Mysql server

Оглавление

Общие принципы обеспечения безопасности

При работе в MySQL старайтесь следовать приведенным ниже инструкциям:

- **Не предоставляйте никому (за исключением пользователя mysql под именем root) доступа к таблице user в базе данных mysql! Это чрезвычайно важно**. **В MySQL зашифрованный пароль является реальным паролем**. Узнав пароль, занесенный в таблицу user, и имея доступ к удаленному компьютеру, занесенному в соответствующую учетную запись, **войти в систему под именем зарегистрированного владельца пароля легко может кто угодно**.
- Изучите систему прав доступа MySQL. Для управления доступом к MySQL служат команды GRANT и REVOKE. Предоставляйте ровно столько прав, сколько необходимо, и не больше. Никогда не предоставляйте права всем хостам. Полезно проводить следующие контрольные проверки:
    - Выполните команду mysql -u root. Если удается успешно установить соединение с сервером без получения запроса пароля, значит, у вас имеются проблемы. Это означает, что кто угодно может подсоединиться к вашему серверу MySQL как клиент MySQL под именем root, получая таким образом право неограниченного доступа! Проанализируйте инструкцию по инсталляции MySQL, обращая особое внимание на ту часть, которая касается задания пароля пользователя root.
    - С помощью команды SHOW GRANTS проверьте, кто и к каким ресурсам имеет доступ. Воспользуйтесь командой REVOKE, отмените права доступа, которые не являются необходимыми.
- Не храните в базе данных незашифрованных паролей. Если злоумышленнику удастся получить доступ на ваш компьютер, то в его руках окажется полный список паролей, которыми он может воспользоваться. Применяйте для шифрования MD5(), SHA1() или другие односторонние хеш-функции.
- Не используйте в качестве пароля слова из словарей. Для взлома такого рода паролей имеются специальные программы. Даже слова типа ``xfish98'' - это очень плохие пароли. Куда лучше ``duag98'': здесь используется то же слово ``fish'', но при этом буквы в нем заменены ближайшими к ним слева буквами клавиатуры QWERTY. Еще один метод - составить парольное слово из первых букв слов какого либо словосочетания, например ``Mhall'' - по фразе ``Mary had a little lamb.'' Такой пароль легко запоминается и его легко вводить. А вот разгадать его тому, кто не знает ключевой фразы, будет непросто.
- Приобретите брандмауэр. Эта мера обеспечит защиту как минимум от половины всех видов несанкционированного использования любого ПО, с которым вы работаете. Разместите MySQL за брандмауэром или в демилитаризованной зоне (demilitarised zone - DMZ). Полезно проводить следующие контрольные проверки:
- Попробуйте просканировать ваши порты из Internet с помощью утилиты типа nmap. MySQL использует по умолчанию порт 3306. Этот порт должен быть недоступен с неблагонадежных компьютеров. Еще один простой способ проверить, открыт или нет ваш MySQL-порт, - попытаться выполнить с какой либо удаленной машины следующую команду, где server_host - имя хоста, на котором установлен ваш сервер MySQL:
- shell> telnet server_host 3306

Если соединение будет установлено, и вы получите какие-либо бессмысленные символы, это будет означать, что порт открыт, и его нужно закрыть на брандмауэре или маршрутизаторе (если, конечно, нет действительно веских причин держать его открытым). Если же telnet просто зависнет или в подсоединении будет отказано, тогда все в порядке: порт заблокирован.

- Не доверяйте никаким данным, которые вводят пользователи. Возможны попытки перехитрить вашу программу путем ввода последовательностей специальных или экранированных символов в веб-формы, URL-ы или любое приложение, созданное вами. Убедитесь, что защита вашего приложения не будет нарушена, если пользователь введет что-нибудь типа "'; DROP DATABASE mysql;''. Это крайний случай, но действия хакеров, использующих подобную технологию, могут привести к потере информации и появлению брешей в системе безопасности, если вы не готовы к ним. Не следует также забывать о необходимости проверки цифровых данных (распространенной ошибкой является защита только строк). Некоторые полагают, что если в базе данных хранятся только открытые данные, то в ее защите нет необходимости. Это неверно. Такие базы могут стать объектом успешных атак типа отказа от обслуживания. Простейший способ защиты от взломов такого типа - заключать числовые константы в кавычки: SELECT * FROM table WHERE ID='234', а не SELECT * FROM table WHERE ID=234. MySQL автоматически преобразует эту строку в число и выбросит из нее все нецифровые символы. Полезно проводить следующие контрольные проверки:
  - Для всех веб-приложений:
    - Попробуйте ввести во все ваши веб-формы одинарные и двойные кавычки - `'' и `"'. Если MySQL выдаст любое сообщение об ошибке, немедленно разберитесь, в чем дело.
    - Попробуйте видоизменять динамические URL, добавляя в них %22 (`"'), %23 (`#'), и %27 (`'').
    - Попробуйте модифицировать типы данных в динамических URL - замените числовые на символьные, используя символы из предыдущих примеров. Ваше приложение должно быть устойчиво к подобного рода атакам.
    - Попробуйте вводить в числовые поля вместо цифр буквы, пробелы и специальные символы. Ваше приложение должно либо удалять их до передачи в MySQL, либо выдавать сообщение об ошибке. Пропускать в MySQL значения без проверки очень опасно!
    - Проверяйте размер данных перед тем, как они будут переданы в MySQL.
    - При подключении приложения к базе данных лучше использовать имя пользователя, отличное от того, которое вы используете для целей администрирования. Не предоставляйте своим приложениям больше прав доступа, чем это необходимо.
  - Пользователям PHP:
    - Проверьте функцию addslashes(). Что касается PHP 4.0.3, то в нем имеется функция mysql_escape_string(), базирующаяся на функции с тем же именем из MySQL C API.
  - Пользователям MySQL C API:
    - Проверьте API-вызов mysql_real_escape_string().
  - Пользователям MySQL++:
    - Проверьте такие модификаторы, как escape и quote, - для потоков запросов.
  - Пользователям Perl DBI:

        - Проверьте метод quote() или используйте для проверки заполнители.
    - Пользователям Java JDBC:
        - Используйте для проверки объект PreparedStatement и символы-заполнители.
- Не передавайте по Internet открытые (незашифрованные) данные. Они могут оказаться у кого угодно, имеющего достаточно времени и возможностей для того, чтобы перехватить их и использовать в своих целях. Используйте вместо этого протоколы с шифрованием данных, такие как SSL и SSH. MySQL, начиная с версии 4.0.0, поддерживает собственные SSL-соединения. Пересылка по SSH (SSH Port Forwarding) может быть использована для создания туннеля передачи данных с шифрованием и сжатием.
- Научитесь пользоваться утилитами tcpdump и strings. В большинстве случаев проверить, являются ли потоки данных MySQL зашифрованными, можно с помощью команды, подобной той, которая приведена ниже:
- shell> tcpdump -l -i eth0 -w - src or dst port 3306 | strings

  (Она работает под Linux, и будет, с незначительными изменениями, работать под другими системами.) Предупреждение: если вы не видите данных, это еще не гарантирует того, что они зашифрованы. Если требуется высокий уровень безопасности, обратитесь к экспертам в этой области.

Как обезопасить MySQL от хакеров

Для обеспечения безопасности MySQL-системы необходимо строго придерживаться следующих рекомендаций:

- У всех пользователей MySQL должны быть пароли. Для приложений клиент/сервер является общепринятым, что клиент может указывать любое имя пользователя, но если для other_user не задан пароль, то кто угодно может зайти под любым именем, просто введя mysql -u other_user db_name. Чтобы этого избежать, можно изменить пароль для всех пользователей, отредактировав скрипт mysql_install_db перед запуском приложения, или только пароль для root-пользователя MySQL, как это показано ниже:
- shell> mysql -u root mysql
- mysql> UPDATE user SET Password=PASSWORD('new_password')
- WHERE user='root';
- mysql> FLUSH PRIVILEGES;
- Не запускайте демон MySQL от имени пользователя Unix root. Это очень опасно, потому что любой пользователь, имеющий привилегию FILE, будет в состоянии создавать файлы как пользователь root (например ~root/.bashrc). Чтобы предотвратить это, mysqld откажется запускаться от имени пользователя root, если это не будет задано напрямую с помощью опции --user=root. В то же время mysqld может быть запущена от имени обычного непривилегированного пользователя. Можно также, в целях еще большего укрепления безопасности, создать новый аккаунт Unix-пользователя mysql. При запуске mysqld от имени другого пользователя Unix у вас отпадает необходимость заменять имя пользователя root в таблице user, так как имена пользователя в MySQL не имеют ничего общего с аккаунтами пользователей Unix. Для запуска mysqld от имени другого пользователя Unix добавьте в группу [mysqld] файла опций `/etc/my.cnf' или файла опций `my.cnf', находящегося в каталоге данных сервера, строку user, задающую имя пользователя. Например:

- [mysqld]
- user=mysql

  В результате сервер будет запущен от имени назначенного пользователя, независимо от того, производится запуск вручную или

  посредствомsafe_mysqld или mysql.server.

- Откажитесь от поддержки символических ссылок на таблицы (ее можно запретить с помощью опции --skip-symlink). Это особенно важно в том случае, если вы запускаете mysqld от имени пользователя root, поскольку у того, кто имеет право доступа для записи в каталоги данных mysqld, появляется возможность стереть любой файл в системе!.
- Удостоверьтесь, что пользователь Unix, от имени которого запускается mysqld, является единственным пользователем, имеющим привилегии чтения/записи в директории базы данных.
- Не предоставляйте привилегии PROCESS всем пользователям. Команда mysqladmin processlist выводит текст запросов, обрабатываемых в данный момент. Следовательно, любой пользователь, имеющий право на выполнение этой команды, получает возможность прочитать, например, такой запрос другого пользователя, как UPDATE user SET password=PASSWORD('not_secure'). mysqld резервирует добавочное подключение для пользователей, имеющих привилегию PROCESS, так что пользователь MySQL под именем root может подключиться и осуществлять контроль даже в том случае, когда все обычные подключения заняты.
- Не предоставляйте привилегии FILE всем пользователям. Любой пользователь, имеющий такую привилегию, может записать в любом месте файловой системы файл с привилегиями демона mysqld! Чтобы обеспечить здесь хоть минимальную защиту, все файлы создаваемые с помощью команды SELECT ... INTO OUTFILE, сделаны общедоступными для чтения, но перезаписать существующие файлы нельзя. Привилегия FILE может быть также использована для чтения любого файла, доступного пользователю Unix, от имени которого запускается сервер. Это может быть использовано в корыстных целях. Возможно, например, с помощью команды LOAD DATA загрузить `/etc/passwd' в таблицу и прочесть ее позже с помощью SELECT.
- Если вы не доверяете своему DNS-серверу, используйте в таблицах привилегий вместо имен хостов IP-адреса. В любом случае следует очень осторожно относиться к внесению в таблицы привилегий записей, в которых значения имени хоста содержат шаблонные символы!
- Чтобы ограничить число подключений, доступных для отдельного пользователя, можно в mysqld задать значение переменной max_user_connections.

Предотвращение катастроф и восстановление

BACKUP TABLE tbl_name[,tbl_name...] TO '/path/to/backup/directory'

Копирует в каталог резервного копирования тот минимум табличных файлов, который достаточен для восстановления таблицы. На данный момент работает только для таблиц MyISAM. Для таблицMyISAM копирует файлы `.frm' (определений) и `.MYD' (данных). Индексные файлы могут быть реконструированы по этим двум.

RESTORE TABLE tbl_name[,tbl_name...] FROM '/path/to/backup/directory'

Восстанавливает таблицу(ы) из резервной копии, созданной с помощью BACKUP TABLE. Существующие таблицы не перезаписываются: при попытке восстановления поверх существующей таблицы будет выдана ошибка. Восстановление занимает больше времени, нежели BACKUP - из-за необходимости повторного построения индекса. Чем больше в таблице будет ключей, тем больше времени заберет реконструкция. Эта команда, так же как и BACKUP TABLE, в настоящее время работает только для таблиц MyISAM.

CHECK TABLE tbl_name[,tbl_name...] [option [option...]]

option = QUICK | FAST | MEDIUM | EXTENDED | CHANGED

CHECK TABLE работает только на таблицах MyISAM и InnoDB. На таблицах типа MyISAM команда эквивалентна запуску на таблице myisamchk -m table_name. Если опция не указана, используется MEDIUM. Проверяет таблицу(ы) на наличие ошибок. Для таблиц MyISAM обновляется статистика ключей.

Различные типы проверки означают следующее:

Тип	**Действия**
QUICK	Не сканировать строки для проверки на неправильные связи.
FAST	Проверять только таблицы, которые не были корректно закрыты.
CHANGED	Проверять только таблицы, которые изменились со времени последней проверки или не были закрыты корректно.
MEDIUM	Сканировать строки для проверки того, что уничтоженные связи в порядке. При этом также подсчитывается ключевая контрольная сумма для строки и сравнивается с подсчитанной контрольной суммой для ключей.
EXTENDED	Выполнить полный просмотр ключа для всех ключей для каждой строки. Успех такой проверки гарантирует 100%-ное отсутствие противоречий в таблице, но на проверку уйдет немало времени!

Проверочные опции можно сочетать:

CHECK TABLE test_table FAST QUICK;

Эта команда просто вызовет быструю проверку таблицы для выявления того, была ли она закрыта корректно.

Примечание: в некоторых случаях CHECK TABLE изменяет таблицу! Это происходит, если таблица помечена как 'поврежденная/corrupted' или 'не закрытая корректно/not closed properly', а CHECK TABLE не находит никаких проблем в таблице. В этом случае CHECK TABLE отметит в таблице, что с ней все нормально.

Если таблица повреждена, то, скорее всего, проблема в индексах, а не в данных. Проверки всех типов обеспечивают всестороннюю проверку индексов и тем самым должны обнаруживать большинство ошибок.

REPAIR TABLE tbl_name[,tbl_name...] [QUICK] [EXTENDED] [USE_FRM]

REPAIR TABLE работает только на таблицах типа MyISAM и эквивалентна выполнению на таблице myisamchk -r table_name.

Если указан QUICK, то MySQL будет пытаться сделать REPAIR только индексного дерева.

Если используется EXTENDED, то MySQL будет создавать индекс строка за строкой вместо создания по одному индексу единоразово с помощью сортировки; такая техника может работать лучше сортировки для ключей фиксированной длины, если речь идет о хорошо сжимаемых ключах типа char() большой длины.

Что касается MySQL 4.0.2, то тут для REPAIR существует режим USE_FRM. Используйте его, если отсутствует файл `.MYI' или поврежден его заголовок. В этом режиме MySQL воссоздаст таблицу, используя информацию из файла `.frm'. Этот вид исправления в myisamchk недоступен.

Использование myisamchk для профилактики таблиц и послеаварийного

Начиная с версии MySQL 3.23.13 таблицы MyISAM можно проверять с помощью команды CHECK TABLE. Для исправления таблиц используется команда REPAIR TABLE.

Для проверки/ремонта таблиц типа MyISAM (`.MYI' и `.MYD') следует использовать утилиту myisamchk, а для ISAM (`.ISM' и `.ISD'') - утилиту isamchk .

Ниже мы будем говорить о myisamchk, но все сказанное справедливо также и для более старой isamchk.

Утилиту myisamchk можно использовать для получения информации о таблицах рабочей базы данных, для их проверки и исправления или же оптимизации. В следующих разделах описывается, как запускать myisamchk (включая описание ее опций), как настроить график профилактики таблицы и как использовать myisamchk для выполнения различных функций.

В большинстве случаев для оптимизации и исправления таблиц можно также использовать команду OPTIMIZE TABLES, но этот вариант не такой быстрый и не такой надежный (в случае действительно фатальных ошибок), как myisamchk. С другой стороны, OPTIMIZE TABLE проще в использовании и освобождает от забот со сбросом таблиц на диск.

Хотя исправление при помощи myisamchk и достаточно безопасно, никогда не будет лишним сделать резервную копию прежде, чем выполнять ремонт (или любые другие действия, которые могут привнести в таблицу значительные изменения)

Синтаксис вызова: myisamchk [options] tbl_name

Опции options определяют, что должна сделать myisamchk. В данном разделе дается описание этих опций (список опций можно также получить, запустивmyisamchk --help). Если опции не указаны, myisamchk просто проверяет таблицу. Чтобы получить

дополнительную информацию или указать myisamchkвыполнить корректирующие действия, надо задать опции, как это описано в этом и в следующих разделах.

tbl_name - это таблица базы данных, которую нужно проверить/исправить.
Если myisamchk запускается не из каталога базы данных, то следует задать путь к файлу, поскольку myisamchk не имеет представления о том, где искать базу данных. В действительности для myisamchk не важно, где находятся рабочие файлы - в каталоге базы данных или нет; можно скопировать файлы, относящиеся к базе данных, в другое место и выполнить операции восстановления над ними там.

При желании в командной строке myisamchk можно перечислить имена нескольких таблиц. В качестве имени можно также указать имя индексного файла (с суффиксом `.MYI'), что позволит задавать все таблицы в каталоге при помощи шаблона `*.MYI'. Например, находясь в каталоге базы данных, можно проверить все таблицы этого каталога следующим образом:

```
shell> myisamchk *.MYI
```

Если каталог базы данных не является текущим, то все таблицы каталога можно проверить, указав к нему путь:

```
shell> myisamchk /path/to/database_dir/*.MYI
```

Можно даже проверить все таблицы во всех базах данных, если задать шаблон вместе с путем к каталогу данных MySQL:

```
shell> myisamchk /path/to/datadir/*/*.MYI
```

Быстро проверять все таблицы рекомендуется следующим образом:

```
myisamchk --silent --fast /path/to/datadir/*/*.MYI
isamchk --silent /path/to/datadir/*/*.ISM
```

Если необходимо проверить все таблицы и исправить все поврежденные из них, можно использовать следующую строку:

```
myisamchk --silent --force --fast --update-state -O key_buffer=64M \
          -O sort_buffer=64M -O read_buffer=1M -O write_buffer=1M \
          /path/to/datadir/*/*.MYI
isamchk --silent --force -O key_buffer=64M -O sort_buffer=64M \
        -O read_buffer=1M -O write_buffer=1M /path/to/datadir/*/*.ISM
```

Эти команды предполагают, что имеется более чем 64 Мб свободного пространства.

Следует отметить, что если выдается ошибка, подобная следующей:

```
myisamchk: warning: 1 clients is using or hasn't closed the table properly
```

то это означает, что делается попытка проверить таблицу, обновленную другой программой (такой как mysqld), которая еще не закрыла файл или чье выполнение было прервано без возможности корректно закрыть файл.

Общие опции для myisamchk

myisamchk поддерживает следующие опции.

-# или --debug=debug_options

Вывод отладочной информации. Часто строка debug_options имеет следующий вид d:t:o,filename.

-? или --help

Отображение справочного сообщения с завершением работы.

-O var=option, --set-variable var=option

Устанавливает значение переменной. Вывести список допустимых переменных и их значений по умолчанию для myisamchk можно с помощью myisamchk --help:

Переменная	**Значение**
key_buffer_size	523264
read_buffer_size	262136
write_buffer_size	262136
sort_buffer_size	2097144
sort_key_blocks	16
decode_bits	9

sort_buffer_size применяется, когда ключи исправляются посредством сортировки ключей (обычный случай при указании --recover), akey_buffer_size - если таблица проверяется с --extended-check или если ключи исправляются посредством вставки ключей в таблицу построчно (как при выполнении обычных вставок). Исправление через ключевой буфер применяется в следующих случаях:

- Если используется --safe-recover.
- Если размер требуемых для сортировки временных файлов будет более чем вдвое превышать объем, требующийся при создании ключевого файла непосредственно. Так часто обстоит дело, когда присутствуют большие ключи типов CHAR, VARCHAR или TEXT, поскольку при сортировке необходимо сохранять ключи целиком. Имея временное пространство на диске в избытке, можно заставить myisamchk делать исправления посредством сортировки, задав опцию --sort-recover.

Ремонт посредством ключевого буфера требует значительно меньше пространства, чем при использовании сортировки, однако выполняется

значительно медленнее. Когда желательно ускорить выполнение ремонта/исправления, переменные нужно установить равными приблизительно 1/4 доступной памяти. Можно для обеих переменных задавать большие значения, поскольку всякий раз будет использоваться только один из рассматриваемых буферов.

-s или --silent

Молчаливый режим. Выдавать сообщения только при возникновении ошибок. Можно использовать -s дважды (-ss), чтобы предельно ограничить выдачу сообщений утилитой myisamchk.

-v или --verbose

Расширенный режим вывода. Выдается больше информации. Можно использовать с -d и -e. Можно использовать -v многократно (-vv, -vvv) - чтобы еще более расширить сводку!

-V или --version

Отображение версии myisamchk и завершение работы.

-w или, --wait

Если таблица заблокирована, то не выдавать ошибки, а, дождавшись снятия блокировки с таблицы, продолжить выполнение. Заметим, что если mysqldвыполняется на таблице с --skip-locking, то таблица может быть заблокирована только другой командой myisamchk.

Проверочные опции для myisamchk

-c или --check

Проверить таблицы на ошибки. Является операцией по умолчанию, если myisamchk не передаются другие опции, меняющие это поведение.

-e или --extend-check

Проверить таблицу очень тщательно (выполняется достаточно медленно в случае большого количества индексов). Эту опцию следует использовать в экстремальных ситуациях. В большинстве случаев myisamchk или myisamchk --medium-check вполне достаточно для выявления ошибок в таблице. Если используется --extended-check и система располагает приличным объемом памяти, то следует значительно увеличить значениеkey_buffer_size!

-F или --fast

Проверять только таблицы, которые не были корректно закрыты.

-C или --check-only-changed

Проверять только таблицы, изменившиеся с момента последней проверки.

-f или --force

Выполнять перезапуск myisamchk с -r (исправить) на таблице, если myisamchk найдет в ней хоть одну ошибку.

-i или --information

Выдавать статистическую информацию о проверяемой таблице.

-m или --medium-check

Быстрее, чем расширенная проверка (extended-check), но при этом обнаруживается только 99,99% из общего числа ошибок (чего, однако, в большинстве случаев вполне достаточно).

-U или --update-state

Отмечать в файле `.MYI' факт проверки таблицы и наличие повреждений. Опцию следует использовать для получения максимального эффекта от опции --check-only-changed, однако ее применение недопустимо, если mysqld работает с таблицей и был запущен с опцией --skip-locking.

-T или --read-only

Не отмечать таблицу как проверенную. Это может пригодиться, когда myisamchk используется для проверки таблиц, используемых каким-то другим приложением, и это приложение не выполняет блокировку (как mysqld --skip-locking).

Опции исправления для myisamchk

Следующие опции используются, если myisamchk запускается с -r или -o:

-D # или --data-file-length=# Максимальная длина файла данных (когда файл данных пересоздается при его ``переполнении'').

-e или --extend-check Пробовать исправлять каждую возможную строку из файла данных. Обычно при этом обнаруживается масса замусоренных строк. Использовать эту опцию следует только в самом крайнем случае, когда больше ничего не остается.

-f или --force Писать поверх старых временных файлов (`table_name.TMD') вместо аварийного прекращения.

-k # или keys-used=# Если используется ISAM, то данный параметр предписывает обработчику таблиц ISAM на необходимость обновить только первые# индексов. Если используется MyISAM, то определяет, какие ключи использовать, при этом каждый двоичный бит соответствует одному ключу (первый ключ - это бит 0). Может использоваться для ускорения вставок!

Отключенные индексы можно снова активизировать с помощью myisamchk -r. keys.

-l или --no-symlinks Не рассматривать символические ссылки.
Обычно myisamchk исправляет таблицы, на которые указывают символические ссылки. Данная опция отсутствует в MySQL 4.0, в связи с тем, что MySQL 4.0 не удаляет символические ссылки во время восстановления.

-r или --recover При указании этой опции можно исправить практически все, кроме уникальных ключей, в которых есть повторения (ошибка, вероятность которой мизерна для таблиц ISAM/MyISAM). Если необходимо восстановить таблицу, то начинать надо с этой опции. Только еслиmyisamchk сообщит, что таблица не может быть восстановлена с помощью -r, тогда следует пытаться применять -o (отметим, что в тех маловероятных случаях, когда -r не срабатывает, файл данных остается неизменным), В случае большого объема памяти следует увеличить размерsort_buffer_size!

-o или --safe-recover

Используется старый метод восстановления (читаются подряд все строки и обновляются все деревья индексов на основе найденных строк); такой алгоритм работает на порядок медленнее -r, но метод справляется с несколькими редкими случаями, непосильными для -r. При этом методе восстановления также используется значительно меньше дискового пространства, нежели в случае -r. Обычно всегда следует начинать с исправления посредством -r, и только если результат не будет достигнут, использовать -o. Для систем с большим объемом памяти следует увеличить размер key_buffer_size!

-n или --sort-recover

Заставляет myisamchk использовать сортировку при разрешении ключей, даже если это потребует временных файлов очень большого размера.

--character-sets-dir=...

Каталог, где хранятся кодировки.

--set-character-set=name

Изменить используемую для индекса кодировку

-t или --tmpdir=path

Путь для хранения временных файлов. Если не задан, myisamchk использует для пути переменную окружения TMPDIR.

-q или --quick

Быстрый ремонт без изменения файла данных. Можно добавить вторую -q, чтобы дать myisamchk санкцию на изменение исходного файла данных в случае дублирования ключей

-u или --unpack

Распаковать файл, упакованный в myisampack.

Другие опции для myisamchk

Кроме ремонта и проверки таблиц, myisamchk может выполнять другие операции:

-a или --analyze

Анализировать распределение ключей. Улучшает эффективность операции связывания за счет включения оптимизатора связей. Он обеспечивает лучший порядок связывания таблиц и определяет, какие ключи при этом следует использовать: myisamchk --describe --verbose table_name или посредством SHOW KEYS в MySQL.

-d или --description

Отображает некоторую информацию о таблице.

-A или --set-auto-increment[=value]

Предписывает, чтобы отсчет значений AUTO_INCREMENT начинался с value или большего значения. Если значение не указано, то в качестве следующего значения AUTO_INCREMENT берется наибольшее использованное значение для автоинкрементного ключа + 1.

-S или --sort-index

Сортировать блоки индексного дерева в порядке от больших к меньшим (high-low). Этим оптимизируются операции поиска и повышается скорость сканирования по ключу.

-R или --sort-records=#

Сортирует записи в соответствии с индексом. Это значительно повышает локализацию данных и может ускорить операции SELECT и ORDER BY, которые выполняются по индексу и выбирают данные по какому-либо интервалу. (Возможно, что первая сортировка будет выполняться очень медленно!) Чтобы узнать номера индексов таблицы, нужно использовать команду SHOW INDEX, показывающую индексы таблицы в том же порядке, в каком их видитmyisamchk. Индексы нумеруются начиная с 1.

Использование myisamchk для послеаварийного восстановления

При выполнении mysqld со --skip-locking (установка по умолчанию в некоторых системах, подобных Linux) применение myisamchk для проверки таблицы, когда она

используется mysqld, не совсем безопасно. Если есть уверенность, что никто не обратится к таблицам через mysqld во время выполненияmyisamchk, то достаточно до начала проверки таблиц выполнить mysqladmin flush-tables, если нет - то на время проверки таблиц необходимо приостановить mysqld. При запуске myisamchk в то время, когда mysqld обновляет таблицы, может быть выдано предупреждение о повреждении таблицы - даже в случае, если этого не произошло.

Если --skip-locking не используется, то проверять таблицы с помощью myisamchk можно в любое время. Во время проверки все пытающиеся обновить таблицу клиенты получат возможность сделать это, только дождавшись готовности myisamchk.

Если myisamchk применяется для ремонта или оптимизации таблиц, то всегда необходимо обеспечить отсутствие обращений сервера mysqld к таблице (это также относится к случаю использования --skip-locking). Если mysqld не может быть приостановлен, то до myisamchk, как минимум, надо выполнитьmysqladmin flush-tables. Таблицы могут быть повреждены, если сервер и myisamchk обратятся к таблицам одновременно.

В данном разделе описывается, как выявлять повреждения данных в базах данных MySQL и что делать с повреждениями дальше. Если таблица повреждается часто, то надо постараться отыскать причину этих повреждений!

При выполнении послеаварийного восстановления важно понимать, что каждой таблице tbl_name в базе данных соответствуют три файла в каталоге базы данных:

Файл	**Назначение**
`tbl_name.frm'	Файл определения таблицы (формы)
`tbl_name.MYD'	Файл данных
`tbl_name.MYI'	Индексный файл

Каждый из этих трех типов файлов ``имеет'' свои виды повреждений, но наиболее часто проблемы возникают с файлами данных и индексными файлами.

Во время своей работы myisamchk построчно создает копию файла (данных) `.MYD'. Стадия исправления завершается тем, что программа удаляет старый файл `.MYD' и переименовывает новый путем присвоения ему имени исходного. Если используется --quick, myisamchk не создает временного файла`.'MYD, а, исходя из предположения, что файл `.MYD' правилен, только формирует новый индексный файл, никак не меняя файл `.'MYD. Это безопасно, поскольку myisamchk автоматически распознает, что файл `.MYD' запорчен, и в этом случае прерывает исправление. Можно также задавать для myisamchkдве опции --quick. В этом случае myisamchk не прерывается аварийно по некоторым ошибкам (таким как дублирование ключа), а пытается исправить их путем модификации файла `.MYD'. Обычно использование двух опций --quick имеет смысл только в случае, если свободного места на диске недостаточно для выполнения нормального исправления. Тогда перед запуском myisamchk следует по крайней мере выполнить резервное копирование.

Как проверять таблицы на ошибки

Для проверки таблицы MyISAM используются следующие команды:

myisamchk tbl_name

> Находит 99,99% всех ошибок. Не в состоянии отыскать повреждений, затрагивающих только файл данных (которые весьма необычны). Если необходимо только проверить таблицу, то обычно следует выполнить myisamchk без опций либо с одной из опций -s или --silent.

myisamchk -m tbl_name

> Находит 99,999% всех ошибок. Сначала на ошибки проверяются все индексные элементы, а затем читаются все строки подряд. Программа вычисляет контрольную сумму для всех ключей в строке и проверяет, совпадает ли она с контрольной суммой в индексном дереве.

myisamchk -e tbl_name

> В этом случае выполняется полная и тщательная проверка всех данных (-e означает ``расширенная проверка''). Происходит тестовое чтение каждого ключа для каждой строки с целью контроля того, что ключи указывают на нужные строки. Для большой таблицы с множеством ключей на это может потребоваться много времени. myisamchk обычно останавливается после обнаружения первой ошибки, но если желательно получить более подробную информацию, можно добавить опцию --verbose (-v) - таким образом выполнение myisamchk будет продолжаться вплоть до максимума в 20 ошибок. При нормальной работе достаточно просто запустить myisamchk (без аргументов за исключением имени таблицы).

myisamchk -e -i tbl_name

> Аналогична предыдущей команде, но опция -i указывает myisamchk дополнительно отображать некоторую статистическую информацию.

Как ремонтировать таблицы

В данном разделе рассматривается только использование myisamchk на таблицах MyISAM (расширения `.MYI' и `.MYD'). Если же в системе применяются таблицы ISAM (расширения `.'ISM и `.'ISD), то следует пользоваться isamchk.

Начиная с версии MySQL 3.23.14 можно ремонтировать таблицы MyISAM при помощи команды REPAIR TABLE.

К симптомам повреждения таблицы относятся неожиданные прерывания выполнения запросов и появление следующих ошибок:

- `tbl_name.frm' is locked against change (Файл заблокирован для изменений)

- Can't find file `tbl_name.MYI' (Errcode: ###) (Не могу найти файл `tbl_name.MYI' (Ошибка: ###))
- Unexpected end of file (Неожиданно наступил конец)
- Record file is crashed (Файл записей испорчен)
- Got error ### from table handler (Получена ошибка ### от дескриптора таблицы). Для получения более подробной информации об ошибке можно выполнить perror ###. Чаще всего о проблемах с таблицей свидетельствуют следующие ошибки:
- shell> perror 126 127 132 134 135 136 141 144 145
- 126 = Index file is crashed / Wrong file format
- 127 = Record-file is crashed
- 132 = Old database file
- 134 = Record was already deleted (or record file crashed)
- 135 = No more room in record file
- 136 = No more room in index file
- 141 = Duplicate unique key or constraint on write or update
- 144 = Table is crashed and last repair failed
- 145 = Table was marked as crashed and should be repaired

  Заметим, что ошибка 135 - 'no more room in record file' ('не осталось места в файле записей'), не может быть исправлена просто выполнением ремонта. В этом случае необходимо использовать следующую команду:

  ALTER TABLE table MAX_ROWS=xxx AVG_ROW_LENGTH=yyy;

В других случаях следует выполнять ремонт таблиц. myisamchk обычно может обнаружить и исправить большинство неполадок.

Процесс ремонтирования включает до четырех описанных здесь стадий. Перед тем как приступить к ремонту, необходимо выполнить `cd' в каталог базы данных и проверить права доступа к табличным файлам. Файлы должны быть доступны для чтения Unix-пользователю, от имени которого выполняетсяmysqld, (а также выполняющему ремонт, поскольку ему приходится обращаться к проверяемым файлам). Если появится необходимость изменять файлы, то проверяющий также должен иметь доступ для записи.

Если ремонт таблицы планируется выполнять из командной строки то сначала требуется остановить сервер. Следует отметить, что при выполненииmysqladmin shutdown с удаленного сервера mysqld все еще будет некоторое время работать после завершения mysqladmin, пока не будут остановлены все запросы и сброшены на диск все ключи.

Стадия 1: проверка таблиц

Выполните myisamchk *.MYI или, если вы располагаете временем, myisamchk -e *.MYI. Используйте опцию -s (молчаливый режим) для подавления ненужной информации.

Если mysqld остановлен, то следует использовать опцию --update-state для указания myisamchk отмечать таблицы как 'проверенные'(checked).

Ремонтировать следует только те таблицы, для которых myisamchk выдала ошибки. Для таких таблиц следует перейти к стадии 2.

Если во время проверки будут получены странные ошибки (подобные out of memory), или myisamchk завершится аварийно, то перейдите к стадии 3.

Стадия 2: легкий безопасный ремонт

Примечание: если есть желание ускорить ремонт, рекомендуется добавить: -O sort_buffer=# -O key_buffer=# (где # примерно 1/4 от имеющейся памяти) во всех командах isamchk/myisamchk.

Сначала надо попробовать запустить myisamchk -r -q tbl_name (-r -q означает "режим быстрого восстановления"). При этом будет сделана попытка исправить индексный файл без изменения файла данных. Если в файле данных содержится все необходимое, а удаленные связи указывают на правильные позиции в файле данных, то команда должна дать результат и таблица будет исправлена. Перейдите к ремонту следующей таблицы. В противном случае следует выполнить следующие действия:

1. Сделать резервную копию файла данных.
2. Использовать myisamchk -r tbl_name (-r означает "режим восстановления"). При этом из файла данных будут удалены некорректные и уничтоженные записи, и будет заново создан индексный файл.
3. Если на предыдущем шаге проблему решить не удастся, то используйте myisamchk --safe-recover tbl_name. В режиме безопасного восстановления используется старый метод восстановления, справляющийся с некоторыми случаями, которые оказываются не под силу для режима обычного исправления (но работает этот метод медленнее).

Если во время проверки будут получены странные ошибки (подобные out of memory) или myisamchk аварийно завершается, то перейдите к стадии 3.

Стадия 3: сложный ремонт

До этой стадии дело доходит, только если первый 16-килобайтный блок в индексном файле разрушен или содержит неверную информацию, либо когда индексный файл отсутствует. В этом случае необходимо создать новый индексный файл. Необходимо выполнить следующие действия:

1. Переместить файл данных в какое-нибудь безопасное место.
2. Использовать файл описания таблицы для создания новых (пустых) файлов - данных и индексного:
3. shell> mysql db_name
4. mysql> SET AUTOCOMMIT=1;
5. mysql> TRUNCATE TABLE table_name;
6. mysql> quit

   Если используемая версия SQL не располагает TRUNCATE TABLE, то взамен используется DELETE FROM table_name.

7. Скопируйте старый файл данных на место недавно созданного (делать перемещение старого файла обратно на место нового нецелесообразно, поскольку в старом файле может снова возникнуть потребность, если что-то пойдет не так).

Вернитесь к стадии 2. myisamchk -r -q теперь должна сработать (но бесконечно повторять стадии не следует).

Что касается MySQL 4.0.2, то тут можно воспользоваться REPAIR ... USE_FRM, выполняющей всю эту процедуру автоматически.

Стадия 4: очень сложный ремонт

До этой стадии вы дойдете только в случае, если ко всему прочему запорчен и файл описания. Такого происходить не должно, поскольку файл описания после создания таблицы не изменяется. Выполните следующие действия:

1. Восстановите файл описания из резервной копии и перейдите к стадии 3. Можно также восстановить индексный файл и вернуться к стадии 2. Во втором случае начинать надо с myisamchk -r.
2. Если резервной копии нет, но точно известно, как таблица создавалась, то создается копия таблицы в другой базе данных. Новый файл данных удаляется, затем файл описания с индексным файлом перемещаются из другой базы данных в поврежденную. Таким образом вы получаете новый файл описания и индексный файл, не затрагивая при этом файла данных. Делается возврат к стадии 2 с попыткой воссоздать индексный файл.

Оптимизация таблиц

Чтобы объединить фрагментированные записи и избавиться от потерь пространства, происходящих из-за удаления и обновления записей, нужно запуститьmyisamchk в режиме восстановления:

shell> myisamchk -r tbl_name

Такую же оптимизацию таблицы можно произвести, используя команду SQL OPTIMIZE TABLE. OPTIMIZE TABLE выполняет ремонт таблицы и анализ ключей, а также сортирует дерево индексов для ускорения поиска ключей. Вдобавок сводится на нет нежелательное взаимодействие между утилитой и сервером, поскольку при использовании OPTIMIZE TABLE работу выполняет сам сервер.

myisamchk также располагает рядом других опций, которые можно использовать для повышения производительности таблицы:

- -S, --sort-index
- -R index_num, --sort-records=index_num
- -a, --analyze
-

Синтаксис команды OPTIMIZE TABLE

OPTIMIZE TABLE tbl_name[,tbl_name]...

Команда OPTIMIZE TABLE должна использоваться после удаления большей части таблицы или если в таблице было внесено много изменений в строки переменной длины (таблицы, в которых есть столбцы VARCHAR, BLOB или TEXT). Удаленные записи

поддерживаются при помощи связного списка, и последующие операции INSERT повторно используют позиции старых записей. Чтобы перераспределить неиспользуемое пространство и дефрагментировать файл данных, можно воспользоваться командой OPTIMIZE TABLE.

На данный момент команда OPTIMIZE TABLE работает только с таблицами MyISAM и BDB. Для таблиц BDB команда OPTIMIZE TABLE выполняет ANALYZE TABLE.

Можно применить OPTIMIZE TABLE к таблицам других типов, запустив mysqld с параметром --skip-new или --safe-mode, но в этом случае OPTIMIZE TABLEлишь только выполняет ALTER TABLE.

Команда OPTIMIZE TABLE работает следующим образом:

- Если в таблице есть удаленные или разделенные строки, восстанавливает таблицу.
- Если индексные страницы не отсортированы - сортирует их.
- Если статистические данные не обновлены (и восстановление нельзя осуществить путем сортировки индексов), обновляет их.

Команда OPTIMIZE TABLE для MyISAM представляет собой эквивалент выполнения myisamchk --quick --check-only-changed --sort-index --analyze над таблицей.

Обратите внимание: во время работы OPTIMIZE TABLE таблица заблокирована!

Синтаксис команды ANALYZE TABLE

ANALYZE TABLE tbl_name[,tbl_name...]

Анализирует и сохраняет распределение ключей для таблицы. Во время проведения анализа таблица заблокирована для чтения. Эта функция работает для таблиц MyISAM и BDB.

Данная команда является эквивалентом выполнения myisamchk -a для таблицы.

Сохраненное распределение ключей в MySQL используется для принятия решения о том, в каком порядке следует связывать таблицы, когда для связывания используются не константы, а другая база.

Эта команда выдает таблицу со следующими столбцами:

Синтаксис команды FLUSH

FLUSH flush_option [,flush_option] ...

Команда FLUSH применяется для очистки части кэша, используемого MySQL. Для запуска FLUSH необходимо обладать привилегиями RELOAD.

Параметр flush_option может быть одним из следующих:

Параметр	**Описание**
HOSTS	Производится очистка таблиц кэша удаленных компьютеров. Сброс таблиц удаленного компьютера следует производить, если один из удаленных компьютеров изменил IP-адрес или если было получено сообщение об ошибке Host ... is blocked. Если во время соединения с сервером MySQL происходит больше ошибок подряд, чем указано в max_connect_errors для определенного удаленного компьютера, то MySQL предполагает, что что-то не в порядке, и блокирует последующие попытки установления соединения со стороны этого удаленного компьютера. Сброс таблиц удаленного компьютера позволяет снова попытаться установить соединение. Ошибка Host '...' is blocked. Чтобы это сообщение об ошибке не появлялось, запустите mysqld с параметром -O max_connection_errors=999999999.
DES_KEY_FILE	Производится перезагрузка ключей DES из файла, указанного параметром --des-key-file, при запуске сервера.
LOGS	Закрываются и повторно открывается все файлы журналов. Если файл журнала обновлений или файл бинарного журнала был указан без расширения, номер расширения файла журнала будет увеличен на единицу относительно предыдущего файла. Если в имени файла было указано расширение, MySQL закроет и повторно откроет файл журнала обновлений. Эти действия аналогичны отправке сигнала SIGHUP на сервер mysqld.
PRIVILEGES	Производится перезагрузка привилегий из таблиц привилегий в базе данных mysql.
QUERY CACHE	Производится дефрагментация кэша запросов, чтобы эффективнее использовать его память. Эта команда не удаляет запросы из кэша, как команда RESET QUERY CACHE.
TABLES	Закрываются все открытые таблицы и принудительно закрываются все используемые таблицы. Также сбрасывается кэш запросов.
[TABLE | TABLES] tbl_name [,tbl_name...]	Производится сброс только указанных таблиц.
TABLES WITH READ LOCK	Закрываются все открытые таблицы и блокируется доступ для чтения всех таблиц для всех баз данных, пока не будет запущена командаUNLOCK TABLES. Это очень удобный способ создавать резервные копии, если у вас файловая система наподобие Veritas, которая может обеспечить моментальные снимки данных в режиме

	реального времени.
STATUS	Большинство переменных состояния сбрасываются в нуль. Эту команду необходимо использовать при отладке запроса.
USER_RESOURCES	Все ресурсы пользователя сбрасываются в нулевое значение. Это позволяет заблокированному пользователю подсоединиться еще раз.

Ко всем приведенным выше командам можно получить доступ при помощи утилиты mysqladmin, используя команды flush-hosts, flush-logs, reload илиflush-tables.

Рекомендуется также ознакомиться с командой RESET, которая применяется с репликацией.

Синтаксис команды RESET

RESET reset_option [,reset_option] ...

Команда RESET используется для очистки. Кроме того, она также действует как более сильная версия команды FLUSH.

Чтобы запустить команду RESET, необходимо обладать привилегиями RELOAD.

Параметр	Описание
MASTER	Удаляет все бинарные журналы, перечисленные в индексном файле, обнуляет значения индексного файла binlog. В версиях до 3.23.26 - FLUSH MASTER (Master)
SLAVE	Сбрасывает положение репликации подчиненного компьютера в журналах головного компьютера. В версиях до 3.23.26 эта команда называласьFLUSH SLAVE (Slave)
QUERY CACHE	Удаляет все результаты запросов из кэша запросов.

Синтаксис команды KILL

KILL thread_id

Каждое соединение с mysqld запускается в отдельном потоке. При помощи команды SHOW PROCESSLIST можно просмотреть список запущенных потоков, а при помощи команды KILL thread_id - удалить поток.

Если у вас есть привилегия PROCESS, можно просмотреть все потоки. Обладая привилегией SUPER, можно удалять любые потоки. В противном случае можно просматривать и удалять только свои собственные потоки.

Для просмотра и удаления потоков можно также применять команды mysqladmin processlist и mysqladmin kill.

При использовании команды KILL для потока устанавливается специальный флаг kill flag.

В большинстве случаев удаление потока занимает некоторое время, поскольку этот флаг проверяется с определенным интервалом.

- В циклах SELECT, ORDER BY и GROUP BY флаг проверяется только после считывания блока строк. Если установлен флаг удаления, то выполнение оператора будет отменено.
- При выполнении команды ALTER TABLE флаг удаления проверяется перед считыванием каждого блока строк из исходной таблицы. Если флаг установлен, то выполнение команды отменяется и временная таблица удаляется. При выполнении команд UPDATE TABLE и DELETE TABLE флаг удаления проверяется после каждого считывания блока, а также после каждого обновления или удаления строки. Если флаг удаления установлен, то выполнение оператора отменяется. Обратите внимание: если не используются транзакции, то отменить изменения будет невозможно!
- GET_LOCK() будет отменен при помощи NULL.
- Поток INSERT DELAYED быстро сбросит все строки, которые он содержит в памяти и будет удален.
- Если поток находится в заблокированной таблице (состояние: Locked), то блокировка таблицы будет быстро отменена.
- Если поток ожидает освобождения дискового пространства в запросе write, запись будет отменена с выдачей сообщения о переполнении диска.