# **Cодержание**

Введение……………………………………………………………………………4

1. Общее описание, запуск и настройка

прав доступа к базам данных MySQL……………………….……..…..5

1. Команды пакета MySQL……………………………………………….…..10
   1. Основные команды………………...…………………………………..11
   2. Дополнительные команды…………………...………………….…….23
2. Настройка mysqld (серверная часть пакета MySQL)…………………….27
3. Возможность языка Perl для работы с MySQL сервером…………...…..29

Заключение……………………………………………..………………..……….31

Список литературы………………………………………….…………………..33

# **Введение.**

На сегодняшний день СУБД MySQL является одной из самых известных, надежных и быстрых из всего семейства существующих СУБД. Почему именно она? Одной из причин являются правила ее распространения — за нее не надо платить деньги и распространяется она вместе со своими исходными текстами, другая причина – это то, что MySQL относительно быстрая СУБД. PostgreSql, например, также распространяется под лицензией \*GNU GPL, но она не получила столь широкого распространения. Одна из причин — это заметная медлительность. Итак, две главные причины популярности MySQL: цена и производительность.

MySQL написан под десятки видов операционных систем. Это и FreeBSD, OpenBSD, MacOS, OS/2, SunOS, Win9x/00/NT и Linux. Сегодня MySQL особенно распространена на платформах Linux и Windows. Причем на последней встречается гораздо реже.

Принцип работы СУБД MySQL аналогичен принципу работы любой СУБД, использующей SQL (Structured Query Language, язык структурированных запросов) в качестве командного языка для создания/удаления баз данных, таблиц, для пополнения таблиц данными, для осуществления выборки данных.

Целью данной работы является описание конфигурирования, администрирования и программирования СУБД MySQL и создание программы иллюстрирующей возможности MySQL.

1. **Общее описание, запуск и настройка прав доступа к**

**базам данных MуSQL.**

MySQL, как и любая другая СУБД представляет собой программу-сервер, которая находится в памяти компьютера и обслуживает TCP порт. В случае с MySQL, номером порта будет являться число 3306. А клиентская программа, будь то CGI-приложение на Perl либо программный продукт на C, соединяется с СУБД по этому порту и посылает ему строчки на SQL. Тот в свою очередь их интерпретирует, выполняя необходимые действия, и отсылает результаты запроса обратно клиенту. Таким способом происходит общение сервера баз данных с клиентскими программами.

##### Для запуска MуSQL-сервера необходимо выполнить файл mysqld.exe. Сервер запускается как безоконный фоновый процесс. При этом он остается в памяти и обрабатывает запросы от клиентских приложений.

Для остановки сервера следует выполнить команду:

mysqladmin -u root shutdown.

Если сервер не был остановлен корректно, то при последующем запуске в файле mysql.err будет добавлена запись о некорректном завершении. Корректная остановка сервера необходима для сохранения всех данных, которые находятся в кэшах MySQL.

MуSQL имеет развитую систему доступа к базам данных. Пользователю базы данных может быть предоставлен доступ ко всей базе данных, отдельным таблицам и отдельным столбцам таблиц. Имеется разграничение на действия, которые может производить пользователь с записями. Для организации такой сложной (на первый взгляд) структуры доступа используется несколько таблиц в специальной базе данных. На основании значений этих таблиц выстраивается политика предоставления доступа.

База данных, которую сервер MуSQL использует для хранения внутренней информации о пользователях, по умолчанию имеет имя mуsql. В этой базе данных определены таблицы для хранения информации пользовательских учетных записей. Содержание БД mysql в таблице 1.1.

Таблица 1.1. Список таблиц прав доступа

|  |  |  |
| --- | --- | --- |
| Порядок проверки | Название таблицы | Описание |
| 1 | user | Список всех пользователей сервера MySQL с правами доступа ко всей базе данных в целом |
| 2 | db | Список баз данных с сопоставленными им пользователями и их привилегиями на выполнение операции |
| 3 | host | Список компьютеров, с которых возможен доступ к сопоставленной компьютеру базе данных, и привилегии на выполнение операций, которые могут проводиться через соединение с этого компьютера |
| 4 | tables\_priv | Список таблиц баз данных, к которым возможен доступ пользователя |
| 5 | columns\_priv | Список столбцов базы данных, к которым возможен доступ пользователя |

Проверка прав доступа производится в порядке, указанном в таблице 1.1. Прежде всего проверяется, существует ли пользователь и какие возможности он имеет при управлении базой данных (таблица1.2.).

Таблица 1.2. Поля таблицы user

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип | Значение по умолчанию | Описание |
| 1 | 2 | 3 | 4 |
| Host | Char(60) |  | Компьютер пользователя, с которого разрешено подключение |
| User | Char(16) |  | Имя пользователя |
| Password | Char(16) |  | Пароль пользователя |
| Select\_priv | Enum(‘N’, ‘Y’) | N | Привилегии на выборку данных оператором SELECT |
| Insert\_priv | Enum(‘N’, ‘Y’) | N | Привилегии на вставку данных оператором INSERT |
| Update\_priv | Enum(‘N’, ‘Y’) | N | Привилегии на изменение данных оператором UPDATE |
| Delete\_priv | Enum(‘N’, ‘Y’) | N | Привилегии на удаление данных оператором DELETE |
| Reload\_priv | Enum(‘N’, ‘Y’) | N | Привилегии на перезапуск сервера MySQL, а также сброс данных из памяти на диск |
| 1 | 2 | 3 | 4 |
| Shutdown\_priv | Enum(‘N’, ‘Y’) | N | Привилегии на отключение сервера MySQL |
| Process\_priv | Enum(‘N’, ‘Y’) | N | Привилегии на просмотр состояния текущих процессов сервера |
| File\_priv | Enum(‘N’, ‘Y’) | N | Привилегии, необходимые для работы с файловой системой, например оператором LOAD DATA INFILE |
| Grant\_priv | Enum(‘N’, ‘Y’) | N | Привилегии, позволяющие пользователю устанавливать привилегии другим пользователям |
| References\_priv | Enum(‘N’, ‘Y’) | N | Привилегии, позволяющие ссылаться на объект |
| Index\_priv | Enum(‘N’, ‘Y’) | N | Привилегии, необходимые для индексирования данных |
| Alter\_priv | Enum(‘N’, ‘Y’) | N | Привилегии, позволяющие обновлять данные |

Поля с типом Enum ( ‘N’, ‘Y’ ) считаются разрешенными, если установлено значение Y, и неразрешенными, если установлено значение N.

Для администратора сервера MуSQL необходимы все привилегии, для обычных пользователей в данной таблице обычно не разрешена ни одна из привилегии, т. е. Обычные пользователи не могут изменять конфигурацию сервера MySQL. Исключением можно считать привилегии File\_priv, если пользователь активно работает с файловой системой.

Поле host может быть описано как IP-адрес или имя, а также может быть пустым. В имени компьютера и в имени пользователя разрешается применение регулярных выражений.

Пароли хранятся в зашифрованном виде и не имеют функции обратной дешифровки. Для того чтобы задать пароль, необходимо использовать функцию password ().

Если привилегии на просмотр процессов сервера не разрешены, тогда пользователь может видеть только собственные процессы.

На этапе проверки значений таблицы user пропускаются для дальнейшей обработки только те соединения, у которых поля Host, User и значения функции password () от значения пароля, полученного от клиента, совпадают.

Дальнейшая обработка приводится в таблице db. На этом этапе определяются привилегии, которыми обладает пользователь для работы с конкретной базой данных (таблица. 1.3).

Таблица 1.3. Привилегии для работы с базой данных

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип | Значение по умолчанию | Описание |
| Host |  |  | Адрес компьютера пользователя, с которого разрешено подключение |
| Db |  |  | Имя базы данных |
| User |  |  | Имя пользователя |
| Select\_priv | Enum(‘N’, ‘Y’) | N | Привилегии на выборку данных оператором SELECT |
| Insert\_priv | Enum(‘N’, ‘Y’) | N | Привилегии на вставку данных оператором INSERT |
| Update\_priv | Enum(‘N’, ‘Y’) | N | Привилегии изменения данных оператором UPDATE |
| Delete\_priv | Enum(‘N’, ‘Y’) | N | Привилегии на удаление данных DELETE |
| Create\_priv | Enum(‘N’, ‘Y’) | N | Привилегии, необходимые для создани таблиц оператором CREATE TABLE |
| Drop\_priv | Enum(‘N’, ‘Y’) | N | Привилегии, необходимые для удаления таблиц оператором DROP TABLE |
| Grant\_priv | Enum(‘N’, ‘Y’) | N | Привилегии, позволяющие пользователю устанавливать привилегии другим пользователям |
| References\_priv | Enum(‘N’, ‘Y’) | N | Привилегии, позволяющие ссылаться на объект |
| Index\_priv | Enum(‘N’, ‘Y’) | N | Привилегии, необходимые для индексирования данных |
| Alter\_priv | Enum(‘N’, ‘Y’) | N | Привилегии, позволяющие обновлять данные |

Привилегии, указанные в таблице db, перекрывают значения, полученные в таблице user, именно поэтому пользователи могут не иметь ни одной привилегии в таблице user.

Один пользователь может иметь несколько различных адресов, с которых может производиться подключение, поэтому на данном этапе проверяется значение Host. Если значение поле Host пусто, тогда проводится дополнительная проверка в таблице host. Таблица Host идентична таблице Db, за исключением того, что в ней отсутствует поле User. Столбцы таблицы host аналогичны столбцам таблицы db, разница состоит только в том, что данные из таблицы host при совпадении полей Host и Db имеют более высокий приоритет, и перекрывают ранее установленные значения.

MySQL сервер позволяет устанавливать ограничения на доступ к отдельным таблицам и столбцам таблицы (эта информация хранится в таблицах tables\_priv и columns\_priv), но увеличение числа проверок приводит к уменьшению производительности.

**2. Команды пакета MySQL.**

СУБД MySQL поставляется со следующими основными программами и скриптами:

* mysql
* mysqlaccess
* mysqladmin
* mysqld
* mysqldump
* mysqlshow
* isamchk
* isamlog
* safe\_mysqld

Еще есть несколько утилит. Они не жизненно важны для MySQL, но обеспечивают полезные дополнительные функциональные возможности.

* comp\_err
* msql2mysql
* mysqlbug
* perror
* replace
* which1
* zap

## 2.1. Основные команды

## mysql

Клиентская программа MySQL.

Запуск:

mysql [OPTIONS] database

ОПИСАНИЕ:

Клиентская часть СУБД MySQL названа MySQL. Она обеспечивает интерфейс командной строки с СУБД MySQL, и возможность неинтерактивной пакетной обработки.

Программой mysql поддерживаются следующие опции. Можно использовать или "короткий" одиночный символ или более подробную версию.

|  |  |
| --- | --- |
| -\?, --help | Справка. |
| -d, --debug=[options] | Вывести в протокол отладочную информацию. В общем виде 'd:t:o,filename`. |
| -d, --debug-info | Вывести отладочную информацию при выходе из программы. |
| -e, --exec | Выполнить команду и выйти, неявная форма опции --batch. |
| -f, --force | Продолжить, даже если есть SQL ошибка. |
| -h, --hostname=[hostname] | Задает имя сервера, с которым необходимо соединиться. |
| -P, --port=[port] | Порт, для соединения с сервером MySQL. |
| -p, --password=[password] | Пароль пользователя, для соединения с сервером MySQL. Не должно быть пробела между -p и паролем. |
| -q, --quick | Быстрый (небуферизованный вывод), может замедлить сервер, если вывод приостановлен. |
| -s, --silent | Работать молча (подавить вывод). |
| -u, --user=[user] | Имя пользователя для соединения с сервером MySQL. Необязательно, если имя пользователя такое же, как логин. По умолчанию логин используется в качестве имени пользователя. |
| -v, --verbose | Подробный вывод. -v опция может быть удвоена или утроена для более подробного вывода. |
| -w, --wait | Если подключение терпит неудачу, то подождать и повторить попытку. |
| -B, --batch | Выполнить в пакетном режиме. Никаких запросов и никаких ошибок в STDOUT. Устанавливается автоматически при чтении из/записи в канал. Результаты будут выведены в формате с разделением табуляцией. Одна строка результата соответствует одной строке вывода. |
| -I, --help | Справка, эквивалент -\?. |
| -V, --version | Вывести информацию о версии пакета. |

## mysqlaccess

Проверка прав доступа пользователя.

СИНТАКСИС:

mysqlaccess [host] [user] [db] OPTIONS

ОПИСАНИЕ:

Скрипт mysqlaccess используется, чтобы внести в список привилегии пользователя для конкретной базы данных. Это дает информацию, которая может быть полезна при диагностировании проблем с доступом пользователей к данной базе данных.

Скрипт mysqlaccess поддерживает следующие параметры (можно использовать "короткий" одиночный символ или более подробную версию):

|  |  |
| --- | --- |
| -?, --help | Справка. |
| -v, --version | Информация о версии |
| -u, --user=... | Имя пользователя для доступа к базе данных. |
| -p, --password=... | Пароль пользователя, для соединения с сервером MySQL. |
| -h, --host=... | Имя сервера, используется для проверки прав доступа. |
| -d, --db=... | Имя базы данных, используется для проверки прав доступа. |
| -U, --superuser=... | Логин администратора. |
| -P, --spassword=... | Пароль администратора. |
| -b, --brief | Вывести краткие сведения о таблице. |
| --relnotes | Вывести заметки по реализации. |
| --plan | Вывести идеи для будущих реализаций. |
| --howto | Вывести примеры использования `mysqlaccess' |
| --debug=N | Уровень отладки N (0..3) |

Необходимо указать по крайней мере имя пользователя и имя базы данных, которые нужно проверить. Если имя сервера не задано, то подразумевается имя 'localhost'.

## mysqladmin

Выполняет административные функции.

СИНТАКСИС:

mysqladmin [OPTIONS] command command...

ОПИСАНИЕ:

Программа mysqladmin используется, для управления различными аспектами функционирования СУБД MySQL. Допустимы следующие опции (можно использовать "короткий" одиночный символ или более подробную версию):

|  |  |
| --- | --- |
| -\?, --help | Справка. |
| -d, --debug=[options] | Вывести в протокол отладочную информацию. В общем виде 'd:t:o,filename`. |
| -f, --force | Не спрашивать подтверждения при пропуске таблицы. |
| -h, --host=[hostname] | Имя сервера, если не localhost. |
| -i, --sleep=[seconds] | Выполнить команды несколько раз с паузой в [секунд] между ними. |
| -p, --password[password] | Пароль пользователя, для соединения с сервером MySQL. Обратите внимание, что не должно быть пробела между -p и паролем. |
| -u, --user=[user] | Имя пользователя. Если не указано, используется текущий логин. |
| -P, --port=[port] | Порт, для соединения с сервером MySQL. |
| -V, --version | Вывести информацию о версии. |

Кроме этого программа mysqladmin поддерживает следующие команды:

|  |  |
| --- | --- |
| create [имя базы данных] | Создать базу данных. |
| drop [имя базы данных] | Удалить базу данных (вместе со всеми таблицами). |
| processlist | Вывести сведения о работающих потоках MySQL. |
| reload | Перечитать настройки и очистить все кэши. |
| shutdown | Завершить работу СУБД MySQL. Все запущенные MySQL-сессии помечаются как 'killed'. Это означает, что все потоки, которые простаивают в настоящее время, будут закрыты немедленно, а управление потоками будет закрыто, когда они достигнут точек завершения, определенных сервером. Клиенты получат сообщение об ошибке 'mysql server has gone away'. |
| status | Вывести короткое сообщение о статусе сервера. |

## mysqld

Серверная часть пакета MySQL (mysqld).

СИНТАКСИС:

mysqld [OPTIONS]

ОПИСАНИЕ:

Программа mysqld является ядром СУБД MySQL. Она запускается как демон в системе и принимает подключения из клиентских программ, выполняя запросы и возвращая результаты. Она многопоточная, то есть обработает больше чем один запрос одновременно.

Может наблюдаться спад производительности при использовании опции --log-isam, поскольку при этом очень многое пишется в файл протокола. Непериодические изменения, обновления и удаления могут уменьшить производительность на 5-10%. Производительность может сильно понизиться на обновлениях, требующих большого числа одновременных изменений.

Опция --log-isam может использоваться, для копирования базы данных. ISAM журналы могут стать ОЧЕНЬ большими.

Каждая операция UPDATE, DELETE и INSERT имеет заголовок в 13 байтов дополнительно к команде.

Каждая открытая таблица требует девять байтов плюс длина имени файла таблицы. Это необходимо только для таблиц, которые еще не в кэше таблиц.

Кроме того будут иметься одиннадцать байтов дополнительно для любых команд, которые кэшируют/блокируют таблицы. Эти события обычно вызваны внутренними командами mysql. Наиболее часто, когда используется SELECT.

В большинстве случаев нужно выполнять mysqld из скрипта safe\_mysqld .

Программа mysqld поддерживает следующие опции командной строки:

|  |  |
| --- | --- |
| -\?, --help | Справка |
| -#, --debug=[options] | Вывести в протокол отладочную информацию. В общем виде 'd:t:o,filename`. |
| -b, --basedir=[path] | Полное имя каталога в который установлен пакет. |
| -h, --datadir [homedir] | Полное имя каталога в котором хранятся базы данных. |
| -l, --log=[filename] | Имя файла протокола запросов к базам данных и подключений. |
| --log-isam=[filename] | Имя файла протокола изменений isam. |
| -O, --set-variable var=option | Установить переменную. См. ниже. |
| -L, --language=[language] | Значение по умолчанию 'english/'. Может быть так же 'swedish/', 'germany/','french/' или 'czech/'. Текущий список можно посмотреть в подкаталог share/mysql каталога, в который установлен MySQL. |
| -P, --port=[port] | Порт для соединения. |
| -T, --debug-info | Вывести отладочную информацию. |
| --skip-new-routines | Не использовать новые возможности этой версии. |
| --skip-grant-tables | Игнорировать таблицы предоставления доступа. Это дает любому ПОЛНЫЙ доступ ко всем таблицам. |
| --skip-locking | Не использовать блокировку системы. Может дать лучшую эффективность, но не должен использоваться вместе с isamchk. ТО ЕСТЬ, сначала остановите сервер. |
| --skip-name-resolve | Эта опция заставит mysqld принимать адреса IP только тех серверов, которые явно указаны в базе данных привилегий mysql. DNS можно более или менее просто хакнуть, а эта опция позволяет избежать ряда проблем, если прикладная программа требует высокого уровня защиты. |
| --skip-networking | Использовать подключения только через интерфейс localhost. Эта опция не будет работать с MIT потоками. Если подключения к базе данных будут только локальные, то использование этой опции защитит от создания удаленных подключений. |
| --skip-unsafe-select | Пропустить возможно опасные оптимизации. |
| --socket=[socket] | Имя сокет-файла для MySQL. Недоступно при использовании версии MySQL, скомпилированной с MIT потоками.  ПРИМЕР: mysqld --socket=/tmp/mysql.sock |
| -V, --version | Вывести информацию о версии. |

Если опция -h не определена, mysql будет считать, что основной каталог = "/my/data/sql/mysql".

Все базы данных размещены в каталоге [homedir]/[имя базы данных] .

Опция -l должна использоваться осторожно. Когда используется сервер, с большим объемом транзакций, этот файл может стать большим очень быстро. Если не определяется имя logfile, при использовании опции -l mysqld будет писать протокол в файл [homedir]/[hostname].log.

Опция -O позволяет определять значения для следующих параметров:

|  |  |
| --- | --- |
| Имя | Значение по умолчанию |
| back\_log | 5 |
| keybuffer | 1048568 |
| max\_allowed\_packet | 65536 |
| net\_buffer\_length | 8192 |
| max\_connections | 90 |
| table\_cache | 64 |
| recordbuffer | 131072 |
| sortbuffer | 2097144 |
| max\_sort\_length | 1024 |

## mysqldump

Дамп содержания базы данных.

СИНТАКСИС:

mysqldump [OPTIONS] [database [table [field]]]

ОПИСАНИЕ:

Программа mysqldump используется для создания дампа содержания базы данных MySQL. Она пишет инструкции SQL в стандартный вывод. Эти инструкции SQL могут быть переназначены в файл. Можно резервировать базу данных MySQL, используя mysqldump, но при этом Необходимо убедиться, что в этот момент с базой данных не выполняется никаких других действий.

Программа mysqldump поддерживает следующие параметры (можно использовать короткую или подробную версию):

|  |  |
| --- | --- |
| -#, --debug=[options] | Вывести в протокол отладочную информацию. В общем виде 'd:t:o,filename`. |
| -?, --help | Справка. |
| -c, --compleat-insert | Генерируйте полные инструкции insert (не исключая значений, которые соответствуют значениям столбца по умолчанию). |
| -h, --host=[hostname] | Соединиться с сервером hostname. |
| -d, --no-data | Экспорт только схемы информации (исключая данные). |
| -t, --no-create-info | Экспорт только данных, исключая информацию для создания таблицы. Противоположность -d. |
| -p, --password=[password] | Пароль пользователя, для соединения с сервером MySQL. Обратите внимание, что не должно быть пробела между -p и паролем. |
| -q, --quick | Не буферизовать результаты запроса, дамп выдать непосредственно к STDOUT. |
| -u, --user=[username] | Имя пользователя. Если не задано, используется текущий логин. |
| -v, --verbose | Вывести подробную информацию относительно различных стадий выполнения mysqldump. |
| -P, --port=[port] | Порт для связи. |
| -V, --version | Информация о версии. |

Можно направить вывод mysqldump в клиентскую программу MySQL, чтобы копировать базу данных. Необходимо убедиться, что база данных не изменяется в это время, иначе получится противоречивая копию!

## mysqlshow

Показать информацию о сервере, базе данных или таблице.

СИНТАКСИС:

mysqlshow [OPTIONS] [database [table [field]]]

ОПИСАНИЕ:

Программа mysqlshow может использоваться, чтобы показать, с какими базами данных MySQL работает, какие таблицы данная база данных содержит, и какие поля есть в таблице в данной базе данных.

Программа mysqlshow поддерживает следующие параметры:

|  |  |
| --- | --- |
| -#, --debug=[options] | Вывести в протокол отладочную информацию. В общем виде 'd:t:o,filename`. |
| -?, --help | Справка. |
| -h, --host=[hostname] | Связаться с сервером hostname. |
| -k, --key | Вывести ключ(и) для таблиц(ы). |
| -p, --password=[password] | Пароль пользователя, для соединения с сервером MySQL. |
| -u, --user=[username] | Имя пользователя. Если не задано, используется текущий логин. |
| -P, --port=[port] | Порт для связи. |
| -V, --version | Информация о версии. |

Mysqlshow без аргументов покажет все базы данных. Mysqlshow с именем базы данных покажет все таблицы в ней. Mysqlshow с именем базы данных и именем таблицы покажет схему этой таблицы.

Если последний параметр содержит '?' или '\*', то они используются как подстановочные знаки.

## isamchk

Проверка, восстановление, управление и сбор статистики по таблицам MySQL.

СИНТАКСИС:

isamchk [-?adeiqrsvwzIV] [-k #] [-O xxxx=size] [-Si] [-Sr #]

[-O keybuffer=#] [-O readbuffer=#] [-O writebuffer=#]

[-O sortbuffer=#] [-O sort\_key\_blocks=#] files

ОПИСАНИЕ:

Программа isamchk используется для проверки непротиворечивости таблиц и устранения любых проблем, которые это может вызвать. Isamchk может также использоваться, чтобы усилить таблицы в базах данных, которые содержат BLOB или поля символов переменной длины. Это необходимо только, если часто добавляются и удаляются записи из этих таблиц.

Необходимо остановить СУБД MySQL перед запуском isamchk с единственным переключателем -r.

Вообще, нужно использовать переключатель -rq для ремонта таблицы, потому что он выполняет "оперативный" ремонт. Для такого ремонта не нужно временного места в памяти, таким образом он проходит быстро, так как isamchk не копирует файл данных.

Перед началом работы, надо перейти в каталог с таблицей, которая нуждается в проверке и/или ремонте. В общем случае, это $DATADIR/DBNAME.

Опции программы isamchk:

|  |  |
| --- | --- |
| -# | Вывести в протокол отладочную информацию. В общем виде 'd:t:o,filename`. |
| -? | Справка. |
| -a | Анализ. Используется для для оптимизации размещения таблиц. |
| -d | Информация о таблице. |
| -e | Расширенная проверка. При запуске с этой опцией надо остановить демон mysqld. |
| -f | Принудительно создать временный файл. Эта опция вызовет 'isamchk -r', если таблица разрушена. |
| -k# | Используемые ключи. |
| -i | Дополнительная информация. |
| -q | Быстрая проверка. |
| -r[o] | Восстановление, Эта опция также уплотнит внутренние индексы B-Tree, используемые MySQL. Используйте эту опцию, чтобы уменьшить место, занимаемое таблицей, за счет удаления неиспользуемых индексов. |
| -s | Работать молча. |
| -u | Распаковать упакованный файл. |
| -v | Подробный вывод. Используется вместе с другими параметрами, чтобы получить большее количество информации. Можно задать большее количество символов v для повышения объема выдаваемой информации (например, vv). |
| -w | Ждать, если таблица заблокирована. |
| -I | Справка. |
| -S[ir]# | Сортировать индекс/записи по ключу #. Эта опция оптимизирует размещение данных в таблице. |
| -V | Информация о версии. |
| -O var=# | Опция оптимизации var=#[k][m] |

Можно использовать вторую опцию '-q', чтобы использовать старый файл данных. -ro использует старый метод восстановления: медленнее, но надежнее -r. -r может исправить почти все ошибки, кроме уникальных ключей, которые оказались все же не уникальными. Если -f задан при проверке таблицы, то таблицы с ошибками будут автоматически исправлены.

ПРИМЕРЫ:

|  |  |
| --- | --- |
| isamchk -r [table\_name] | Убрать все дырки, которые сформировались из-за использования BLOBS или VARCHARS. Также установить какие еще есть проблемы. |
| isamchk -ei [table\_name] | Проверить таблицу и отобразить статистику. |
| isamchk [table\_name] | Найти большинство ошибок. |
| isamchk -rq [table\_name] | Обновить только индексный файл. Быстро, но не исправляет ошибки в файле данных. |
| isamchk -d -v [table\_name] | Описание таблицы. |
| isamchk -rq -Si [table\_name] | Полная оптимизация таблицы [table\_name]. |

## isamlog

Получение статистики, и использование isam-журналов.

СИНТАКСИС:

isamlog [-?iruvIV] [-c #] [-f #] [-p #] [-F filepath/] [-o #]

[-R file recordpos] [-w write\_file] [log-filename]

ОПИСАНИЕ:

Эта программа используется вместе с опцией --log-isam=file\_name демона mysqld.

Файл file\_name будет хранить протокол всех изменений для всех таблиц. Программа isamlog может быть использована для получения информации об этом файле и обновить все таблицы и базы данных.

Чтобы восстановить базу данных, нужна копия, которая содержит вашу базу данных до того, как mysqld сгенерировал ISAM журнал, или полный набор ISAM журналов с начала работы вашей базы данных.

|  |  |
| --- | --- |
| -? or -I | Справка. |
| -V | Информация о версии. |
| -c # | Выполнить только # команд. |
| -f # | Максимальное число открытых файлов. Если откроется больше чем # файлов, то лишние будут автоматически закрыты и вновь открыты по мере необходимости. |
| -F [path] | Файл протокола ISAM содержит полный путь к таблицам. Можно использовать эту опцию, чтобы отменить сохраненный путь.  ПРИМЕР:  -F '/var/mirror/' заставит isamlog работать с файлами в /var/mirror, а не с теми, которые указаны в сохраненном пути. |
| -i | Вывести более детальную информацию. |
| -o # | Начать со смещения #. |
| -p # | Удалить # компонент из пути. |
| -r | Игнорировать все ошибки, с которыми сталкиваемся в течение модификаций. |
| -R | Указать позицию в файле ISAM, о которой надо выдать сведения.  ПРИМЕР:  isamlog -R /usr/local/data/mysql/user.ISM 1234  Выведет все изменения в таблице /usr/local/data/mysql/user, которые найдены в позиции 1234. |
| -u | Пытаться модифицировать все таблицы. Остановиться, если встречена ошибка. |
| -v | Подробный вывод. Можно задать большее количество символов v для повышения объема выдаваемой информации (например, vv). |
| -w [filename] | Записать все записи вставленные или измененные опцией -R в [filename]. Эта опция может использоваться, чтобы получить двоичный файл всех вставок и модификаций конкретной таблицы для использования в отладке. |

## safe\_mysqld

Скрипт для запуска демона mysqld.

СИНТАКСИС:

safe\_mysqld [options to mysqld]

ОПИСАНИЕ:

Этот скрипт обычно выполняется при начальной загрузке, чтобы запустить mysqld. Если необходимо, чтобы DATADIR был помещен в иерархии каталога MySQL, что и mysqld, нужно подправить DATADIR в этом скрипте.

## 2.2. Дополнительные команды

## comp\_err

Генерирует файлы сообщений об ошибках на национальном языке.

СИНТАКСИС:

comp\_err [-?] [-I] [-V] fromfile[s] tofile

ОПИСАНИЕ:

Эта программа используется для компиляции текстовых файлов которые содержат соответствие между кодами ошибок mysql и текстовыми сообщениями об этих ошибках в формат, который понимает mysql. Это нужно для изменения уже существующего файла сообщений или генерации такого файла для другого языка.

ПРИМЕР:

comp\_err share/english/errmsg.txt share/english/errmsg.sys

## msql2mysql

Выполняет начальное преобразование mSQL скрипта в MySQL.

СИНТАКСИС:

msql2mysql [filename]

ОПИСАНИЕ:

Скрипт msql2mysql используется как первый шаг при портировании msql скриптов в MySQL.

Программа msql2mysql - фактически довольно простой скрипт, который использует команду replace, которая поставляется с СУБД MySQL, чтобы заменить mSQL обращения к функции на их MySQL эквиваленты. Меняются только имена, дополнительные параметры не добавляются.

## mysqlbug

Генерирует отчет об ошибках.

СИНТАКСИС:

mysqlbug [address]

ОПИСАНИЕ:

Этот скрипт используется при сообщении об ошибках в MySQL. Он генерирует форму, которую для заполнения помещает в любой текстовый редактор, указанный в переменной $VISUAL (Vi по умолчанию). Форма будет содержать автоматически сгенерированную информацию относительно вашей системы, включая версии OS и MySQL и архитектуру. При окончании, форма будет отправлена по почте в список рассылки MySQL.

## perror

Выводит короткое текстовое объяснение числового кода ошибки.

СИНТАКСИС:

$ perror [-?vIV] [errorcodes]

ОПИСАНИЕ:

Программа perror предоставляет короткие текстовые объяснения числовых кодов ошибок, возвращаемых системой или СУБД MySQL.

Программа perror поддерживает опции:

|  |  |
| --- | --- |
| -? or -I | Справка. |
| -v | Подробная информация. |
| -V | Информация о версии. |

## replace

СИНТАКСИС:

replace [-?svIV] from to from to ... -- [files]

ОПИСАНИЕ:

Программа program используется скриптом msql2mysql. Replace может быть использована для замены строки на месте в файле или получать строку через канал (пайп), производить замену и выводить результат в STDOUT.

Допустимы следующие опции:

|  |  |
| --- | --- |
| -? | Справка. |
| -s | Работать молча (никаких статусных сообщений). |
| -v | Подробная информация о ходе работы (выводить дополнительные сообщения). |

ПРИМЕРЫ:

replace Apple Orange somefile

Заменит все вхождения Apple на Orange в файле somefile.

cat INFILE | replace Apple Orange Blimp Train > OUTFILE

Все вхождения Apple в файле INFILE будут заменены на Orange и результат выведен в файл OUTFILE. В то же время, все вхождения Blimp в файле INFILE будут заменены на Train и тоже выведены в файл OUTFILE. Таким образом, можно заменять более чем одно выражение за один проход.

Можно использовать специальные символы во входных строках:

|  |  |
| --- | --- |
| \^ | Соответствует началу строки. |
| \$ | Соответствует концу строки. |
| \b | Соответствует пробелу. Может появляться или отдельно, или в начале, или в конце лексемы. Если \b используется в конце лексемы, следующие замены начнутся после пробела. \b соответствует только пробелу. |

Replace простая и очень полезная утилита с большими потенциальными возможностями использования в MySQL.

## which1

Найти команду.

СИНТАКСИС:

which1 [cmd]

ОПИСАНИЕ:

Эта команда, вероятно, полезна только на системах, которые не имеют, команды which, встроенной в оболочку, или доступной как команда. Выводит только первую найденную команду.

## zap

Послать сигнал процессу, который соответствует образцу.

СИНТАКСИС:

zap [-signal] [-?Ift] pattern

ОПИСАНИЕ:

По умолчанию zap запросит подтверждение для каждого подходящего процесса.

Программа zap поддерживает опции:

|  |  |
| --- | --- |
| -I or -? | Справка. |
| -f | Послать сигнал без запроса подтверждения. |
| -t | Вывести список процессов, соответствующих образцу, но не сигнал им не посылать. |

В качестве сигнала допускается любой сигнал системы unix. По умолчанию сигнал = 9 (завершить процесс).

## 3. Настройка mysqld (серверная часть пакета MySQL)

При указании опции -O, отдельно (без дополнительных параметров) можно получить список всех переменных и их текущих значений. Это может выглядеть примерно так:

$ mysqld -O

Возможные переменные в опции -O:

|  |  |
| --- | --- |
| Имя | Значение по умолчанию |
| back\_log | Определите размер очереди для входящих tcp/ip подключений.. Некоторые ОС могут иметь максимальное значение этой переменной 128 или 256. |
| **keybuffer** | Размер кэш-буфера, для хранения всех недавно использованных ключей. Большой буфер дает самую лучшую эффективность. С переключателем -Sl распределен только один буфер. |
| **max\_allowed\_packet** | Буфер подключений сервера может быть изменен до этого значения, если пользователь дает длинную команду. Начальный буфер = 'net\_buffer\_length'. На каждое подключение выделяется один буфер. |
| **net\_buffer\_length** | Начальный размер буфера подключений. На каждое подключение выделяется один буфер. |
| **max\_connections** | Максимальное число подключений, которые mysqld может иметь открытыми в одно и то же время. |
| **table\_cache** | Максимальное число таблиц сохраняемых открытыми на сервере. Таблицы хранятся открытыми, для ускорения запроса к часто используемым таблицам. Однако, каждая открытая таблица требует много памяти. |
| **recordbuffer** | Размер кэш-буфера для хранения прочитанных записей. На каждое подключение выделяется один буфер. |
| **sortbuffer** | Размер буфера, используемого при сортировке. На каждое подключение выделяется один буфер. |
| **max\_sort\_length** | Максимальное время для сортировки? |

### Полезные замечания

* Все потоки совместно используют ту же самую основную память.
* MySQL в настоящее время не использует memmap. Это может изменяться.
* Можно определять размер буфера для ключей при старте mysqld. Буфер ключей будет кэшировать все ключи во всех таблицах В ПОРЯДКЕ ПОСТУПЛЕНИЯ запросов (переменная keybuffer)
* Каждое подключение использует некоторую часть потока, стек и буфер подключений (переменная net\_buffer\_length).
* Каждый запрос, делающий последовательный просмотр записей в таблице, распределяет буфер чтений (переменная recordbuffer).
* Каждый запрос, выполняющий сортировку, распределяет sortbuffer и один или два временных файла. Максимально необходимое дисковое пространство = (sort\_key\_length + sizeof(long))\*2.
* Все объединения выполняются за один проход, и результат сохраняется во временной таблице в памяти. Выполнение полного объединения и сортировки на двух больших таблицах может потребовать очень большого объема памяти.
* Индексные файлы открываются один раз. Файлы данных открываются один раз для каждого параллельного потока. Каждый параллельный поток имеет полную таблицу struct и поле struct плюс память для трех полных строк столбцов. BLOB требует еще 5-8 байт.
* Если таблица имеет BLOB, буфер будет распределен, чтобы читать данные BLOB. Этот буфер будет расти по мере необходимости, до размеров самого большого BLOB в таблице.
* Когда таблица используется, она сохраняется в кэше. Этот кэш будет загружен и опорожнен В ПОРЯДКЕ ПОСТУПЛЕНИЯ запросов (FIFO). По умолчанию mysqld будет кэшировать 64 таблицы. Если много потоков обращаются к одной и той же таблице одновременно, будет иметься одна запись на поток. ТО ЕСТЬ, если два потока одновременно обращаются к той же самой таблице, будет две записи в кэше.
* Команда mysqladmin reload закрывает все таблицы, которые не используются, и отмечает все используемые таблицы, которые будут закрыты, когда работающие потоки завершатся. Это позволяет эффективно использовать память, своевременно ее освобождая.

4. Возможности языка Perl для работы с MySQL

сервером.

При написании программы к данной работе выбран язык Perl, т. к. сегодня MySQL более широкое распространение получил на Web-серверах. А Perl как нельзя лучше других языков подходит для написания CGI-приложений. Взаимодействие сценария на языке Perl с MySQL можно изобразить схемой:

Сценарий Perl–>DBI –>DBD::MySQL–>Cервер MySQL.

DBI обеспечивает единый интерфейс взаимодействия с различными системами управления базами данных. А DBD связывает этот интерфейс непосредственно с конкретной СУБД. В нашем случае это mysql. Для того, чтобы программа на Perl могла общаться и работать с базой данных, необходимо подключить модуль DBI и драйвер DBD. Это выполняется следующими операторами:

use DBI; use DBD::mysql;

Для того чтобы выполнять SQL запросы к базе данных, нужно установить логическое соединение с MySQL. Это выполняется при помощи метода объекта DBI. При этом он возвращает объект (дескриптор) соединения, который будет использоваться для непосредственного общения с базой данных.

$dbh = DBI->connect('DBI:mysql:dbname= GRYADKA; host= 127.0.0.1; port=3306','','',0);

Вызывая метод connect, ему передаются некоторые параметры. Первый параметр определяет свойства соединения.

Эта строка имеет следующую структуру:

"dbi:<имя DBD>:dbname= <имя базы данных>; host=<имя хоста>; port=<номер порта>"

Далее в методе connect определяются имя пользователя, пароль и флаги соединения.

Существует два способа работы с базами данных MySQL. Первый — так называемый механизм курсоров, а второй, соответственно, без использования таковых.

Курсоры — это объекты Perl, которые обеспечивают последовательный доступ к результатам запросов.

Каждый курсор отвечает за закрепленный за ним запрос.

$cur = $dbh->prepare('select \* from MAIN;');

$cur — это курсор, который для начала следует выполнить, используя метод execute:

$cur->execute;

Теперь, после выполнения запроса, чтобы получить результат запроса, используется метод fetchrow\_array. Он поочередно возвращает массив полей результативной виртуальной таблицы.

Очень часто бывает необходимо подготовить запрос и после, при его выполнении, передавать ему различные данные. Это довольно просто решается при помощи механизма placeholders:

$cur = $dbh->prepare('insert into MAIN values(?,?)');

где вопросительные знаки следует заменить на значения переменных, переданных в качестве параметров метода execute:

$a00 = 2; $a01 = 'second record'; $cur->execute($a00, $a01);

Для того чтобы закрыть курсор, необходимо выполнить метод finish.

$cur->finish;

Курсоры очень удобный момент в технологии DBI. Однако можно обойтись и без них. Существует множество методов, которые вполне неплохо заменяют курсоры:

selectrow\_array — возвращает одну строку запроса в виде массива;

selectall\_arrayref — возвращает весь ответ сервера в виде массива, ссылка на массивы;

do — выполняет запрос, ничего не возвращая (очень удобно при работе с insert, update и пр.)

Для того чтобы разорвать соединение с MySQL, необходимо выполнить метод disconnect.

$dbh->disconnect;

**Заключение.**

В процессе написания данной работы были выяснены следующие свойства MySQL сервера:

* Многопоточность. Поддержка нескольких одновременных запросов.
* Оптимизация связей с присоединением многих данных за один проход.
* Записи фиксированной и переменной длины.
* Гибкая система привилегий и паролей.
* До 16 ключей в таблице. Каждый ключ может иметь до 15 полей.
* Поддержка ключевых полей и специальных полей в операторе CREATE.
* Поддержка чисел длинной от 1 до 4 байт (ints, float, double, fixed), строк переменной длины и меток времени.
* Интерфейс с языками C и perl.
* Основанная на потоках, быстрая система памяти.
* Утилита проверки и ремонта таблицы (isamchk).
* Все операции работы со строками не обращают внимания на регистр символов в обрабатываемых строках.
* Псевдонимы применимы как к таблицам, так и к отдельным колонкам в таблице.
* Все поля имеют значение по умолчанию. INSERT можно использовать на любом подмножестве полей.
* Легкость управления таблицей, включая добавление и удаление ключей и полей.

Итак, популярность MySQL сервера определяется не только тем, что пакет распространяется бесплатно и имеет высокую производительность, но ещё и тем, что имеет множество вышеперечисленных достоинств, многих из которых нет у других СУБД.

И виртуальный магазин, и сервер знакомств, и всевозможные сайты, предоставляющие платные услуги, — все нуждаются в возможности обрабатывать и хранить большие объемы информации. MySQL предоставляет всем эту возможность совершенно бесплатно. Это действительно одна из самых мощных, быстрых и надежных СУБД.

В ходе выполнения курсовой работы была написана программа, которая представляет собой Интернет-каталог. Она иллюстрирует возможности MySQL сервера, возможности интеграции MySQL c сетью Internet, что на сегодня весьма актуально. Созданная программа может использоваться в качестве Интернет-каталога в Интернет-магазине.

###### **Список литературы**

1. М. Каба. MySQL и Perl – СПб.: Питер, 2001.

###### А. Паутов. Документация по MySQL.

1. М. Грубер. Понимание SQL.
2. Материалы интернет-сайтов:

www.mysql.com

www.infocity.kiev.ua