# 3. Администрирование MySQL – сервера

# 1. Проблемы безопасности и система привилегий доступа MySQL

## 1.1. Система привилегий

Система привилегий MySQL обеспечивает пользователям возможность выполнять только те действия, которые им разрешены.

**MySQL идентифицирует пользователя по имени хоста и по имени пользователя.**

**Управление доступом в MySQL осуществляется в два этапа:**

**Этап 1.** Сервер проверяет, имеется ли у вас вообще **разрешение на подсоединение**.

**Этап 2.** Если разрешение имеется, сервер **проверяет каждый запрос**, чтобы убедиться в том, что привилегий для его выполнения достаточно .

Например, если вы пытаетесь выбрать строки в таблице базы данных или удалить таблицу из базы данных, сервер в первом случае проверяет, имеется ли у вас для этой таблицы привилегия **SELECT**, а во втором - имеется ли у вас для этой базы данных привилегия **DROP**.

На обоих этапах управления доступом сервер использует таблицы

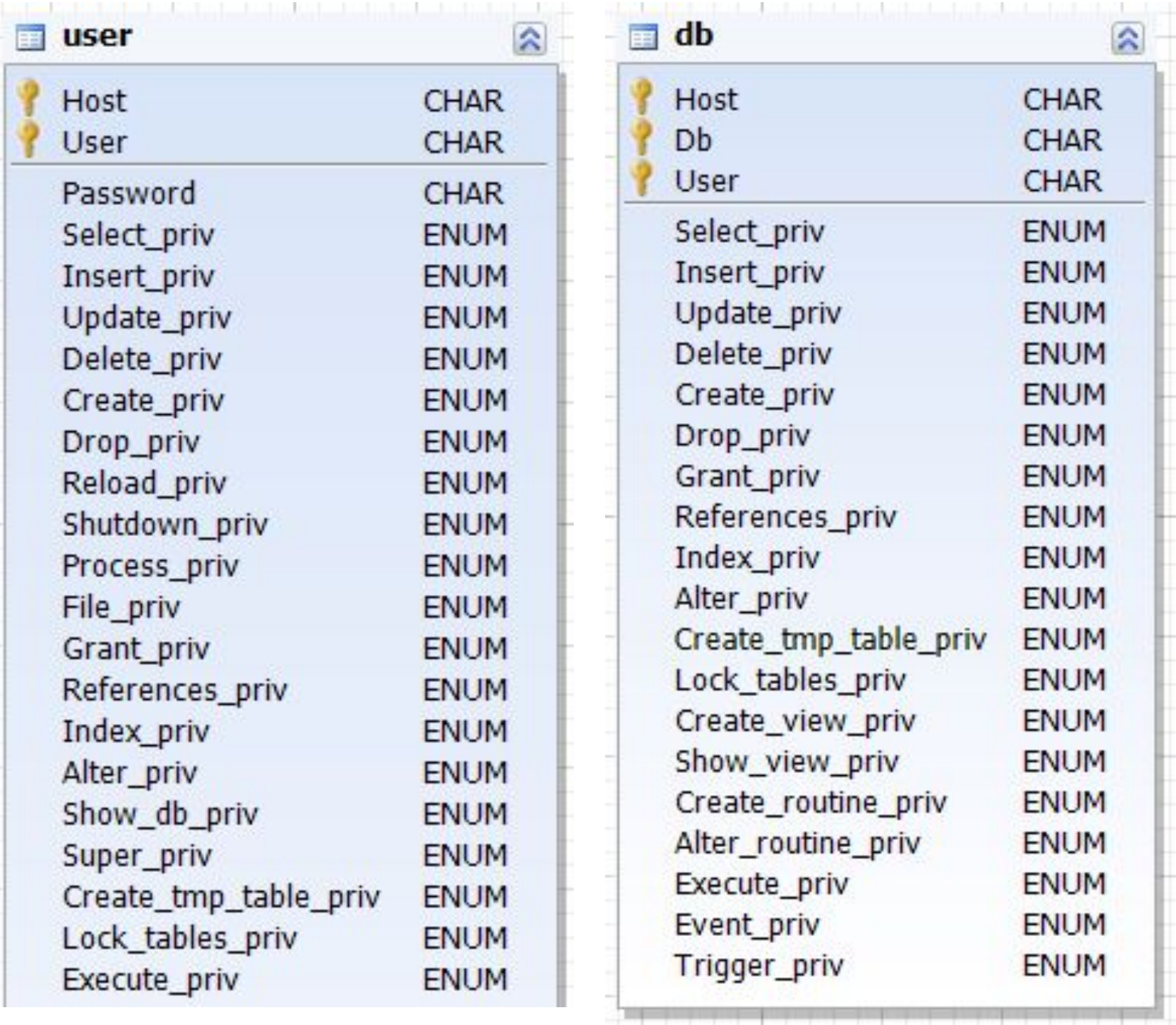
- **user**
- **db**
- **host**

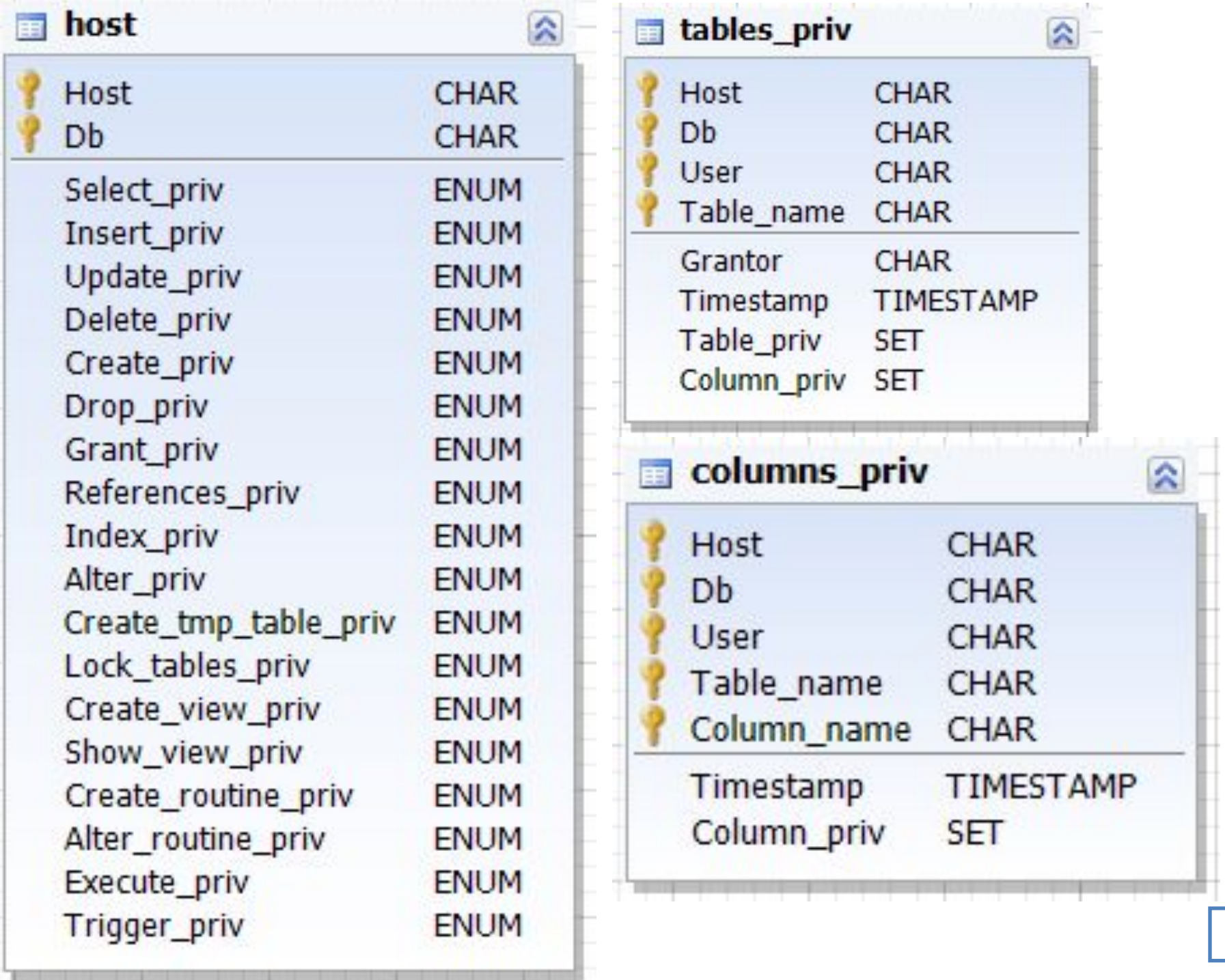
из базы данных **mysql**.

На втором этапе управления доступом сервер может дополнительно обратиться к таблицам

- **tables_priv**
- **columns_priv.**

**Таблицы привилегий используются следующим образом:**

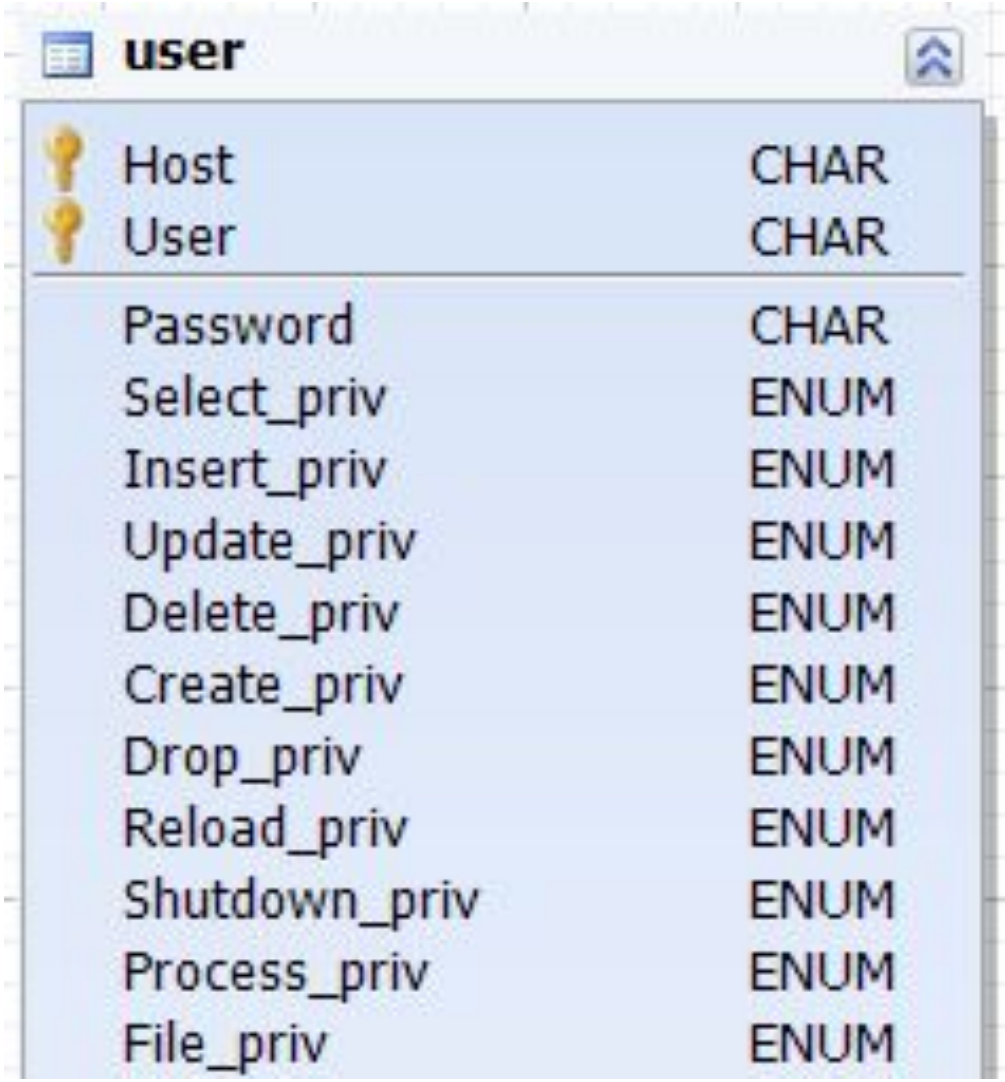

user
Host CHAR
User CHAR
Password CHAR
Select_priv ENUM
Insert_priv ENUM
Update_priv ENUM
Delete_priv ENUM
Create_priv ENUM
Drop_priv ENUM
Reload_priv ENUM
Shutdown_priv ENUM
Process_priv ENUM
File_priv ENUM
Grant_priv ENUM
References_priv ENUM
Index_priv ENUM
Alter_priv ENUM
Show_db_priv ENUM
Super_priv ENUM
Create_tmp_table_priv ENUM
Lock_tables_priv ENUM
Execute_priv ENUM
db
Host CHAR
Db CHAR
User CHAR
Select_priv ENUM
Insert_priv ENUM
Update_priv ENUM
Delete_priv ENUM
Create_priv ENUM
Drop_priv ENUM
Grant_priv ENUM
References_priv ENUM
Index_priv ENUM
Alter_priv ENUM
Create_tmp_table_priv ENUM
Lock_tables_priv ENUM
Create_view_priv ENUM
Show_view_priv ENUM
Create_routine_priv ENUM
Alter_routine_priv ENUM
Execute_priv ENUM
Event_priv ENUM
Trigger_priv ENUM


host
Host CHAR
Db CHAR
Select_priv ENUM
Insert_priv ENUM
Update_priv ENUM
Delete_priv ENUM
Create_priv ENUM
Drop_priv ENUM
Grant_priv ENUM
References_priv ENUM
Index_priv ENUM
Alter_priv ENUM
Create_tmp_table_priv ENUM
Lock_tables_priv ENUM
Create_view_priv ENUM
Show_view_priv ENUM
Create_routine_priv ENUM
Alter_routine_priv ENUM
Execute_priv ENUM
Trigger_priv ENUM
tables_priv
Host CHAR
Db CHAR
User CHAR
Table_name CHAR
Grantor CHAR
Timestamp TIMESTAMP
Table_priv SET
Column_priv SET
columns_priv
Host CHAR
Db CHAR
User CHAR
Table_name CHAR
Column_name CHAR
Timestamp TIMESTAMP
Column_priv SET

**1.** Таблица **user** определяет, разрешить соединение или отказать в нем.

Любые привилегии из таблицы user означают **глобальные привилегии** пользователя (привилегии администратора).

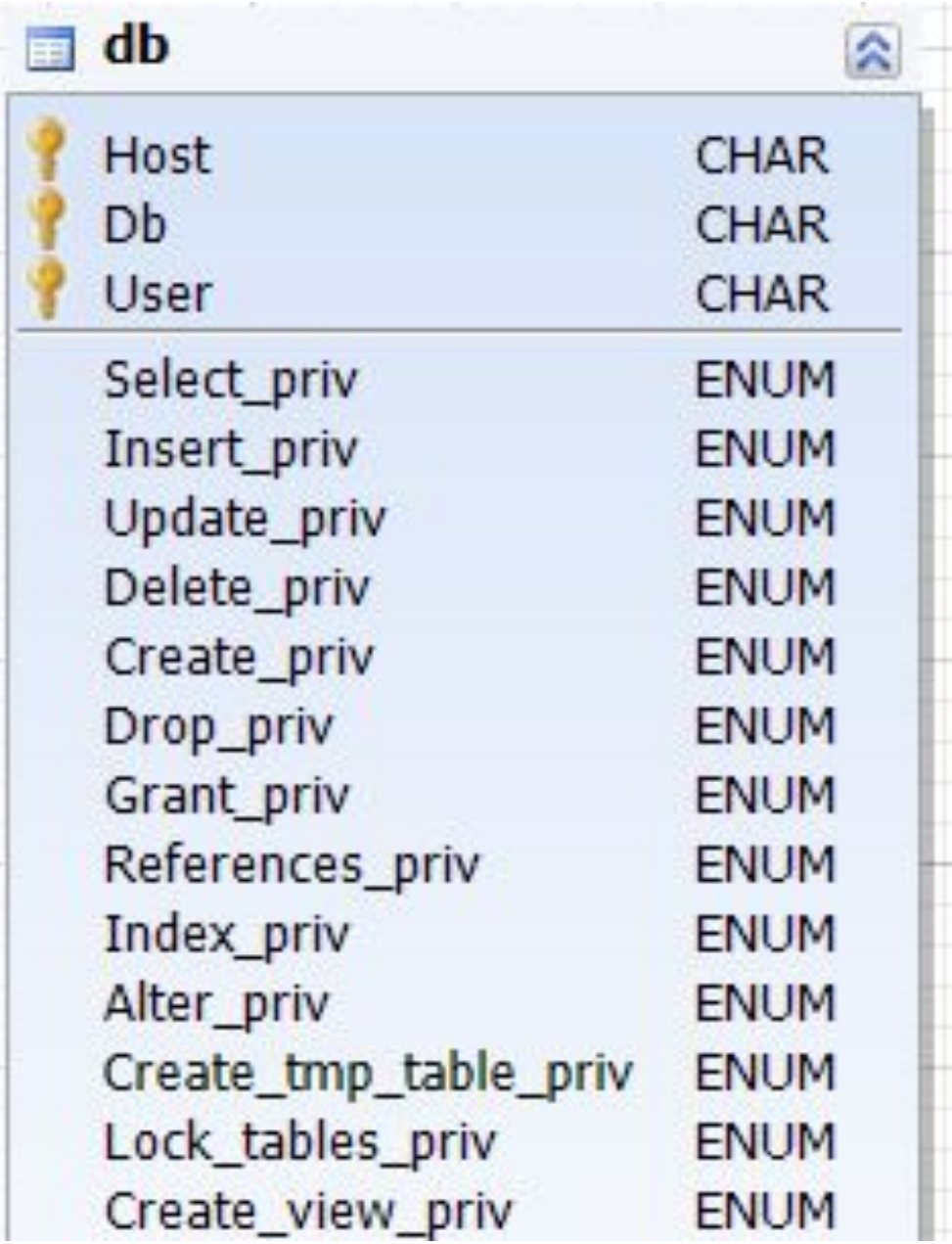
Эти привилегии распространяются на **все базы данных**, размещенные на сервере.



**2.** Таблицы **db** и **host** используются совместно:

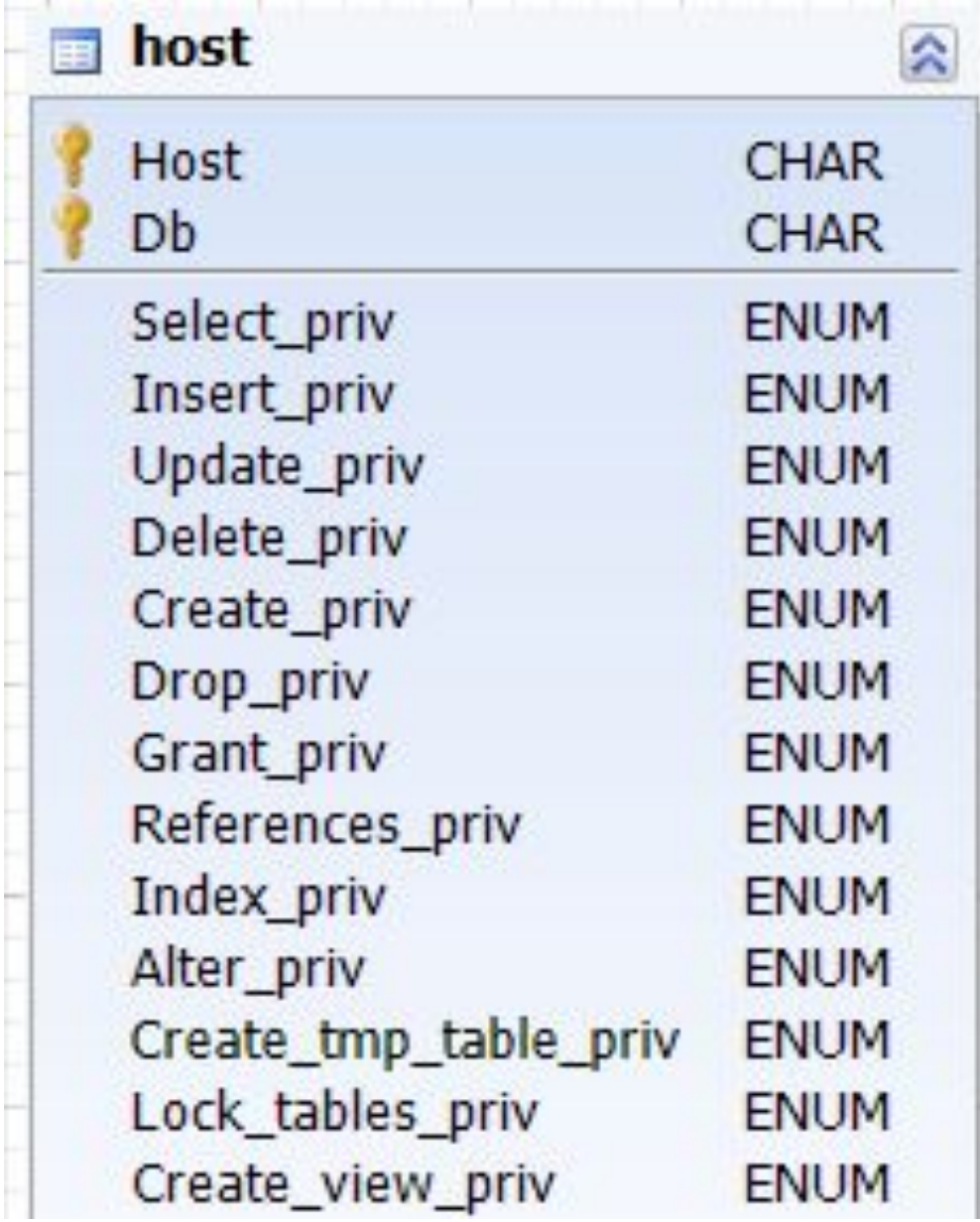
**a)** Таблица **db** определяет, каким пользователям, при подсоединении с каких хостов разрешен доступ к каким базам данных.

Поля привилегий определяют разрешенные операции.



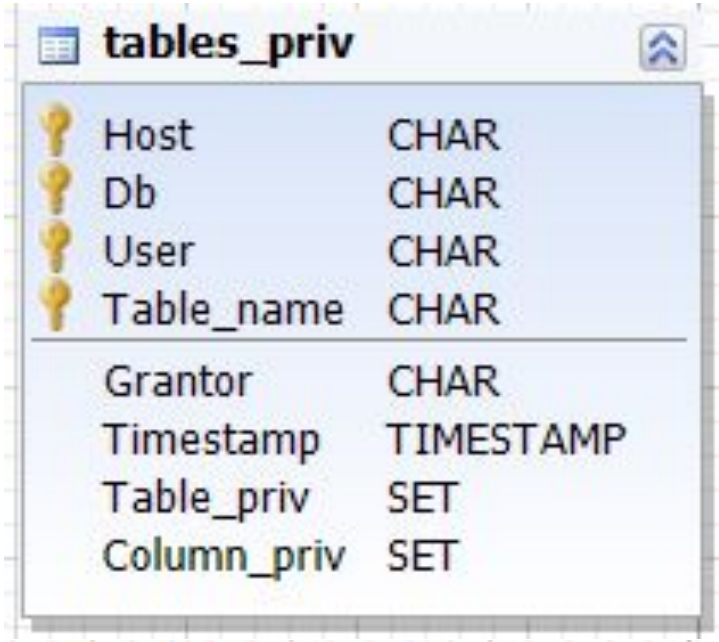
**б)** Таблица **host** используется в качестве расширения таблицы **db** в случае, если необходимо применить некоторую запись из таблицы **db** к разным хостам.

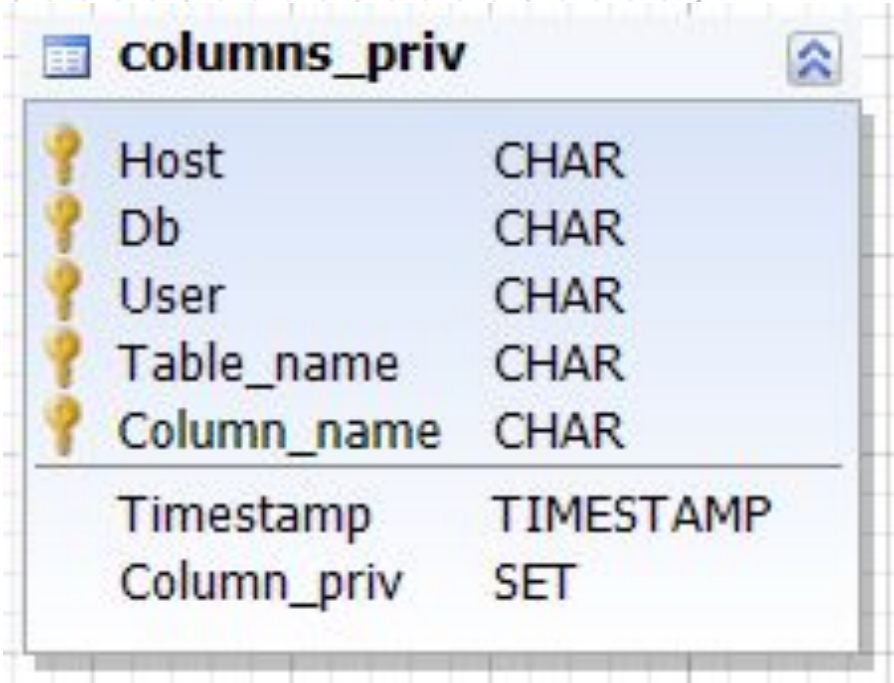
Если вы хотите предоставить пользователю возможность обращаться к базе данных с различных хостов сети, оставьте пустым поле **host** в записи этого пользователя в таблице db, а затем внесите в таблицу host запись для каждого из этих хостов.



**3.** Таблицы **tables_priv** и **columns_priv**

подобны таблице db, но областью их действия является уже уровень таблиц и столбцов, а не баз данных.



**Примечание 1.** Привилегии администрирования (RELOAD, SHUTDOWN и т.д..) задаются только в таблице user. Это связано с тем, что операции администрирования являются операциями над самим сервером, а не над базами данных, поэтому не смысла перечислять такие привилегии в других таблицах привилегий. Фактически для того, чтобы выяснить, имеете ли вы привилегии выполнять операции администрирования, достаточно обратиться только к таблице user.

Привилегия **FILE** также задается только в таблице user.

**Примечание 2.** Сервер считывает содержимое таблиц привилегий единожды, при его запуске.

# 1.2. Общие принципы обеспечения безопасности

При работе в MySQL старайтесь следовать приведенным ниже **инструкциям** (это самый краткий перечень):

**1.** Не предоставляйте никому (за исключением пользователя под именем root или его аналога) доступа к таблицам в базе данных **mysql**.

**Это чрезвычайно важно.**

**2.** Предоставляйте пользователям ровно столько прав, сколько необходимо, и не больше.

Полезно проводить следующие контрольные проверки:

- Выполните команду **mysql -u root**. Если удается успешно установить соединение с сервером без получения запроса пароля, значит, у вас имеются проблемы. Это означает, что кто угодно может подсоединиться к вашему серверу MySQL как клиент MySQL под именем root, получая таким образом право неограниченного доступа.

- С помощью команды SHOW GRANTS проверьте, кто и к каким ресурсам имеет доступ. Воспользуйтесь командой REVOKE, отмените права доступа, которые не являются необходимыми.

**3.** Не используйте в качестве пароля слова из словарей.

Для взлома такого рода паролей имеются специальные программы. Даже слова типа **petrov1998**- это очень плохие пароли. Лучше **owreic0886**: здесь используется то же слово petrov1998, но при этом буквы в нем заменены ближайшими к ним слева буквами клавиатуры. Еще один метод - составить парольное слово из первых букв слов какого либо словосочетания (аббревиатуру). Назгадать его тому, кто не знает ключевой фразы, будет непросто.

4. Установите брандмауэр, если его нет. Эта мера обеспечит защиту как минимум от половины всех видов несанкционированного использования любого ПО, с которым вы работаете. Разместите MySQL за брандмауэром.

5. Не доверяйте никаким данным и запросам, которые вводят пользователи. Возможны попытки перехитрить вашу программу путем ввода последовательностей специальных или экранированных символов в веб-формы, URL-ы или любое приложение, созданное вами. Убедитесь, что защита вашего приложения не будет нарушена, если пользователь введет что-нибудь типа

**DROP DATABASE mysql;**

Это крайний случай, но действия хакеров, использующих подобную технологию, могут привести к потере информации и появлению брешей в системе безопасности, если вы не готовы к ним.

# 1.3. Привилегии MySQL

Информация о привилегиях пользователя хранится в таблицах user, db, host, tables_priv и columns_priv базы данных mysql (в базе данных с именем mysql).

Сервер MySQL считывает содержимое этих таблиц во время запуска. Для обновления привилегий используется команда

**FLUSH PRIVILEGES;**

Некоторые привилегии:

**DELETE** Delete_priv

**INDEX** Index_priv

**INSERT** Insert_priv

**SELECT** Select_priv

**UPDATE** Update_priv

**CREATE** Create_priv Создавать базы данных или таблицы

**DROP** Drop_priv Удалять базы данных или таблицы

**GRANT** Grant_priv Раздавать привилегии

**FILE** File_priv Доступ к файлам на сервере

**RELOAD** Reload_priv Перезагрузка сервера

**SHOW DATABASES** Show_db_priv Список баз данных

**SHUTDOWN** Shutdown_priv Остановка сервера

…

Привилегии SELECT, INSERT, UPDATE и DELETE позволяют выполнять операции над строками таблиц.

Для операторов SELECT привилегия SELECT требуется только в том случае, если они действительно извлекают строки из таблицы.. В ряде случае можно выполнять операторы SELECT, даже не имея разрешения на доступ ни к одной базе данных на сервере. Например: клиент mysql вы можете использовать в качестве обычного калькулятора:

```
mysql> SELECT 2*2;
```

Привилегия **GRANT** позволяет вам предоставлять другим пользователям привилегии, которыми обладаете вы сами (подробно позднее).

Привилегия **FILE** дает вам право читать и записывать файлы на сервере с помощью операторов

**LOAD DATA INFILE**

и

**SELECT ... INTO OUTFILE**

Любой пользователь, которому предоставлена такая привилегия, имеет право прочитать или записать любой файл, который может прочитать или записать сервер MySQL. Привилегия FILE может использоваться **злонамеренно** для считывания любого доступного файла, хранящегося на сервере, или любого файла в каталоге текущей базы данных.

# 2. Соединение с сервером MySQL

Для получения доступа к MySQL - серверу необходимо сообщить клиентской программе следующие параметры соединения:

- хост, с которого происходит соединение;
- имя пользователя;
- пароль.

Например, клиент **mysql.exe** можно запустить следующим образом (необязательные аргументы заключены в квадратные скобки "[" и "]"):

**mysql [-h host_name] [-u user_name] [-pyour_pass] [database] [-Pport]**

Альтернативной формой опций -h, -u, и -p являются

- --host=host_name,
- --user=user_name и
- --password=your_pass.

Между **-p** или **--password=** и следующим за ними паролем **пробела нет**.

**Внимание.** Указывать пароль в командной строке небезопасно! Многие пользователи в вашей системе могут впоследствии отыскать ваш пароль.
Лучше писать просто **–p**.

В mysql используются следующие **значения по умолчанию** для параметров, пропущенных в командной строке:

- Значением по умолчанию для имени хоста является **localhost**.
- Значением по умолчанию для имени пользователя является ваш **Unix-аккаунт**.
- При отсутствии префикса «-p» никакого пароля не указывается.

Таким образом, для Unix-пользователя **jo** следующие команды являются эквивалентными:

```
shell> mysql -h localhost -u jo
shell> mysql -h localhost
shell> mysql -u jo
shell> mysql
```

# 2.1. Управление доступом, этап 1: контроль соединения

При попытке соединения с сервером MySQL он либо устанавливает соединение, либо отказывает в нем. В случае успеха сервер устанавливает соединение, и ожидает запросов.

Личность пользователь определяется двумя порциями информации:

- хостом, с которого вы соединяетесь;
- вашим именем пользователя MySQL .

Проверка пользователя осуществляется с помощью трех полей таблицы user (Host, User и Password).

## Значения в полях таблицы user могут задаваться следующим образом:

1. В поле **Host** может указываться **имя хоста**, либо его **IP-адрес**, либо **localhost** для обозначения локального хоста.

## Значения в полях таблицы user могут задаваться следующим образом:

1. В поле **Host** может указываться **имя хоста**, либо его **IP-адрес**, либо **localhost** для обозначения локального хоста.
2. В поле **Host** разрешается использовать шаблонные символы **%** и **_** (знак подчеркивания).

   **Пример**. **%.am.tpu.ru** – любой компьютер кафедры

## Значения в полях таблицы user могут задаваться следующим образом:

1. В поле **Host** может указываться **имя хоста**, либо его **IP-адрес**, либо **localhost** для обозначения локального хоста.
2. В поле **Host** разрешается использовать шаблонные символы **%** и **_** (знак подчеркивания).

   **Пример**. **%.am.tpu.ru** – любой компьютер кафедры
3. Значение "**%**" в поле **Host** означает любое имя хоста.

**Значения в полях таблицы user могут задаваться следующим образом:**

1. В поле **Host** может указываться **имя хоста**, либо его **IP-адрес**, либо **localhost** для обозначения локального хоста.
2. В поле **Host** разрешается использовать шаблонные символы **%** и **_** (знак подчеркивания).

   **Пример**. **%.am.tpu.ru** – любой компьютер кафедры
3. Значение "**%**" в поле **Host** означает любое имя хоста.
4. В поле **User** запрещено использовать шаблонные символы, но пустое значение разрешено, и оно соответствует **любому** имени. Если запись в таблице user, соответствующая входящему подсоединению, содержит пустое имя пользователя, **данный пользователь считается анонимным пользователем** (пользователем без имени). **Пустое имя не рекомендуется.**

5. Поле **Password** может быть пустым. Это не означает, что в данном случае подходит любой пароль. Это означает, что создан **пользователь без пароля**. **Создание пользователя без пароля не рекомендуется.** **Можно использовать не в сети Internet для отладки чего либо.**

5. Поле **Password** может быть пустым. Это не означает, что в данном случае подходит любой пароль. Это означает, что создан **пользователь без пароля**. **Создание пользователя без пароля не рекомендуется. Можно использовать не в сети Internet для отладки чего либо.**

6. Непустые значения в поле Password представляют собой зашифрованные пароли. В MySQL пароль шифруется. Заметим, что MySQL считает зашифрованный пароль **РЕАЛЬНЫМ** паролем, поэтому не следует допускать к нему кого бы то ни было! В частности, не разрешайте обычным пользователям доступ для чтения к таблицам в базе mysql!

**Пример 1.** MySQL – сервер ПМ. Фрагмент таблицы USER

| Host | User | Password | Select_priv | Ins |
|---|---|---|---|---|
| localhost | root | *47B84A4D20181854AE4D3D31... | Y | Y |
| % | stud | *B41B7209286C581DD12D78A3... | N | N |
| localhost | stud | *B41B7209286C581DD12D78A3... | N | N |
| % | root | *47B84A4D20181854AE4D3D31... | Y | Y |

**Пример 2.** Показывает, каким входящим соединениям соответствуют различные комбинации значений, указанных в полях **Host** и **User** таблицы USER:

| HOST | USER | Соединение |
|---|---|---|
| "am.tpu.ru" | "tom" | |
| "am.tpu.ru" | "" | |
| "%" | "tom" | |
| "%" | "" | |
| "%.ru" | "tom" | |
| "195.208.170.60" | "tom" | |
| "195.208.170.%" | "tom" | |

| HOST | USER | Соединение |
|---|---|---|
| "am.tpu.ru" | "tom" | tom, подключающийся с "am.tpu.ru" |
| "am.tpu.ru" | "" | Любой пользователь, подключающийся с "am.tpu.ru" |
| "%" | "tom" | tom, подключающийся с любого хоста |
| "%" | "" | Любой пользователь с любого хоста |
| "%.ru" | "tom" | tom, подключающийся с любого хоста домена ru |
| "195.208.170.60" | "tom" | tom, подключающийся с IP-адреса 195.208.170.60 |
| "195.208.170.%" | "tom" | tom, подключающийся с любого хоста в подсети 195.208.170 |

Входящее соединение может совпадать с несколькими записями в таблице **user**.

Например, соединению с am.tpu.ru пользователем **tom** могут подходить разные записи.

Каким образом сервер определяет, какую из записей использовать, при совпадении с более чем одной из них:

После считывания таблицы user во время запуска сервер производит ее **сортировку**, а затем, когда пользователь пытается установить соединение, **записи таблицы просматриваются в порядке их упорядочения**.

Используется **первая** подошедшая запись.

Сортировка таблицы user осуществляется следующим образом. Предположим, таблица **user** имеет следующий вид:

```
+-----------+-----------+
| Host      | User      | ...
+-----------+-----------+
| %         | root      | ...
| %         | mot       | ...
| localhost | root      | ...
| localhost |           | ...
+-----------+-----------+
```

При считывании этой таблицы сервер упорядочивает записи, начиная с наиболее конкретных значений в столбце Host.

**"%"** в столбце Host означает ""любой хост"" и является наименее конкретным.

Записи с одинаковым значением в столбце Host упорядочиваются между собой начиная с наиболее конкретных значений в столбце User.

Пустое значение в столбце User означает "любой пользователь"" и является наименее конкретным.

Окончательно отсортированная таблица user имеет следующий вид:

Окончательно отсортированная таблица user имеет следующий вид:

```
+-----------+-----------+
| Host      | User      | ...
+-----------+-----------+
| localhost | root      | ...
| localhost |           | ...
| %         | mot       | ...
| %         | root      | ...
+-----------+-----------+
```

При попытке соединения сервер просматривает отсортированные записи и использует первую подходящую запись.

Для подсоединения с **localhost** пользователя **mot** первыми подходящими записями являются записи со значением "localhost" в столбце Host. Из них имени пользователя соответствует запись с пустым значением имени пользователя.

Запись **"%" "mot"** тоже подошла бы, но она -- не первая подходящая в этой таблице).

**Пример.** Таблица user:

| Host | User | ... |
|---|---|---|
| % | mot | ... |
| am.tpu.ru | | ... |

Отсортированная таблица выглядит следующим образом:

Отсортированная таблица выглядит следующим образом:

| Host | User | ... |
|---|---|---|
| am.tpu.ru | | ... |
| % | mot | ... |

Для подсоединения пользователя **mot** с **am.tpu.ru** подходит первая запись.

**Распространенное заблуждение:** иногда думают, что при поиске записей для данного имени пользователя, соответствующих определенному подсоединению, сервер первыми будет использовать записи, в которых этот пользователь указан явно.

Это неверно: для подсоединения пользователя mot с am.tpu.ru первой подходящей записью является не запись, содержащая значение mot в поле User, а запись, не содержащая имени пользователя вообще.

Если у вас возникают проблемы с подсоединением к серверу, выведите таблицу user и отсортируйте ее вручную, чтобы увидеть, где происходит первое совпадение.

Если соединение было успешно, но ваши привилегии не такие, которые ожидались, можно использовать функцию

**CURRENT_USER()**

чтобы узнать, какой комбинации **пользователь/компьютер** соединение соответствует.

# 2.2. Управление доступом, этап 2: контроль запросов

После установления соединения сервер приступает к выполнению второго этапа.

Для каждого поступающего запроса сервер проверяет, имеется ли достаточно привилегий для его выполнения, учитывая при этом на тип операции, которую необходимо выполнить.

Информация о привилегиях может находиться в любой из таблиц привилегий - user, db, host, tables_priv или columns_priv.

Изменение привилегий в этих таблиц осуществляется с помощью команд GRANT и REVOKE.

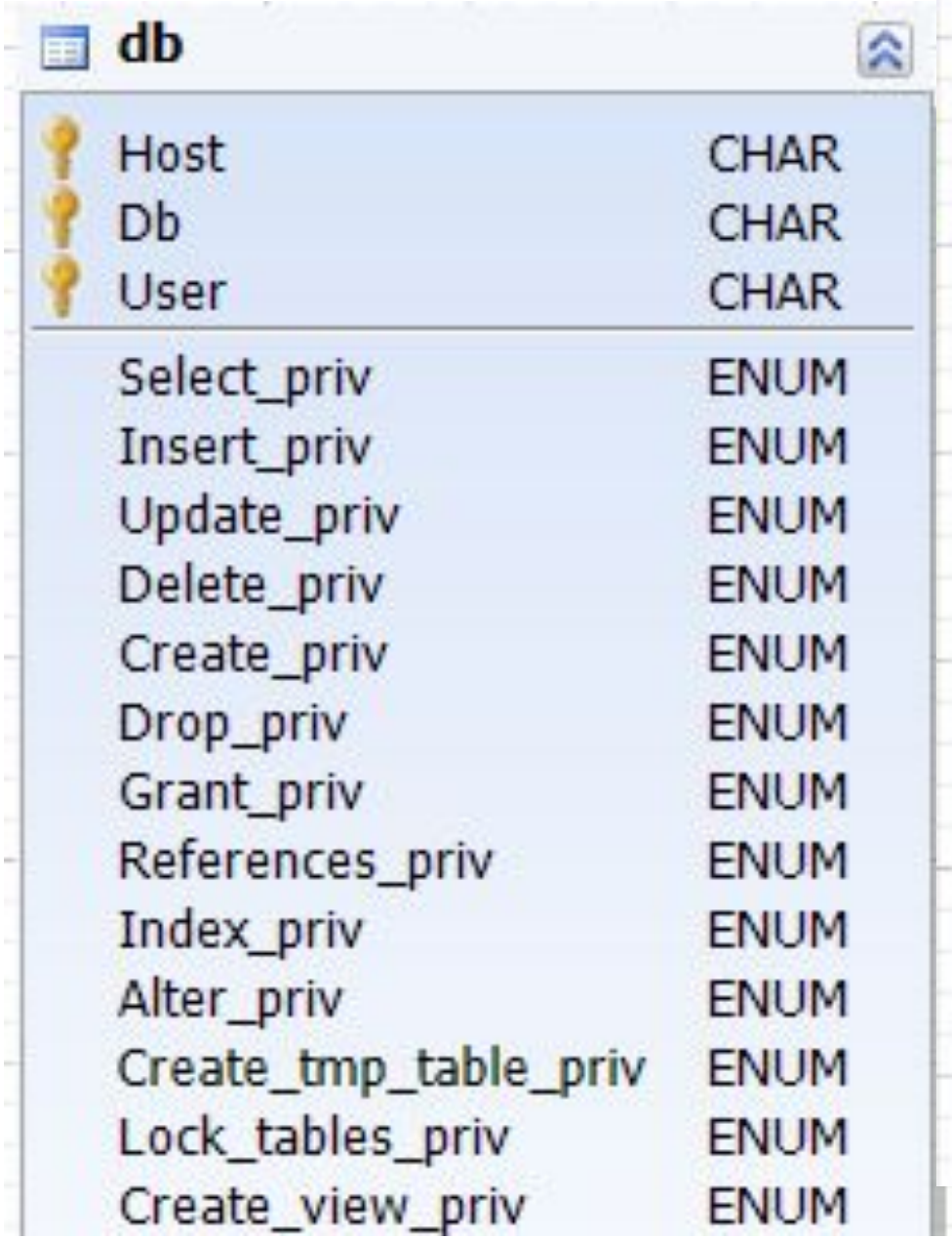
Таблица **user** предоставляет глобальные привилегии, которые действуют на все базы данных.

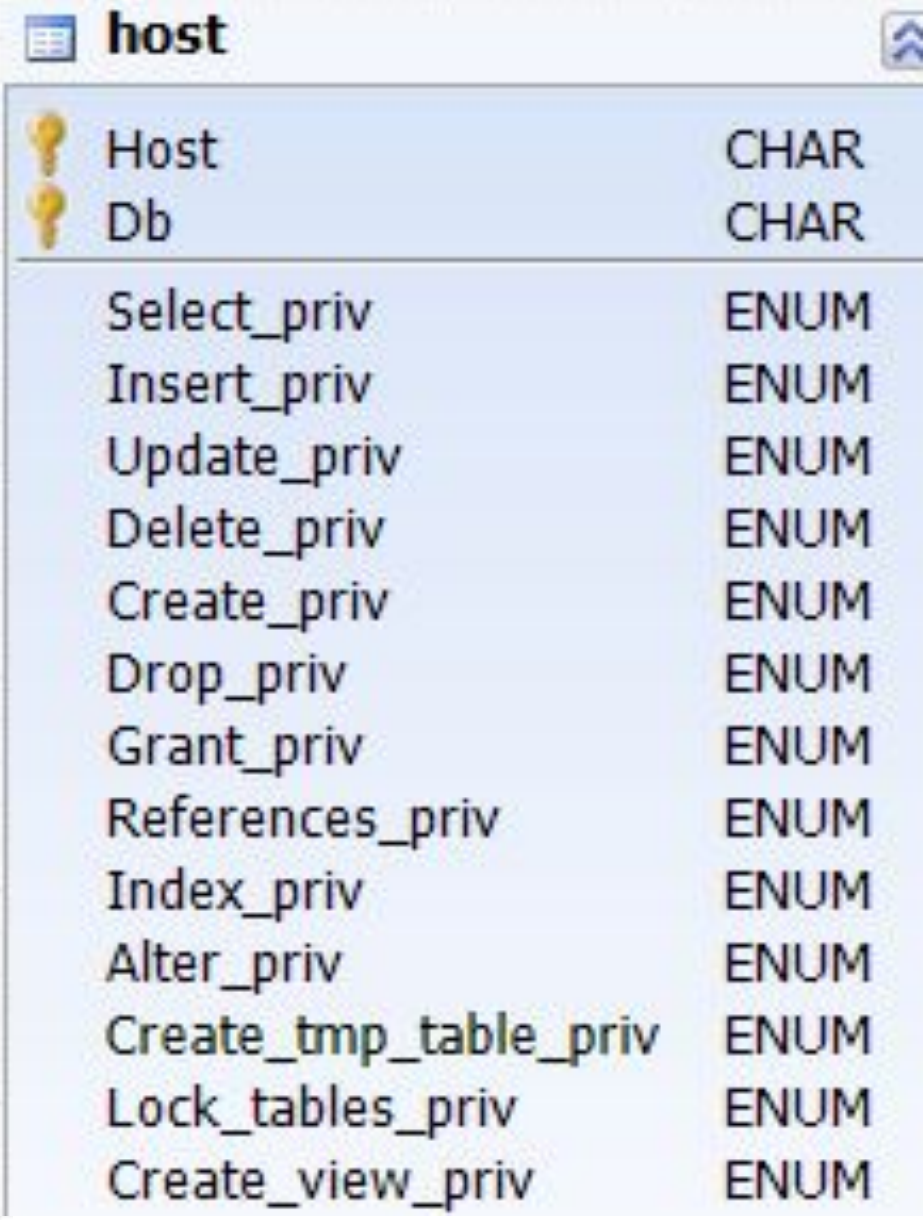
Например, если таблица user предоставляет пользователю привилегию **DELETE**, он может удалять строки из любой таблицы любой базы данных.

Поэтому **целесообразно предоставлять привилегии в таблице user только администраторам сервера или администраторам баз данных**.

У других пользователей для всех привилегий в таблице **user** следует установить значение **"N"** и предоставлять им привилегии только на уровне баз данных, используя для этого таблицы **db** и **host**.

Значения в таблицах **db** и **host** могут задаваться следующим образом:


db
Host CHAR
Db CHAR
User CHAR
Select_priv ENUM
Insert_priv ENUM
Update_priv ENUM
Delete_priv ENUM
Create_priv ENUM
Drop_priv ENUM
Grant_priv ENUM
References_priv ENUM
Index_priv ENUM
Alter_priv ENUM
Create_tmp_table_priv ENUM
Lock_tables_priv ENUM
Create_view_priv ENUM


host
Host CHAR
Db CHAR
Select_priv ENUM
Insert_priv ENUM
Update_priv ENUM
Delete_priv ENUM
Create_priv ENUM
Drop_priv ENUM
Grant_priv ENUM
References_priv ENUM
Index_priv ENUM
Alter_priv ENUM
Create_tmp_table_priv ENUM
Lock_tables_priv ENUM
Create_view_priv ENUM

1. Шаблонный символ "**%**" может использоваться в полях Host и Db любой таблицы.
2. Значение "**%**" в колонке Host таблицы db означает "любой хост". Пустое значение в поле Host таблицы db означает "за дополнительной информацией следует обратиться к таблице host".
3. Значение "**%**" или пустое значение в поле Host таблицы host означает "любой хост".
4. Значение "**%**" или пустое значение в поле Db любой из таблиц означает "любая база данных".
5. Пустое значение в поле User любой из таблиц соответствует анонимному пользователю.

Таблицы **db** и **host** считываются и сортируются при запуске сервера.

Таблица **db** сортируется по полям Host, Db и User, а таблица **host** - по полям Host и Db.

При сортировке первыми отбираются **наиболее конкретные** значения, а последними - **наименее конкретные.**

Когда сервер производит поиск совпадающих записей, используется первая подходящая запись, которую он находит.

Таблицы tables_priv и columns_priv предоставляют привилегии на уровне таблиц и столбцов.

Их работа **аналогична** предыдущему:

(шаблонные символы пустые значения, сортировка и т.д.)

# Процесс контроля

Для запросов на **администрирование** сервер проверяет только строки в таблице user.

Доступ предоставляется при условии, что выбранная строка разрешает затребованные операции, и запрещается в противном случае.

Например, вы хотите завершить работу mysql сервера с помощью команды **mysqladmin shutdown**, но ваша запись в таблице user не предоставляет вам привилегию **SHUTDOWN**.

В доступе будет отказано без дальнейшей проверки таблицы db и host.

В случае **запросов, относящихся к базам данных** (INSERT, UPDATE и т.д.), сервер сначала проверяет глобальные привилегии пользователя (привилегии суперпользователя), просматривая запись в таблице user.

Если эта запись разрешает затребованную операцию, доступ предоставляется.

Если глобальные привилегии, указанные в таблице user, недостаточны, сервер проверяет таблицы db и host и определяет привилегии пользователя на уровне баз данных.

Определив привилегии на уровне базы данных, предоставляемые записями в таблицах db и host, сервер **добавляет** их к глобальным привилегиям, заданным в таблице user.

Если в результате привилегий оказывается достаточно для выполнения затребованной операции, доступ предоставляется.

В противном случае сервер проверяет по таблицам **tables_priv** и **columns_priv** привилегии пользователя на уровне таблиц и столбцов и добавляет их к уже имеющимся привилегиям.

В зависимости от полученного результата доступ либо предоставляется, либо нет.