Для просмотра разрешений на защищаемые объекты можно использовать такие функции:

**Sys.fn\_built\_permissions()** – возвращает описание иерархии встроенных разрешений сервера. Ее синтаксис:

Sys.fn\_built\_permissions(

Default|null|empty\_string| - возвращается полный список встроенных разрешений

|<securable\_class>) - возвращаются разрешения для определенного класса

Данная функция доступна всем ролям в БД.

Примеры:

select \* from sys.fn\_built\_permissions(default);

select \* from sys.fn\_built\_permissions(login);

Также можно получить список классов объектов, на которые предоставляется определенное разрешение.

Select \* from sys.fn\_built\_permissions(default) where permission\_name = ‘select’

**Has\_perms\_by\_name()** – данная функция позволяет просмотреть действующие разрешения для данного объекта. Действующие разрешения включают следующие разрешения, если какие-то из них не были явно отклонены:

1. Предоставленные непосредственно пользователем
2. Вытекающие из разрешения более высокого уровня, имеющегося у пользователя
3. Предоставленные той роли, членом которой является пользователь
4. Принадлежащие роли, членом которой является пользователь

Выполнить ее может каждый пользователь.

Синтаксик:

Has\_perms\_by\_name(

Securable, - имя защищаемого объекта, null, если сервер

securable\_class, - имя класса защищаемых объектов, null, если защищаемым объектом является сервер

permission\_name – не null, имя разрешения, наличие которого мы хотим проверить. Также можно использовать имя разрешения ‘any’ в качестве шаблона, чтобы определить, обладает ли защищаемый объект

[, sub-securable]

[, sub\_securable\_class]

); - true, если объект имеет разрешение, false, если нет. Null, если неудачно завершился запрос.

Пример: has\_perms\_by\_name(‘имя бд’, database, ‘any’)

Для просмотра разрешений определенного пользователя используется функция execute as user = ‘Валя’.

Также могут быть просмотрены разрешения на содержащиеся в схеме объекты, такие как таблицы и представления.

Для просмотра, на какие таблицы текущий пользователь имеет разрешение select, нужно выполнить:

Select has\_perms\_by\_name(name, ‘object’, ‘select’), \* from sys.tables;

Имеет ли текущий пользователь разрешения на таблицу 1 бд

Режимы аутентификации.

Модель безопасности SQL Server имеет два режима аутентификации:

1. Только аутентификация Windows, когда база доступна только внутри организации
2. Смешанный режим (можно подключаться удаленно)

Эти режимы настраиваются на уровне сервера и применяются ко всем базам на сервере.

В аутентификации Windows можно использовать учетные записи пользователей и групп, доступные домену Windows.

Благодаря этому пользователи получают доступ к базе без отдельной учетной записи SQL Server и пароля. Они также не должны следить за множеством своих паролей и при изменении доменного пароля им не потребуется изменять пароль от SQL Server.

SQL Server таких пользователей идентифицирует автоматически или основываясь на имени учетной записи или его членстве в группе.

Если пользователю или группе предоставлен доступ к базе, они получают его автоматически. Некоторые локальные учетные записи получают доступ к SQL Server по умолчанию. Эта учетная запись – локальная группа администраторов.

В смешанном режиме используется аутентификация Windows и учетная запись SQL Server. Приложение, получающее доступ к SQL Server из интернета, можно настроить таким образом, чтобы они автоматически использовали определенную учетную запись SQL Server или запрашивали у пользователя имя учетной записи SQL Server и пароль.

Все серверы SQL Server обладают встроенной учетной записью SA, а также в зависимости от конфигурации экземпляра сервера учетные записи NetworkService и System. У всех баз данных есть встроенные пользователи DBO, Guest, Information\_Schema, sys.

Учетные записи и записи специального назначения.

Учетным записям можно дать различные уровни доступа посредством:

1. Ролей, которым они принадлежат
2. Разрешения доступа к базе данных
3. Предоставление или отклонение разрешений на объект

Существует два типа учетных записей: доменные и учетные записи SQL Server. Доменная создается на основании учетных записей доменов Windows, а вторые создаются посредством указания учетной записи и пароля.

Группа администраторы и ее учетные записи в SQL Server по умолчанию включается в роль сервера сисадмина.

Если сервер настроен в качестве сервера отчетов, будет создана учетная запись для учетки NetworkService, которая станет членом специальной роли БД RSExecRole в базах MSDB, RecordServer и ReportServerTempDB.

Гости

Гость является специальным пользователем, которого можно добавить к любой базе и предоставить доступ к SQL Server всем, кто имеет эту учетную запись.

Пользователя Guest можно удалить из всех БД за исключением БД Master и TempDB. Перед применением пользователя Guest следует учесть, что он:

1. Является членом роли сервера Public и наследует разрешения, предоставленные этой роли.
2. Должен существовать в базе данных прежде, чем кто-либо может получить к ней доступ в качестве гостя.
3. Используется, когда учетная запись получила доступ к SQL Server, но не имеет доступа к базе данных через ассоциированного пользователя с учетной записью пользователя БД.

Работа с пользователем DBO

Пользователю DBO неявно разрешены все действия над базой данных, а также он может передавать их другим пользователем. Поскольку члены роли сервера Sysadmin автоматически сопоставляются с пользователем DBO, все учетные записи этой роли могут выполнять те же задачи.

INFORMATION\_SCHEMA или sys

Позволяет просматривать метаданные.

Разрешения

Разрешения – права пользователя для проведения тех или иных действий на сервере или в БД.

В SQL Server используются три типа разрешений:

1. На объект
2. На инструкции
3. Неявные разрешения

Роли

Роли – группа пользователей, обладающих теми или иными правами.

В SQL Server доступно два вида ролей:

1. Роли сервера
2. Роли базы данных

Роли сервера устанавливаются на уровне сервера и являются предопределенными.

Существуют следующие роли сервера:

1. Bulkadmin

Для операций массового копирования в БД.

Может выполнять (BulkInsert)

1. DBcreator

Удалять и воссоздавать данные из резервных копий.

(ALTER/CREATE/DROP/EXTEND Database, restore database, restore log, sp\_renameDB)

1. Diskadmin

Управляет файлами.

(Disk init, sp\_addumpdevice, sp\_diskdefault, sp\_dropdevice)

1. Processadmin

Отключает БД.

1. Securityadmin

Эти пользователи управляют учетными записями, создают разрешения и читают журнал ошибок, переустанавливают пароли, управляют разрешениями сервера на уровне баз и серверов.

(sp\_addlinkedsrvlogin, addlogin, defaultdb, defaultlanguage, denylogin…)

1. Serveradmin

Пользователь, который устанавливает параметры конфигурации сервера и завершает работу сервера.

1. Setupadmin

Пользователи, управляющие связанными серверами и контролирующие процедуру запуска.

1. Sysadmin

Могут всё и вся!

Роли уровня баз данных.

На уровне баз данных поддерживается три типа ролей:

1. Стандартные пользовательские роли баз данных
2. Пользовательские роли приложений
3. Встроенные роли баз данных

Встроенные роли и их права

1. Public

По умолчанию только просматривают данные.

1. DB\_accessadmin

Роль для пользователей, которым необходима возможность добавлять или удалять других пользователей в БД.

1. BackupOperator

Выполняет резервное копирование БД.

1. DB\_datareader

Для тех, кому нужно просматривать данные в базе.

1. Db\_datawriter

Могут записывать данные

1. Db\_ddladmin

Для пользователей, выполняющих задачи, для которых необходимо использование языка определения данных.

1. Db\_denydatareader

Для тех кому нельзя читать

1. Db\_denydatawriter

Для тех, кому нельзя записывать данные

1. Db\_securityadmin

Для пользователей, управляющих разрешениями на объекты и правами владения на объекты.

1. DB\_owner

Для тех, кому нужен полный контроль над всеми аспектами функционирования бд.

SQL операторы для работы с привилегиями

GRANT – выдать привилегии.

GRANT привилегии ON объект TO пользователь

REVOKE - отозвать привилегии

REVOKE привилегия

ON объект

FROM пользователь

DENY – запрещать выдавать привилегии

DENY привилегия ON объект TO пользователь

При отмене привилегии запросом REVOKE необходимо учитывать ограничения и правила:

1. Привилегии могут быть удалены только у тех пользователей, которым их предоставили.
2. Привилегии, присвоенные другими пользователями, не затрагиваются.
3. Удаление привилегии для пользователя А, которому было дано право передавать ее, автоматически удаляет эту привилегию для всех пользователей, которым он выдал это право.
4. Привилегии, переданные для общего пользования, могут быть отобраны только у public, а не у конкретного пользователя или объекта.

Выдать привилегии просмотра, изменения и удаления пользователю user1 в отношении таблицы 1 с правом передачи этой привилегии другим пользователям:

GRANT INSERT, UPDATE, SELECT

On table1

To user1, user2

With grant option;

Пусть на сервере зарегистрирован пользователь Иванов, и администратор выдает ему права манипулирования и просмотра данных с правом передачи своих прав. Он передал права петрову с правом передачи. Петров выдал право просмотра сидорову. Иванов выдал сидорову права просмотра таблицы один а петров отобрал а сидоров отнял права у петрова.

Потенциальные опасности:

1. Похищение и фальсификация данных
2. Утрата конфиденциальности (нарушение тайны)
3. Нарушение неприкосновенности личных данных
4. Утрата целостности
5. Потеря доступности

Любая угроза функционированию информационной системы должна рассматриваться как ситуация, направленная на катастрофические результаты работы предприятия.

В следующей таблице представлены возможные опасности для ИС.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Опасность | Похищение и фальсификация данных | Утрата конфиденциальности | Нарушение неприкосновенности данных | Утрата целостности | Утрата доступности |
| Использование чужих прав доступа | + | + | + | - | - |
| Несанкц. Изменение или копирование данных | + | +- | +- | + | - |
| Изменение программ | + | - | - | + | + |
| Непродуманность организационной инструкции, допускающее смешение конфиденциальной и неконфиденциальной информации. | + | + | + | - | - |
| Несанкц. Подключение к кабельным сетям | + | + | + | - | - |
| Несанкц. Ввод данных | + | + | + | + | - |
| Отказ аппаратных средства | - | +/- | +/- | + | + |
| Электронные помехи | - | - | - | +/- | + |
| Физическое повреждение оборудования и коммуникации | - | - | - | + | + |

Проблемы обеспечения безопасности баз данных можно подразделить на две категории:

1. Технологическая
2. Организационная

Технологические методы защиты информации.

Технологическую защиту можно свести к следующим аспектам:

1. Обеспечение непрерывности и корректности функционирования систем, от которых зависит безопасность людей и экологическая обстановка.
2. Обеспечение защиты имущественных прав граждан, предприятий и государства в соответствии с правилами гражданского, административного и хозяйственного кодекса, включая защиту секретов и интеллектуальной собственности.
3. Обеспечение защиты гражданских прав и свобод, гарантированных действующим законодательством, включая право на доступ к информации.

Обеспечение защиты информации включает в себя:

1. Разработку показателей, характеризующих технологическую безопасность ИС.
2. Разработку требований к архитектуре БД.
3. Наличие трудовых и материальных ресурсов.
4. Разработку организационных мероприятий для исключения влияния внутренних и внешних дестабилизирующих факторов.
5. Разработка методов и средств, предотвращающих влияние дефектов программ и данных, в том числе разработку компьютерных экспертных систем оценки качества программных продуктов).

Показатели надежности - информация, которая учитывает все возникающие отказы при эксплуатации БД.

Характеристика степени безопасности – информация, учитывающая только отказы, повлиявшие на безопасность системы.

Работоспособное состояние - …

Показатели надежности БД оцениваются по следующим критериям:

1. Способность информационной системы к безотказной работе при наличии сбоев и отказов программных и аппаратных средств.
2. Восстанавливаемость – время и полнота восстановления функционирования программ после перезапуска в случае сбоев и отказов.
3. Коэффициент готовности – степень вероятности восстановления системы в любой произвольный момент.

Основное требование к архитектуре информационных систем – архитектура должна быть достаточно гибкой и допускать наращивание ресурсов ИС без коренных структурных изменений, например за счет развития аппаратных и программных средств.

Средство генерации тестов, разрабатываемых на этапе тестирования, предназначены для подготовки исходных данных при проверке различных режимов функционирования информационной системы. Минимальный состав средств имитации может передаваться пользователям для контроля рабочих версий ИС в реальном времени и входить в комплект поставки каждой пользовательской версии.

Средства оперативного встроенного контроля процесса исполнения программ должны непрерывно контролировать промежуточные и результирующие или включаться по запросу при обнаружении сомнительных результатов.

Непредумышленные угрозы и защищаемые объекты.

1. Динамический вычислительный процесс обработки данных, автоматизированная подготовка решений и выработки управляющих воздействий.
2. Информация, накопленная в базе данных.
3. Объектный код программ, исполняемых вычислительными средствами в процессе функционирования ИС.
4. Информация, выдаваемая потребителем и на исполнительные механизмы.

Непредумышленные дестабилизирующие факторы можно подразделить на внешние и внутренние.

1. Ошибки в программировании.
2. Алгоритмические ошибки проектирования и эксплуатации БД.
3. Системные ошибки при разработке технического задания на разработку удаленных баз данных.
4. Недостаточная эффективность используемых методов и средств оперативной защиты программ и данных.

Внешние источники угроз:

1. Ошибки оперативного и обслуживающего персонала в процессе эксплуатации баз данных.
2. Сбои и отказы аппаратуры.
3. Искажение информации в каналах телекоммуникации.
4. Выход изменений состава и конфигурации информационной системы за пределы проверенной при испытаниях или сертификации.

Современные технологии разработки баз данных РУБД определяют следующие методы и средства, которые позволяют с максимальным эффектом обеспечить технологическую безопасность ИС:

1. Разработка баз данных в полном соответствии с методологией их проектирования.
2. Систематическое тестирование программ управления базами данных на всех этапах ЖЦ.
3. Применение экспертных систем в процессе сертификации СУБД и сдачи их в эксплуатацию.
4. Применение программно-аппаратных методов защиты информации в критических ситуациях.
5. Физическое уничтожение информации в критических ситуациях.

Методы защиты информации:

1. Программно-аппаратные
   1. Авторизация
   2. Применение представлений
   3. Резервное копирование и восстановление
   4. Шифрование и создание массивов независимых дисковых накопителей.
2. Методы физического уничтожения информации в критических ситуациях

Применение представлений

Представление является виртуальным отношением (динамической таблицей) базы данных, которое создается в результате запроса пользователя и доступно только самому пользователю. Механизм представлений служит достаточно эффективным средством защиты баз данных от несанкционированного доступа, поскольку доступен только авторам представления.

Шифрование – процесс преобразования данных с применением специального алгоритма, в результате чего данные становятся недоступными для считывания без ключа дешифрования.

Технологии шифрования подразделяются на 2 категории:

1. Необратимое
2. Обратимое

Для организации защищенной передачи данных по незащищенным сетям использует системы шифрования, включающие в себя следующие компоненты:

1. Алгоритм шифрования
2. Алгоритм дешифрования
3. Ключ шифрования
4. Ключ дешифрования

Некоторые системы шифрования называются симметричными – используют один и тот же ключ для шифрования и дешифрования. Одной из наиболее распространенной является система DES – Data Encryption System, использующая стандартные методы шифрования. В этой системе шифрования и дешифрования используется один и тот же ключ, который должен храниться в секрете, хотя сам алгоритм предусматривает преобразование каждого 64 – битового блока обычного текста с использованием 56 битного ключа шифрования не является секретом.

Создание массивов независимых дисковых накопителей

К аппаратному обеспечению, на котором эксплуатируется СУБД, предъявляются высокие требования по отказоустойчивости. То есть СУБД должна продолжать работать даже при отказе отдельных аппаратных компонентов. Для этого необходимо иметь избыточные компоненты, которые должны быть объединены в систему для сохранения работоспособности ИС.

К числу аппаратных средств, к которым предъявляются высокие требования по отказоустойчивости, относятся:

1. Дисковый накопитель
2. Дисковый контроллер
3. Процессоры
4. Источники питания
5. Вентилятор охлаждения

Дисковые накопители являются наиболее уязвимыми. Их характеризуют самыми низкими показателями непрерывной работы между отказами. Одним из методов повышения отказоустойчивости дисковых накопителей является применение RAID технологий. RAID массив представляет собой массив дисковых накопителей большого объема, состоящий из нескольких независимых дисков, совместное функционирование которых обеспечивает повышение их надежности и производительности.

Повышение производительности осуществляется вследствие полосового распределения данных на диске. То есть распределение данных по сегментам, представляющим собой области дискового пространства одного размера, называемого единицей полосового распределения. Сегменты распределяются по нескольким дискам и обеспечивают прозрачный доступ к данным. Полосовое распределение обеспечивает повышение производительности ввода и вывода, поскольку операция выполняется на разных дисках одновременно. Повышенная надежность RAID массива обеспечивается также благодаря дублированию данных в виде зеркальных копий, которые и создают избыточную информацию.

В RAID массивах используются различные методы повышения производительности и надежности, получившие название «Уровни RAID»:

1. RAID 0 – неизбыточный массив

На нем не применяется дублирование данных, поэтому обеспечивается наивысшая производительность. Полосовое распределение – на уровне дисковых блоков.

1. RAID 1 – массив с зеркальным отображением

На этом уровне создаются 2 идентичные копии на разных дисках. Этот вариант самый дорогой.

1. RAID 0 + 1 – неизбыточный массив с зеркальным отображением
2. RAID 5 – массив, обеспечивающий контроль четности и распределением информации контроля четности.

На этом уровне информация контроля четности применяется в качестве избыточной и обеспечивающей восстановление первоначальных данных по такому же принципу, как и в массиве RAID 3. При этом данные контроля четности распределяются с помощью полосового распределения по всем дискам так же, как происходит распределение исходных данных, что позволяет устранить узкое место, возникающее при хранении всей информации контроля четности на одном диске.

1. RAID 6. Массив с избыточностью P + Q.

Этот уровень аналогичен уровню 5, но предусматривает хранение дополнительных избыточных данных для защиты от отказа сразу нескольких дисков. При этом вместо информации контроля четности используется код исправления ошибок.

Дисковое хранилище с системным уничтожением данных

Для мгновенного удаления информации с магнитных носителей используют отдельно стоящую или встроенную в корпус компьютера технику. В настоящее время оптимальным подходом, обеспечивающим уничтожение информации без уничтожения носителя, является использование физических методов, заключающихся в перестройке структуры магнитного материала рабочих поверхностей носителя. Для уничтожения информации на магнитной пластине накопителя на жестком магнитном диске необходимо устранить неоднородность вектора намагниченности в участках его рабочей поверхности, несущих информацию о предшествующих записях. Изменение структуры поля вектора намагниченности магнитного материала может быть выполнено несколькими принципиально различными способами:

1. Быстрое нагревание рабочего слоя носителя до точки потери намагниченности носителя (точка Кюри).
2. Размагничивание рабочей поверхности носителя.
3. Намагничивание рабочей поверхности до максимума.
4. Комбинированный метод: нагревание и размагничивание или нагревание и намагничивание.

Первый способ основывается на одном из важных эффектов магнетизма: при нагревании ферромагнетика до температуры, превышающей точку Кюри, интенсивность теплового движения атома становится достаточной для разрушения его самопроизвольной намагниченности и он становится парамагнетиком. Следовательно, при такой температуре материал теряет свою остаточную намагниченность и все следы записанной информации гарантированно уничтожаются. Температура, соответствующая точке Кюри, для большинства составляет сотню градусов. При этом надо учитывать, что каждый производитель накопителя на жестком магнитном диске держит в секрете материалы основы и состав ферромагнитного покрытия. Наиболее эффективными для температурных воздействий являются связующие материалы органической природы, которые имеют температуру плавления или деформации меньше точки Кюри.

Второй способ заключается в размагничивании ферромагнетика медленно убывающим переменным магнитным полем. В случае с накопителем на жестком магнитном диске возникают трудности, связанные с большой остаточной намагниченностью.

Третий способ основан на представлении внешнего магнитного поля как аналога поля, создаваемого магнитными головками при записи. Если напряженность внешнего поля будет превышать напряженность магнитного поля, создаваемого головками, назначение, при котором происходит магнитное насыщение, то все магнитные домены будут переориентированы по направлению этого внешнего поля и вся информация на диске будет уничтожена.

Программа для создания зашифрованной области на жестком диске DriveCrypt Plus

Данный программный комплекс разработан компанией используют многие российские предприятия для защиты данных. Надежное хранение конфиденциальных данных обеспечивается при выполнении следующих условий:

1. Компьютер, на котором находится информация, не должен быть подключен ни к каким сетям.
2. Зашифровать не только данные, но и весь диск.
3. Используемая система криптозащиты должна уметь создавать ложную операционную систему, доступ к которой возможен при вводе определенного пароля.
4. Экран для ввода пароля не должен раскрывать факт использования системы криптозащиты.
5. Подбор пароля должен пресекаться.
6. Переустановка операционной системы, подключение диска к другой машин, работа ложной операционной системы не должны выдавать факта использования системы криптозащиты.
7. Продолжительная работа в ложной системе должна вызывать частичное уничтожение или разрушение структуры.
8. Работа с криптосистемой должна быть максимально прозрачной для пользователя, но не должна требовать от пользователя специальных знаний или выполнения действий с определенной периодичностью.

Работа программы по шифрованию включает в себя выполнение следующих этапов:

1. Создание и регистрация ключей ЭЦП – электронная цифровая подпись.
2. Защита и ограничение доступа к операционной системе.
3. Шифрование загрузочного диска или раздела.
4. Создание ложной операционной системы.

SQL – инъекция – намеренное действие пользователя, направленное на внедрение любого SQL запроса в логику работы скрипта. Или это данные в виде фрагментов SQL запросов.

Для примера рассмотрим следующий скрипт, где происходит проверка на соответствие логина и пароля из формы с данными в базе. Если соответствует, выводится сообщение true, в противном случае false.

<? Mysql\_select\_db(‘база’, mysql\_connect(‘localhost’, ‘root’, ‘’));

If ($\_GET[‘login’] && $\_GET[‘pass’])

{ $login = $GET[‘login’];

$pass = $GET[‘pass’];

$sql = “select \* from ‘user’ where ‘login’ = ‘$login’ and

‘password’ = ‘$pass’”;

$resuilt = mysql\_query($sql) or die(mysql\_error());

If mysql\_num\_rows ($result)) echo///

}

Для защиты от инъекций используют функции:

mysql\_real\_escape\_string() – экранирует специальные символы строки таким образом, что синтаксис параметра sql запроса изменяется и приводит его в негодность.

sprintfx() – работает, если пароль – целое число.

Методы защиты от инъекций:

1. Создание менее привилегированного пользователя
2. Отключение сообщения об ошибках
3. Использование запросов с параметром
4. Использование хранимых процедур
5. Использование функции блокировки
6. Применение регулярных выражений

Запрещается в SQL код передавать данные из формы напрямую.

Обычно обеспечивают контроль длины и контроль соответствия типов. Если используется коллекция Parameters, то вводимые данные обрабатываются как буквенное значение, а не исполняемый код.

Данные, поступаемые через входные параметры или из считываемой таблицы, должны быть помещены в функцию QUOTENAME(), REPLACE(). Они имеют ограничение в 128 символов.

SQL инъекции с помощью усечения данных

Любое присваивание переменной другого значения усекается, если оно не умещается в буфер, назначенный для этой переменной. Если злоумышленник способен обеспечить усечение инструкций непредвиденно длинными строками, он получает возможность манипулировать результатом. Так, следующая процедура уязвима:

Show Database отображает все базы на сервере.

Mysql -h имя\_хоста -u логин -p база

Давайте создадим базу, которая будет хранить информацию о домашних животных.

CREATE DATAVASE MyPet;

USE MyPet;

CREATE TABLE pet(

Namep varchar(20),

Owner varchar(50),

Vid varchar(20),

Pol char(1),

Dr date, (год – месяц – день)

Ds DATE);

Загрузка данных из текстового файла:

LOAD DATA LOCAL INFILE ‘pet.txt’ INTO TABLE pet;

возраст

SELECT \* from pet where vid = ‘кошка’ or vid = ‘собака’;

SELECT Namep, dr, current\_date, (year(current\_date) – year(dr)) –

(right(current\_date, 5) < right(dr, 5))

As age

From pet

Возраст если умер

Select namep, dr, ds

(year(ds) – year(dr)) –

(right(ds, 5) < right(dr,5))

As age from pet

Where ds is not null;

Вывести клички и даты рождения для тех животных, у которых день рождения в следующем месяце.

Select namep, dr from pet

Where month(dr) = month(date\_add(now(), interval 1 month));

REGEXP – аналог Like

Существуют следующие правила задания регулярных выражений:

1. Точка соответствует любому одиночному символу.
2. Диапазон возможных значений задается в квадратных скобках [abc] [a-zA-Z].
3. Звездочка соответствует нулю или большему количеству образцов, предшествующих ей.
4. Регулярные выражения чувствительны к регистру.
5. Образец MySQL является соответствующим, если соответствие происходит где-нибудь в проверяемом значении.
6. Чтобы закреплять образец так, чтобы он соответствовал строго началу или концу проверяемого значения, используется символ циркумфлекса ^ в начале или значка доллара в конце образца.

SELECT \* from pet

Where namep regexp “^Б”

SELECT \* from pet

Where namep regexp “ка$”

SELECT \* from pet

Where namep regexp “^…..$”

Некоторые последовательности имеют специальное значение. Каждая из этих последовательностей начинается с обратного слеша и известна как управляющий символ.

\’ \’’ \b \n \t \\ \r

Myisamchk -r – аналог repairtable

Она будет работать, если mysql запущен с функцией -log-update

1. Восстановить оригинальную копию mysqldump
2. Выполнить команду чтобы заново выполнить модификацию в двоичном файле регистрации

Mysqlbinlog hostname -bin.[0-9]\*|mysql

Ls (используется для получения всех журналов модификации в правильном порядке) -l -t -r hostname.[0-9]\* |xargs cat|mysql

Backup table имя\_табл1,[имя\_табл2,…] to “/path/to/backup/directory” делает бэкап таблицы в каталог.

Restore table имятабл from “path/path/”

Проверяет, есть ли в таблице ошибки

Check table имя табл [option quick|fast|medium|extended|changed]

ANALYZE TABLE табл1[,табл2,…] – данная команда анализирует и сохраняет распределение ключей для таблицы. В процессе анализа таблица будет блокирована с доступом только для чтения.

MySQL использует сохраненное распределение ключей, чтобы решить, в каком порядке будут объединены таблицы в ходе выполнения.

Чтобы проверить сохраненное распределение ключей, выполнить SHOW INDEX FROM таблица

FLUSH опция1,..

Опции: HOSTS LOGS PRIVILEGES TABLES TABLE табл1… TABLES WITH READ LOCK

FLUSH HOSTS освобождает ведущие таблицы хэша. Данная команда применяется, когда некоторые из хостов изменяют айпи или было получено сообщение об ошибке.

FLUSH LOGS открывает и закрывает журналы.

FLUSH PRIVILEGES перезагружает привилегии

FLUSH TABLES закрывает все открытые таблицы

FLUSH STATUS сбрасывает большинство переменных к нулю, используется при отладке запроса.

LOCK TABLES закрывает все открытые таблицы и блокирует все таблицы для всех баз данных с доступом только на чтение, пока не будет выполнена UNLOCK TABLES.

SHOW PROCESSLIST

KILL айди процесса

В большинстве случаев может требоваться некоторое время, чтобы процесс остановился, так как метка уничтожения может быть проверена в специфических интервалах:

1. В циклах SELECT, ORDER BY и GROUP BY будет проверена после чтения блока строк.
2. При выполнении ALTER TABLE метка будет проверена прежде, чем каждый блок строк читается из первоначальных таблиц.
3. При выполнении UPDATE TABLE и DELETE TABLE метка будет проверена после чтения каждого блока и после каждого обновления и удаления строки.

Все процессы INSERT DELAYED быстро сбросят на диск все строки в памяти и завершатся.

Язык HTML

Структура документа:

<HTML>

<HEAD>

<SCRIPT>

Программа-скрипт на Java-Script

</SCRIPT>

<TITLE> Имя страницы </TITLE>

</HEAD>

<BODY>

Основное содержание страницы

</BODY>

</HTML>

Для стилевого оформления HTML документа используются теги, которые прописываются в теге BODY

Background = “адрес”

Bgcolor = “цвет 16х”

Alink = “цвет” – при наведении

Wlink = “цвет” – посещенная ссылка

Link = “цвет” – ссылка

Text = “цвет” – текст

Для определения стиля документа могут определяться следующие теги:

<FONT

Size = “n” (n = 1..7)

Face = “Имя шрифта”

Color = “Цвет”

/FONT>

<Hn>, где n = 1..6 - заголовок

<P> - выделение абзаца

<BR> - новая строка

<PRE> - сохраняет то форматирование, которое указано в редакторе.

<I> - italic

<B> - bold

<U> - underlined

<S> - зачеркнутый

<HR> - горизонтальная линия

Для создания ненумерованного списка используется тег <ul>

TYPE = “DISK” | “CIRCLE” | “SQUARE”

<LI> - Элемент списка

Пример:

<HTML>

<BODY>

<UL TYPE = “SQUARE”>

<LI> Иванов

<LI> Петров

<LI> Сидоров

</UL>

</BODY>

</HTML>

<OL>

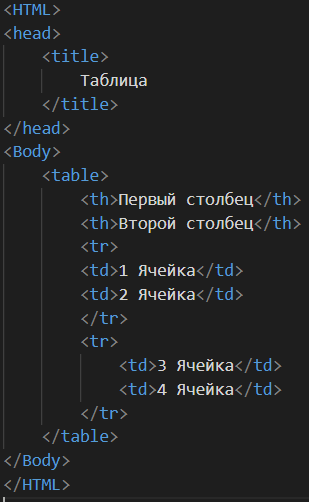
Start = “n”

Type = “1” | “A”

<TABLE>

<TR>

<TD>



<Marquee> - бегущая строка

align = “top” | middle| bottom

Behavior = scroll|slide|alter|native

Bgcolor = color

Height width

Loop = “n”

<img> src = address

Align = left|middle|top|bottom|right

Border = n

Dynsrc = address

Loop – кол-во повторений

Start = “событие для запуска клипа”

<bg sound> - музыка для клипа.

<a href = address>текст</a>

Target = zn zn = имя окна или его части, куда должна загружаться страница.

В веб документах могут также использоваться фреймы. При помощи них можно разделить окно браузера на несколько частей и загрузить каждую из них в своей веб странице.

<frameset> cols = “n”

Rows = “n”

<frame> определяет содержимое каждого фрейма браузера

Обязательный атрибут = src – адрес веб страницы, загружаемой в этот фрейм.

Тег object

Standby = “text”

Name

Codebase = path

Теги для создания интерфейса веб страниц.

Для организации диалога между клиентом и сервером используются формы.

С помощью тега <FORM> можно создавать различные формы.

Action = ‘Имя файла на сервере’

Method = get|post. get – небольшие объемы данных

Создание объектов интерфейса происходит с помощью тега Input

Type = button|submit|reset|password|text|radio|checkbox

Name = имя объекта

Value = значение

Value для:

Button, Submit и reset – определяет надпись на кнопке.

Password, text – текст по умолчанию.

Radio, checkbox – определяет значения, передаваемые программе при выборе этих объектов.

Для создания в форме списков используется парный тег select

Multiple

Name

Size

Элементы списка формируются при помощи тега option

Value = умолчание