Базы данных

Гаврилов Александр Викторович
к.т.н., доцент

Тема 12. Безопасность баз данных

Лекция 12

Вопросы лекции:

1. Понятия безопасности данных
2. Управление доступом в базах данных
3. Управление целостностью данных
4. Управление параллелизмом
5. Восстановление данных

Понятия безопасности данных

Понятие защиты данных

- Под **защитой данных** подразумевается предотвращение доступа к ним со стороны несанкционированных пользователей.
- Под **поддержкой целостности данных** подразумевается предотвращение их разрушения при доступе со стороны санкционированных пользователей.

Защита базы данных

- Обеспечение защищенности базы данных против любых предумышленных или непредумышленных угроз с помощью различных компьютерных и некомпьютерных средств.

Доступ к информации

Доступ к информации - ознакомление с информацией и ее обработка.

Субъект доступа - лицо или процесс, действия которого регламентируются правилами разграничения доступа.

Объект доступа - информационная единица АС, доступ к которой регламентируется правилами разграничения доступа (объектом может быть все что угодно, содержащее конечную информацию: база данных, таблица, строка, столбец и т.

Управление доступом в базах данных

Доступ к информации

Большинство систем БД представляют собой средство единого централизованного хранения данных. Это значительно сокращает избыточность данных, упрощает доступ к данным и позволяет более эффективно защищать данные . Однако, в технологии БД возникает ряд проблем, связанных, например, с тем, что различные пользователи должны иметь доступ к одним данным и не иметь доступа к другим. Поэтому, не используя специальные средства и методы, обеспечить надежное разделение доступа в БД практически невозможно.

Большинство современных СУБД имеют встроенные средства, позволяющие администратору системы определять права пользователей по доступу к различным частям БД, вплоть до конкретного элемента . При этом имеется возможность не только предоставить доступ тому или иному пользователю, но и указать разрешенный тип доступа: что именно может делать конкретный пользователь с конкретными данными (читать, модифицировать, удалять и т . п .), вплоть до реорганизации всей БД Таблицы (списки) управления доступом широко используются в компьютерных системах, например, в ОС для управления доступом к файлам. Особенность использования этого средства для защиты БД состоит в том, что в качестве объектов защиты выступают не только отдельные файлы (области в сетевых БД, отношения в реляционных БД), но и другие структурные элементы БД: элемент, поле, запись, набор данных,

Доступ к информации

Идентификатор доступа – (login) уникальный признак объекта или субъекта доступа.

Пароль - идентификатор субъекта, который является его секретом.

Разграничение доступа

Правила разграничения доступа - совокупность правил, регламентирующих права субъектов доступа к объектам доступа.

Санкционированный доступ - доступ к информации в соответствии с правилами разграничения доступа.

Несанкционированный доступ - доступ к информации, нарушающий правила разграничения доступа в любом проявлении или реализации.

Процесс получения доступа пользователя к БД в СУБД

- **Идентификация** — установление тождественности неизвестного объекта известному на основании совпадения признаков; опознание. **Идентификатор доступа** – имя пользователя (login).
- **Аутентификация** — проверка подлинности предъявленного пользователем идентификатора путём сравнения введённого им пароля с паролем, сохранённым в базе данных пользователей

Два подхода к вопросу обеспечения безопасности данных

В современных СУБД поддерживается один из двух наиболее общих подходов к вопросу обеспечения безопасности данных:

избирательный подход

обязательный подход.

В обоих подходах единицей данных или "объектом данных", для которых должна быть создана система безопасности, может быть как вся база данных целиком, так и любой объект внутри базы данных.

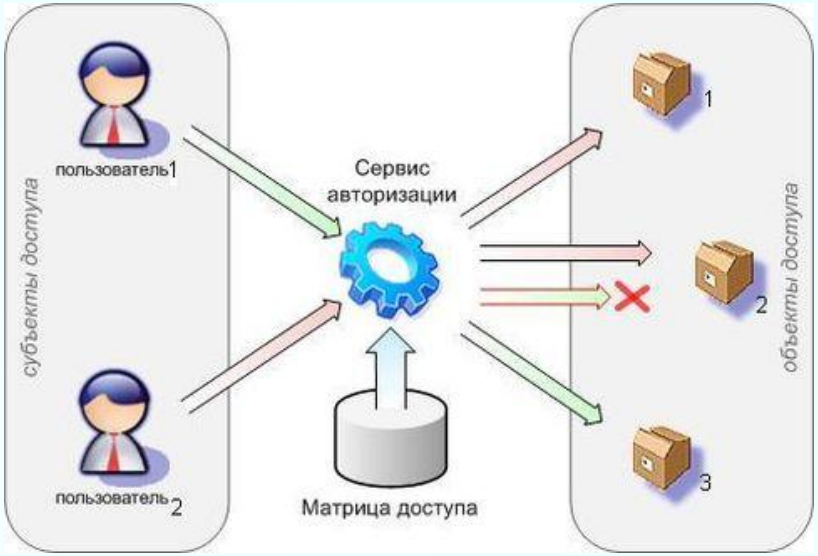
Избирательный подход

В случае **избирательного управления** некоторый пользователь обладает различными правами (привилегиями или полномочиями) при работе с данными объектами. Разные пользователи могут обладать разными **правами доступа** к одному и тому же объекту. Избирательные права характеризуются значительной гибкостью.

Объект / Субъект	O_1	O_2	O_3	O_4(Printer)
S_1	read			
S_2				Print
S_3		Read	Execute	
S_4	read write		read write	

Избирательный подход

Дискреционное управление доступом — разграничение доступа между поименованными субъектами и поименованными объектами.



Доступ к полям *

Доступ к полям

Документ "АвансовыйОтчет"

Таблица/Поле	Пользователь/Роль	Админенко Виктор Сергеевич	Виталов Дмитрий Тимурович	Владимиров Александр Петрович	Володин Валентин Викторович	Егоров Олег Эдуардович	Игров Виталий Евгеньевич	Константинов Сергей Алексеевич	Наталькук Елена Владимировна	Николаев Алексей Максимович	Романенко Петр Алексеевич	Таненко Акила Алимовна	Фёдоров Михаил Васильевич	Администратор ККМ	Бухгалтер
Основная															
ВыданныеАвансы															
Чтение	Доступность	V	V	V	V	V	V	V	V	V	V	V	V	V	V
	Ограничение условием					V									
Добавление	Доступность	V	V	V	V		V	V	V	V	V	V	V	V	V
	Ограничение условием					V									
Удаление	Доступность	V	V	V			V	V	V			V			V
	Ограничение условием				V	V				V	V		V	V	
Изменение	Доступность	V	V	V	V		V	V	V	V	V	V	V	V	V
	Ограничение условием					V									
НомерСтроки															
Чтение	Доступность	V	V	V	V	V	V	V	V	V	V	V	V	V	V
	Ограничение условием					V									
Добавление	Доступность	V	V	V	V		V	V	V	V	V	V	V	V	V
	Ограничение условием					V									
Удаление	Доступность	V	V	V			V	V	V			V			V
	Ограничение условием				V	V				V	V		V	V	
Изменение	Доступность	V	V	V	V		V	V	V	V	V	V	V	V	V
	Ограничение условием					V									
РасходныйКассовыйОрдер															
Сумма															
Товары															
НомерСтроки															
ВидДокВходящий															

Избирательный подход

Развитием политики избирательного управления доступом является ролевое разграничение доступа, при котором права доступа субъектов системы на объекты группируются с учетом специфики их применения, образуя роли.

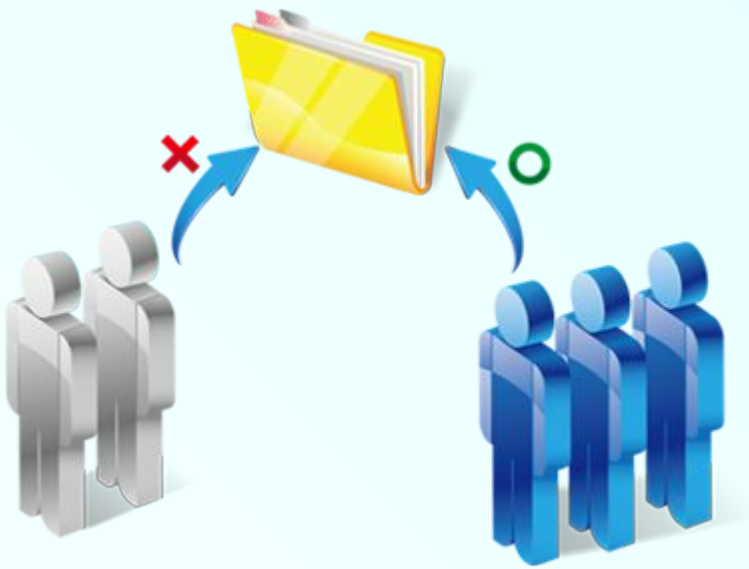
Избирательный подход

Ролевая модель системы

- Возможность выполнения действия определяется ролью пользователя в системе
- Для каждой роли свой набор функций и свой набор представлений

Обязательный подход

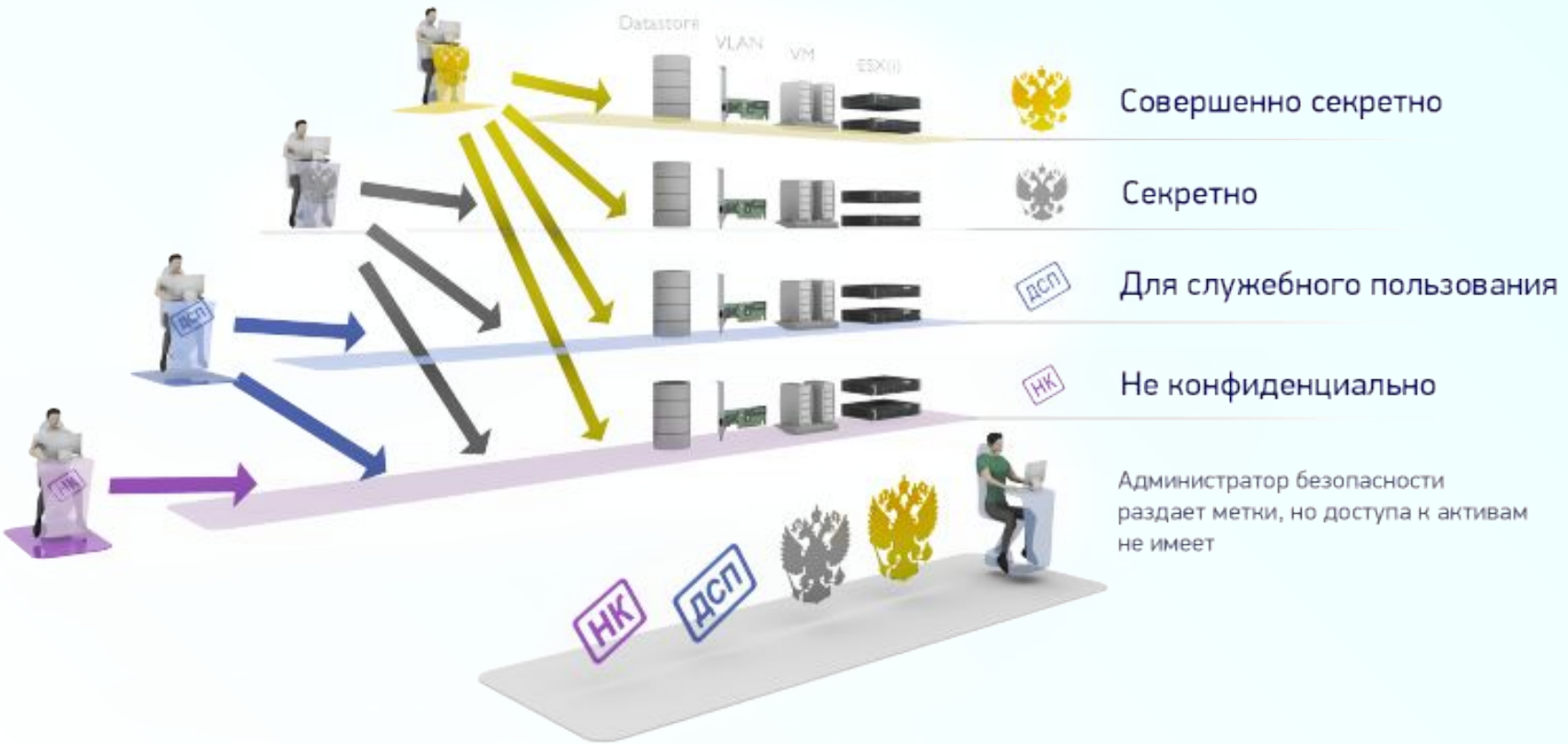
В случае **обязательного управления (мандатного контроля)**, наоборот, каждому объекту данных присваивается некоторый классификационный уровень, а каждый пользователь обладает некоторым уровнем допуска. При таком подходе доступом к определенному объекту данных обладают только пользователи с соответствующим уровнем допуска.



Мандатные схемы обычно имеют иерархическую структуру и поэтому являются более жесткими.

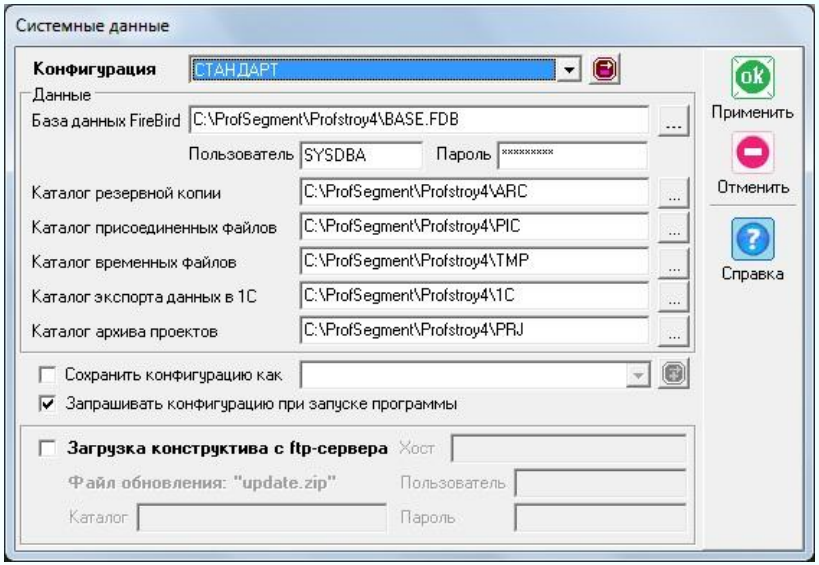
Обязательный подход

Мандатные схемы обычно имеют иерархическую структуру и поэтому являются более жесткими.



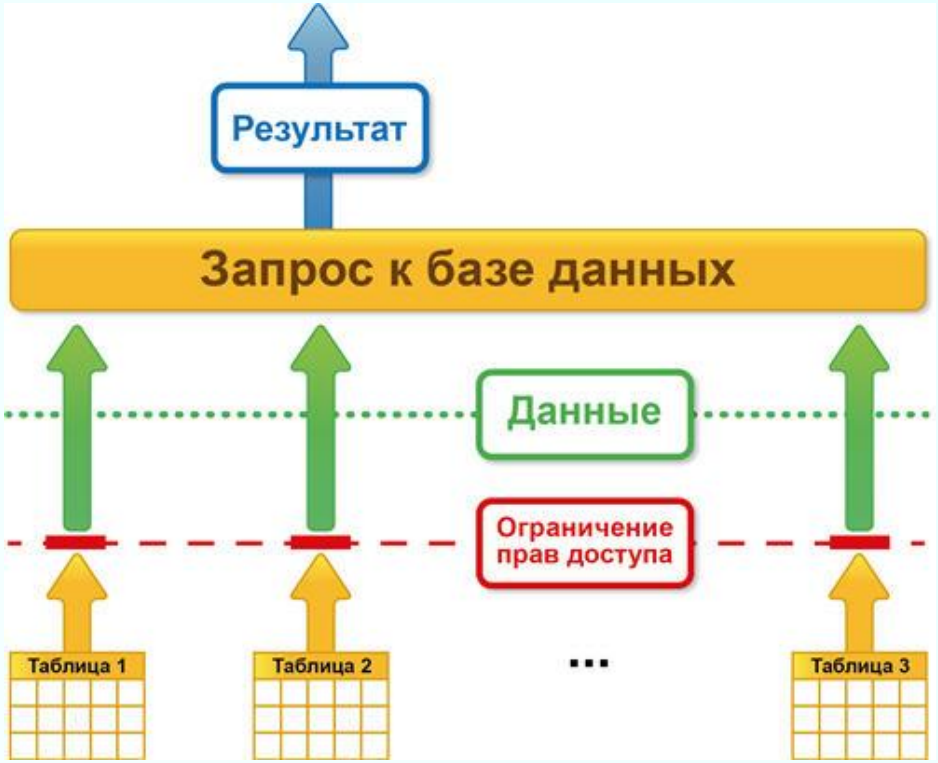
Принципы организации доступа к объектам БД

Принятые организационные решения (относительно предоставления пользователям прав на выполнение тех или иных операций с теми или иными объектами) должны быть доведены до сведения системы (т.е. представлены как ограничения защиты, выраженные с помощью некоторого языка описания требований защиты) и должны быть ей постоянно доступны (храниться в системном каталоге).



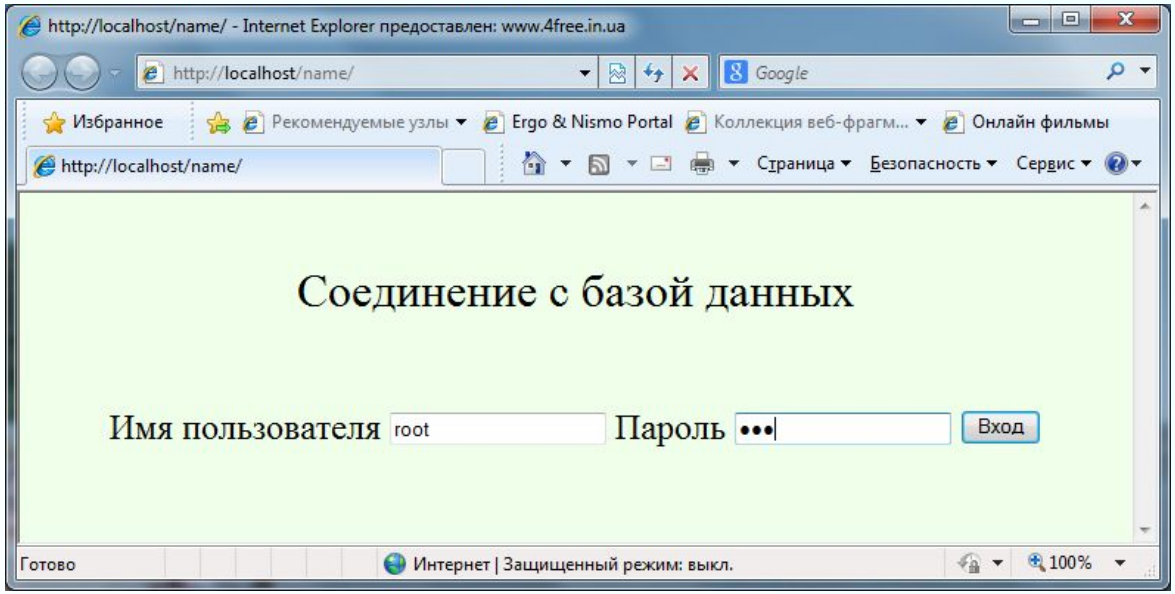
Принципы организации доступа к объектам БД

В системе должны существовать определенные **средства проверки поступающих запросов на получение доступа по отношению к установленным правилам защиты** (запрос на получение доступа включает комбинацию запрашиваемой операции, запрашиваемого объекта и запрашивающего пользователя).



Принципы организации доступа к объектам БД

Для принятия решения о том, какие именно установленные ограничения защиты применимы к данному запросу на получение доступа, **система должна быть способна установить источник** этого **запроса**, т.е. суметь опознать запрашивающего пользователя. Поэтому при подключении к системе от **пользователя** обычно требуется ввести не только свой **идентификатор** (чтобы указать, кто он такой), но и **пароль** (чтобы подтвердить, что он именно тот, за кого себя выдает). Предполагается, что пароль известен только системе и тем лицам, которые имеют право применять данный идентификатор пользователя.



Методы реализации избирательного принципа:

- В базу данных вводится новый тип объектов БД — **пользователи**.
- Каждому пользователю в БД присваивается уникальный *идентификатор*.
- Для дополнительной защиты каждый пользователь кроме уникального идентификатора снабжается уникальным *паролем.* Пароли пользователей хранятся чаще всего в специальном кодированном виде и известны только самим пользователям.

- Пользователи могут быть объединены в специальные группы пользователей.
- Один пользователь может входить в несколько групп.
- В стандарте вводится понятие группы *PUBLIC*, для которой должен быть определен минимальный стандартный набор прав. По умолчанию каждый вновь создаваемый пользователь, относится к группе PUBLIC.

Привилегии и роли

- **Привилегии** или полномочия **пользователей** или групп — это набор действий (операций), которые они могут выполнять над объектами БД.
- **Роль** — это поименованный набор полномочий. Существует ряд стандартных ролей, которые определены в момент установки сервера баз данных. И имеется возможность создавать новые роли, группируя в них произвольные полномочия. Введение ролей позволяет упростить управление привилегиями пользователей, структурировать этот процесс. Кроме того, введение ролей не связано с конкретными пользователями, поэтому роли могут быть определены и сконфигурированы до того, как определены пользователи системы.

Привилегии и роли

- ✔ Пользователю может быть назначена одна или несколько ролей.
- ✔ Объектами БД, которые подлежат защите, являются все объекты, хранимые в БД: **таблицы, представления, хранимые процедуры и триггеры**. Для каждого типа объектов есть свои действия, поэтому для каждого типа объектов могут быть определены разные **права доступа**.

Учетные записи пользователей (на примере СУБД MySQL)

- Под **учетной записью пользователя MySQL** подразумевается строка в таблице **user** системной базы данных **mysql**.
- Первичным ключом в этой таблице служат столбцы **Host** и **User**, т.е. идентификация пользователя основана на комбинации имени пользователя и хоста, с которого одключается пользователь.
- Столбец **User** допускает значения длиной не более 16 символов.
- Столбец Host допускает следующие значения:
  - конкретное имя компьютера или IP-адрес;
  - маска подсети (например, 192.168.1.0/255.255.255.0);
  - маска имени компьютера или маска IP-адреса, которая может содержать подстановочные символы: **%** (заменяет любое количество произвольных символов) и **_** (заменяет один произвольный символ).

MySQL

Регистрация пользователя

- Для создания учетной записи пользователя, используется команда

```
CREATE USER <Идентификатор пользователя>
    [IDENTIFIED BY [PASSWORD] '<Пароль>'];
```

- Обязательным параметром этой команды является идентификатор нового пользователя. Если не задан параметр IDENTIFIED BY, то будет использоваться пустой пароль.

- Например, команда

```
CREATE USER 'anna' IDENTIFIED BY 'annapassword';
```

создает учетную запись для пользователя с именем anna, подключающегося с любого компьютера, и устанавливает для этой учетной записи пароль annapassword.

Установка пароля

- Для установки пароля предназначена команда

```
SET PASSWORD [FOR <Идентификатор пользователя>]
=
    PASSWORD('<Пароль>');
```

- Обязательным параметром этой команды является идентификатор нового пользователя. Если не задан параметр IDENTIFIED BY, то будет использоваться пустой пароль.

- Например, команда

```
CREATE USER 'anna' IDENTIFIED BY
'annapassword';
```

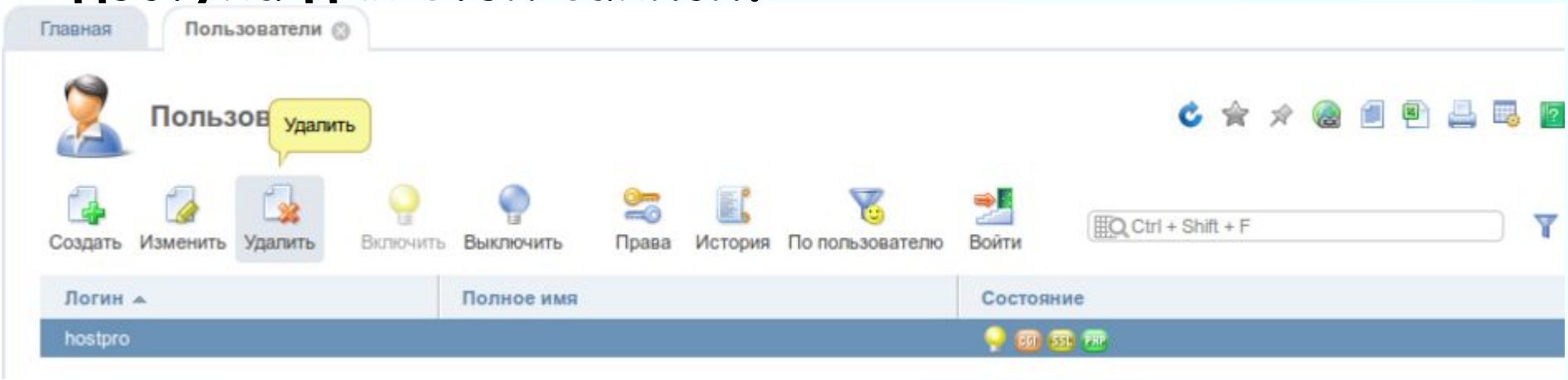
создает учетную запись для пользователя с именем anna, подключающегося с любого компьютера, и устанавливает для этой учетной записи пароль annapassword.

Удаление пользователя

- Для удаления учетной записи используется команда

  DROP USER <Идентификатор пользователя>;

- После удаления пользователь лишается возможности подключаться к серверу MySQL.
- Однако если на момент удаления пользователь был подключен к серверу, то соединение не прерывается.
- Вместе с учетной записью удаляются все привилегии доступа для этой записи.



Система привилегий доступа

Создание привилегии доступа в MySQL подразумевает определение следующих параметров:

- идентификатор учетной записи пользователя, которому предоставляется привилегия;
- тип привилегии, то есть тип операций, которые будут разрешены пользователю;
- область действия привилегии.

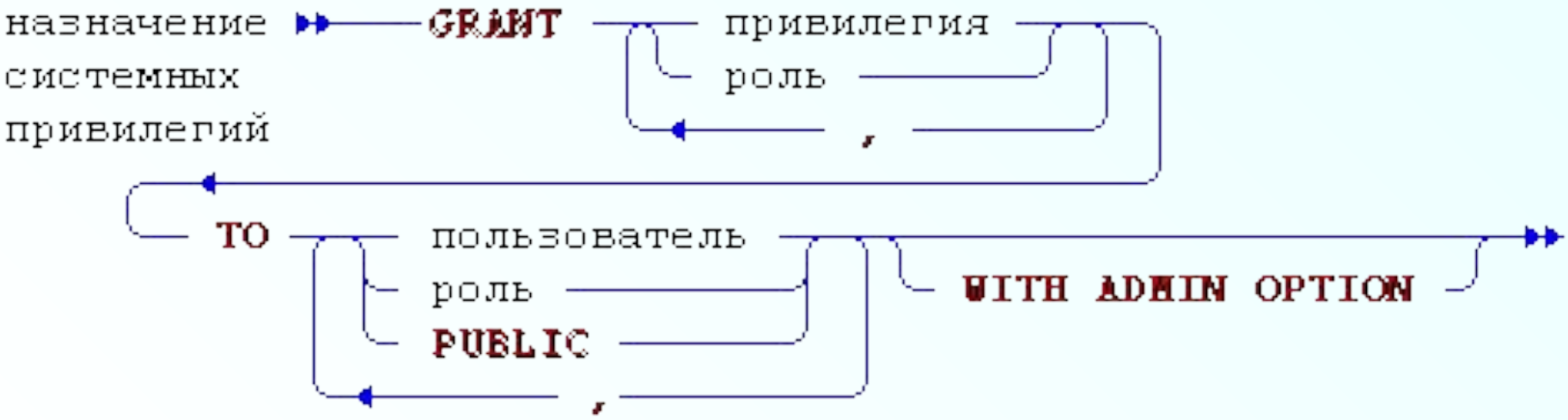
В MySQL используются следующие основные типы привилегий:

- ☑ ALL [PRIVILEGES] – предоставляет все привилегии, кроме GRANT OPTION, для
- ☑ указанной области действия;
- ☑ ALTER – разрешает выполнение команд ALTER DATABASE и ALTER TABLE;
- ☑ CREATE – разрешает выполнение команд CREATE DATABASE и CREATE TABLE;
- ☑ CREATE USER – разрешает выполнение команд CREATE USER, DROP USER, RENAME USER;
- ☑ DELETE – разрешает выполнение команды DELETE;
- ☑ DROP – разрешает выполнение команд DROP DATABASE и DROP TABLE;
- ☑ FILE – разрешает чтение и создание файлов на сервере с помощью команд SELECT… INTO OUTFILE и LOAD DATA INFILE;
- ☑ INDEX – разрешает выполнение команд CREATE INDEX и DROP INDEX;
- ☑ INSERT – разрешает выполнение команды INSERT;
- ☑ SELECT – разрешает выполнение команды SELECT;
- ☑ LOCK TABLES – разрешает выполнение команды LOCK TABLES при наличии привилегии SELECT для блокируемых таблиц;
- ☑ SHOW DATABASES – разрешает отображение всех баз данных при выполнении команды SHOW DATABASES (если эта привилегия отсутствует, то в списке будут отображены только те базы данных, по отношению к которым у пользователя есть какая-либо привилегия);
- ☑ RELOAD – разрешает выполнение команды FLUSH;
- ☑ SUPER – привилегия администратора сервера; в частности, разрешает выполнение
- ☑ команды SET GLOBAL;
- ☑ UPDATE – разрешает выполнение команды UPDATE;
- ☑ GRANT OPTION – разрешает назначать и отменять привилегии другим

Предоставление привилегий

- **Для предоставления привилегий пользователям используется команда**

  GRANT <Тип привилегии>
  [(<Список столбцов>)] ON <Область действия>
  TO <Идентификатор пользователя>
  [WITH GRANT OPTION];



Предоставление привилегий

- В качестве области действия можно указать одно из следующих значений:
  - *.* – привилегия будет действовать глобально;
  - <Имя базы данных>.* – привилегия будет действовать для указанной базы данных;
  - * – привилегия будет действовать для базы данных, которая в момент выполнения команды GRANT являлась текущей;
  - <Имя базы данных>.<Имя таблицы> или <Имя таблицы> – привилегия будет действовать для указанной таблицы (если имя базы данных не указано, подразумевается текущая база данных).
- Если требуется создать привилегию не для всей таблицы, а только для отдельных столбцов, необходимо перечислить эти столбцы в скобках

Отмена привилегий

- Чтобы удалить привилегию, ранее назначенную пользователю, используется команда

```
REVOKE <Тип привилегии>
        [(<Список столбцов>)] ON <Область действия>
                FROM <Идентификатор пользователя>;
```

Например:

```
REVOKE CREATE ON *.* FROM 'anna'@'localhost';
```

Управление целостностью данных

Управление целостностью данных

- Нарушение целостности данных может быть вызвано рядом причин:

  - ☑ сбои оборудования, физические воздействия или стихийные бедствия;
  - ☑ ошибки санкционированных пользователей или умышленные действия несанкционированных пользователей;
  - ☑ программные ошибки СУБД или ОС;
  - ☑ ошибки в прикладных программах;
  - ☑ совместное выполнение конфликтных запросов пользователей и др.

- Нарушение целостности данных возможно и в хорошо отлаженных системах. Поэтому важно не только не допустить нарушения целостности, но и своевременно обнаружить факт нарушения целостности и оперативно

Управление параллелизмом

Управление параллелизмом

- В системах, ориентированных на многопользовательский режим работы, возникает целый ряд новых проблем, связанных с параллельным выполнением конфликтующих запросов пользователей.
- Важнейшим средством механизма защиты целостности БД выступает объединение совокупности операций, в результате которых БД из одного целостного состояния переходит в другое целостное состояние, в один логический элемент работы, называемый транзакцией. Суть механизма транзакций состоит в том, что до завершения транзакции все манипуляции с данными проводятся вне БД, а занесение реальных изменений в БД производится лишь после нормального завершения транзакции.

Управление параллелизмом

- Если транзакция была прервана, то специальные встроенные средства СУБД осуществляют так называемый откат - возврат БД в состояние, предшествующее началу выполнения транзакции Если выполнение одной транзакции не нарушает целостности БД, то в результате одновременного выполнения нескольких транзакций целостность БД может быть нарушена. Чтобы избежать подобного рода ошибок, СУБД должна поддерживать механизмы, обеспечивающие захват транзакциями модифицируемых элементов данных до момента завершения модификации так называемые блокировки. При этом гарантируется, что никто не получит доступа к модифицируемому элементу данных, пока транзакция не освободит его. Применение механизма блокировок приводит к новым проблемам управления параллелизмом, в частности, к возникновению ситуаций клинча двух транзакций. Причем, если некоторая транзакция пытается блокировать объект, который уже блокирован другой транзакцией, то ей придется ждать, пока не будет снята блокировка объекта транзакцией, установившей эту блокировку. Иными словами, блокировку объекта может выполнять только одна транзакция.

Восстановление данных

Восстановление данных

- **Возникновение сбоев в аппаратном или программном обеспечении может вызвать необходимость восстановления и быстрого возвращения в состояние, по возможности близкое к тому, которое было перед возникновением сбоя (ошибки). К числу причин, вызывающих необходимость восстановления, зачастую относится и возникновение тупиковой ситуации.**

Восстановление данных

- **Можно выделить три основных уровня восстановления:**

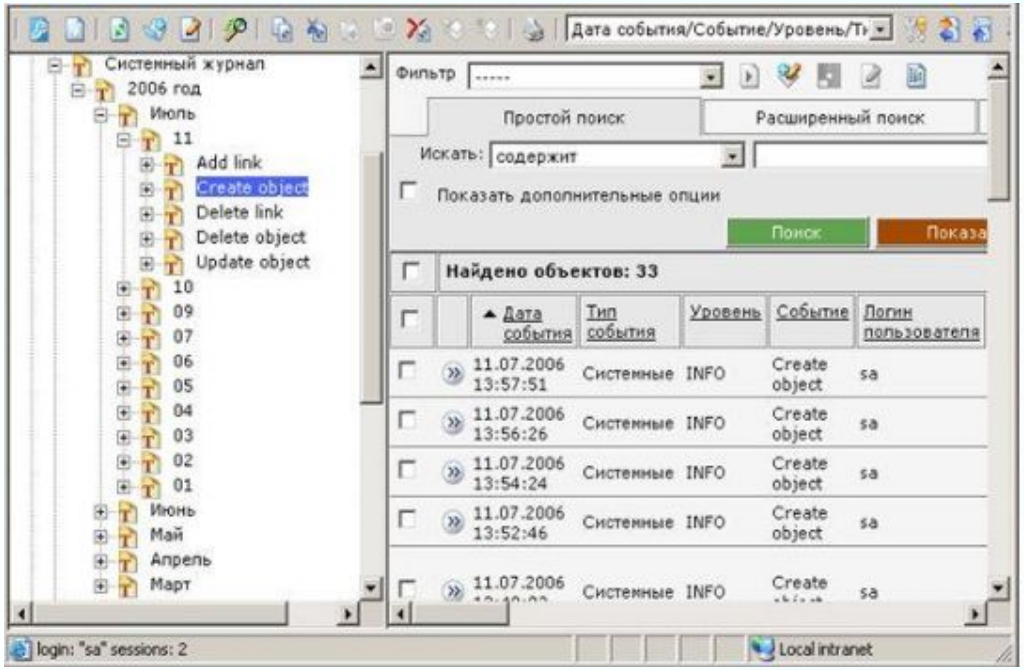
1. **Оперативное восстановление, которое характеризуется возможностью восстановления на уровне отдельных транзакций при ненормальном окончании ситуации манипулирования данными (например, при ошибке в программе).**
2. **Промежуточное восстановление. Если возникают аномалии в работе системы (системно-программные ошибки, сбои программного обеспечения, не связанные с разрушением БД), то требуется восстановить состояние всех выполняемых на момент возникновения сбоя транзакций.**
3. **Длительное восстановление. При разрушении БД в результате дефекта на диске восстановление осуществляется с помощью копии БД. Затем воспроизводят результаты выполненных с момента снятия копии транзакций и возвращают систему в**

Транзакция и восстановление

- Прекращение выполнения транзакции вследствие появления сбоя нарушает целостность БД. Если результаты такого выполнения транзакции потеряны, то имеется возможность их воспроизведения на момент возникновения сбоя. Таким образом, понятие транзакции играет важную роль при восстановлении. Для восстановления целостности БД транзакции должны удовлетворять следующим требованиям:
  - необходимо, чтобы транзакция или выполнялась полностью, или не выполнялась совсем;
  - необходимо, чтобы транзакция допускала возможность возврата в первоначальное состояние, причем, для обеспечения независимого возврата транзакции в начальное состояние монопольную блокировку необходимо осуществлять до момента завершения изменения всех объектов;
  - необходимо иметь возможность воспроизведения процесса выполнения транзакции, причем, для обеспечения этого требования, совместную блокировку необходимо осуществлять до момента завершения просмотра данных всеми

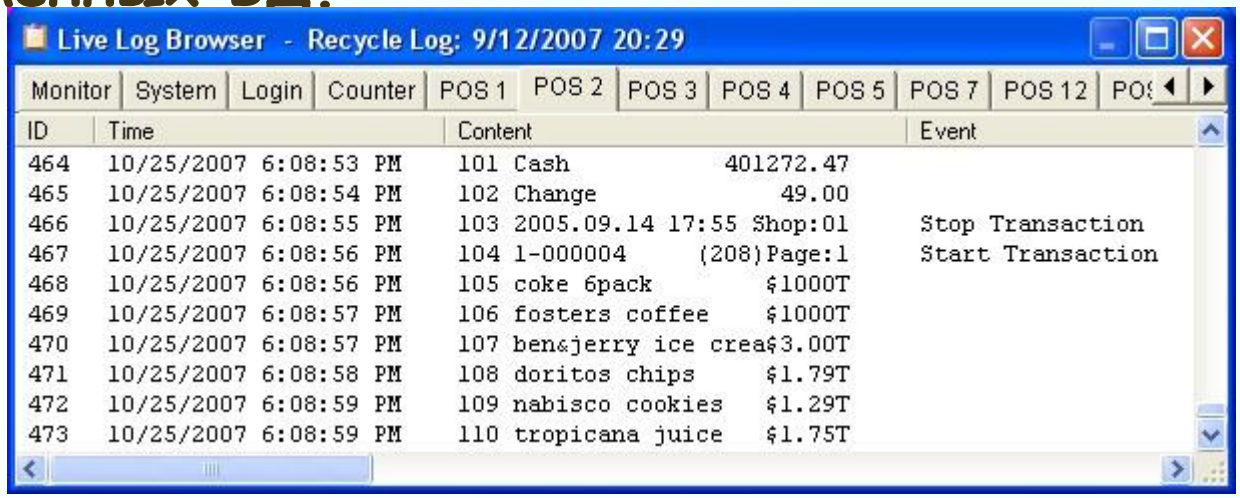
Откат и раскрутка транзакции

- Основным средством, используемым при восстановлении, является системный журнал, в котором регистрируются все изменения, вносимые в БД каждой транзакцией.
- Возврат транзакции в начальное состояние состоит в аннулировании всех изменений, которые осуществлены в процессе выполнения транзакции. Такую операцию называют откатом.



Откат и раскрутка транзакции

- Для воспроизведения результатов выполнения транзакции можно, используя системный журнал, восстановить значения проведенных изменений в порядке их возникновения, либо выполнить транзакцию повторно.
- **Воспроизведение результатов выполнения транзакции с использованием системного журнала называется раскруткой.** Раскрутка является достаточно сложной, но необходимой операцией механизмов восстановления современных БД.



Спасибо за внимание!