позволяет убирать системную буферизацию ввода-вывода и overhead файловой системы, и InnoDB пишет данные прямо на устройство.

Для того, чтобы использовать эту возможность, необходимо дополнить в конфигурации следующее:

```
[mysqld]
innodb_data_home_dir=innodb_data_file_path=/dev/hdd1:3Gnewraw; /dev/hdd2:2Gnewraw
```

После запуска InnoDB выполнит инициализацию блочных устройств. При этим необходимо остановить сервер и в конфигурации поменять «newraw» на «raw»:

```
[mysqld]
innodb_data_home_dir= innodb_data_file_path=/dev/hdd1:3Graw; /dev/hdd2:2Graw
```

и перезапустить сервер. Иначе при следующем перезапуске партиция будет снова отформатирована. Также важно, чтобы пользователь имел права на запись в обозначенные партиции. При использовании данной возможности необходимо выделять для InnoDB логические тома LVM[4]. Выделение LVM существенно упрощает backup и восстановление.

InnoDB предназначается для получения максимальной производительности при обработке больших объемов данных. По эффективности использования процессора этот тип намного превосходит другие модели реляционных баз данных с памятью на дисках.

---

[3]Партиции – части (разделы), на которые разбивается диск. Всего на диске могут быть 4 первичных раздела, один из которых может быть расширенным. Расширенный раздел включает в себя вторичные разделы.

[4]LVM (Logical Volume Manager)- метод распределения пространства жесткого диска по логическим томам, размер которых легко менять, в отличие от разделов.

Преобразование таблиц MyISAM в формат InnoDB

В InnoDB отсутствует специальная оптимизация создания отдельных индексов. Таким образом, этот формат не обеспечивает экспорта и импорта таблиц с последующим созданием индексов.

Самый быстрый способ преобразовать таблицу в формат InnoDB - напрямую вставить данные в таблицу InnoDB, воспользовавшись командой ALTER TABLE ... TYPE=INNODB, или создать пустую таблицу InnoDB с такой же структурой, и вставить строки при помощи команды INSERT INTO ... SELECT * FROM ....

Чтобы лучше контролировать процесс вставки, большие таблицы желательно вставлять по частям:

```
INSERT INTO newtable SELECT * FROM oldtable
WHERE yourkey > something AND yourkey <= somethingelse;
```

После того, как все данные будут вставлены, таблицы можно будет переименовать.

Во время преобразования больших таблиц необходимо задать достаточно большой размер динамического буфера InnoDB, чтобы снизить количество дисковых операций ввода/вывода. Однако размер буфера не должен превышать 80% физической памяти компьютера. Следует установить большие размеры для файлов журналов InnoDB, а также большой размер буфера журналов.

Убедитесь, что у вас достаточно свободного пространства для табличной области: таблицы InnoDB занимают намного больше места, чем таблицы MyISAM. Если во время выполнения команды ALTER TABLE будет исчерпано свободное дисковое пространство, начнется выполнение отката, и это может занять несколько часов, если диск заполнен. Во время вставок для таблицы InnoDB используется буфер вставки, чтобы произвести объединение вторичных индексных записей с индексными таблицами при помощи групповых операций. Это позволяет значительно снизить интенсивность дисковых операций ввода/вывода. При откате такой механизм не используется, поэтому откат может занять в 30 раз больше времени, чем вставка.

В случае, если началось выполнение отката и база данных не содержит ценной информации, лучше прервать этот процесс и удалить все данные InnoDB, файлы журналов, а также все таблицы InnoDB (файлы с расширением `.frm'), и начать свою работу сначала, а не ждать завершения выполнения миллионов операций ввода/вывода диска./10/

InnoDB и репликация в MySQL

Репликация в MySQL для InnoDB работает также, как и в MyISAM.

Можно реплицировать модификации главной тадлицы InnoDB в подчиненную таблицу MyISAM. Чтобы установить новую подчиненную таблицу для главной таблицы, необходимо сделать копию табличного пространства InnoDB и файлов журналов, а также файлов .frm и таблицы InnoDB, и переместить копии в подчиненную таблицу./5/

На репликацию InnoDB существуют некоторые ограничения:

- LOAD TABLE FROM MASTER не работает для таблиц типа InnoDB. С этим можно справится двумя способами:
- Выполняя дамп таблицы на главном сервере и импортируя файл дампа на подчиненный сервер;
- Используя команду ALTER TABLE имя_таблицы TYPE=MyISAM на главном сервере перед настройкой репликации с помощью LOAD TABLE имя_таблицы FROM MASTER, а затем используя ALTER TABLE для последующего преобразования главной таблицы обратно к типу InnoDB;
    - до выхода версии MySQL 4.0.6. незавершенная транзакция будет отключена и следующая команда SLAVE START выполнит только оставшуюся часть транзакции;
    - до выхода версии MySQL 4.0.6. аварийный отказ подчиненного сервера во время выполнения транзакции с несколькими операторами приводил к возникновению тех же проблем, что и при SLAVE STOP;
    - до выхода версии MySQL 4.0.11 репликация оператора SET FOREIGN_KEY_CHECKS=0 работала не корректно./5/

Большинство эти ограничения можно обойти, если использовать более новые версии сервера.

Список литературы:

1. Артеменко Ю. MySQL. Справочник по языку: Пер. с англ. — М.: Издательский дом "Вильямс",
2005. — 432 с. — Парал. тит. англ., стр.18-19.

2. http://www.opennet.ru/tips/1958_mysql_myisam_innodb.shtml

3. www.innodb.com

4. Поль Дюбуа. MySQL,2-е изд.:Пер.с англ.-М.:Издательский дом «Вильямс», 2004.-1056с.:ил.-Парал.тит.англ.

5. Компания MySQL AB. MySQL.Руководство администратора.:Пер.с англ.-М.: Издательский дом «Вильямс»,2005.-624с.-Парал.тит.англ.

6. www.pentarh.com/wp/2011/03/02/mysql-innodb-highload-optimization/

7. www.mysql.ru/docs/man/MyISAM.html

8. www.mysql.ru/docs/man/InnoDB.html

9. www.innodb.com

10. www. mysql.ru/docs/man/Table-types.html

11. В. Васвани. MySQL: использование и администрирование = MySQL Database Usage & Administration. — М.: «Питер», 2011. — 368 с. — ISBN 978-5-459-00264-5

12.Роберт Шелдон, Джоффрей Мойе. MySQL 5: базовый курс = Beginning MySQL. — М.: «Диалектика», 2007. — 880 с. — ISBN 978-5-8459-1167-4

13.Кузнецов Максим, Симдянов Игорь. MySQL на примерах. — Спб.: «БХВ-Петербург», 2008. — С. 952. — ISBN 978-5-9775-0066-1

13.Поль Дюбуа. MySQL, 3-е издание = MySQL, 3ed. — М.: «Вильямс», 2006. — 1168 с. — ISBN 5-8459-1119-2

14.Кузнецов Максим, Симдянов Игорь. MySQL 5. В подлиннике. — Спб.: «БХВ-Петербург», 2006. — С. 1024. — ISBN 5-94157-928-4

15.Кузнецов Максим, Симдянов Игорь. Самоучитель MySQL 5. — Спб.: «БХВ-Петербург», 2006. — С. 560. — ISBN 5-94157-754-0