5. Управление учетными записями пользователей MySQL – сервера

1. Синтаксис команд GRANT и REVOKE

Команды **GRANT** и **REVOKE** позволяют системным администраторам создавать пользователей MySQL, а также предоставлять права пользователям или лишать их прав на четырех уровнях привилегий:

Глобальный уровень

Глобальные привилегии применяются ко всем базам данных на указанном сервере. Эти привилегии хранятся в таблице mysql.user.

Уровень базы данных

Привилегии базы данных применяются ко всем таблицам указанной базы данных. Эти привилегии хранятся в таблицах mysql.db и mysql.host.

Уровень таблицы

Привилегии таблицы применяются ко всем столбцам указанной таблицы. Эти привилегии хранятся в таблице mysql.tables_priv.

Уровень столбца

Привилегии столбца применяются к отдельным столбцам указанной таблицы. Эти привилегии хранятся в таблице mysql.columns_priv.

```
GRANT priv_type [(column_list)] [,priv_type
[(column_list)]...]
    ON {tbl_name | * | *.* | db_name.*}
    TO user_name [IDENTIFIED BY 'password']
        [, user_name ...]
    [WITH GRANT OPTION]

REVOKE priv_type [(column_list)] [,priv_type
[(column_list)]...]
    ON {tbl_name | * | *.* | db_name.*}
    FROM user_name [, user_name ...]
```

Если привилегии предоставляются пользователю, которого не существует, то этот пользователь создается.

Список возможных значений параметра priv_type:

ALL [PRIVILEGES]	Задает все простые привилегии, кроме WITH GRANT OPTION
ALTER	Разрешает использование ALTER TABLE
CREATE	Разрешает использование CREATE TABLE
CREATE TEMPORARY TABLES	Разрешает использование CREATE TEMPORARY TABLE
DELETE	Разрешает использование DELETE
DROP	Разрешает использование DROP TABLE
EXECUTE	Разрешает пользователю запускать хранимые процедуры (для MySQL 5.0)
FILE	Разрешает использование SELECT ... INTO OUTFILE и LOAD DATA INFILE
INDEX	Разрешает использование CREATE INDEX and DROP INDEX
INSERT	Разрешает использование INSERT
LOCK TABLES	Разрешает использование LOCK TABLES на таблицах, для которых есть привилегия SELECT
PROCESS	Разрешает использование SHOW FULL PROCESSLIST

REFERENCES	Зарезервировано для использования в будущем
RELOAD	Разрешает использование FLUSH
REPLICATION CLIENT	Предоставляет пользователю право запрашивать местонахождение головного и подчиненных серверов
REPLICATION SLAVE	Необходимо для подчиненных серверов при репликации (для чтения информации из бинарных журналов головного сервера)
SELECT	Разрешает использование SELECT
SHOW DATABASES	SHOW DATABASES выводит все базы данных
SHUTDOWN	Разрешает использование mysqladmin shutdown
SUPER	Позволяет установить одно соединение (один раз), даже если достигнуто значение max_connections, и запускать команды CHANGE MASTER, KILL thread, mysqladmin debug, PURGE MASTER LOGS и SET GLOBAL
UPDATE	Разрешает использование UPDATE
USAGE	Синоним для "без привилегий"
GRANT OPTION	Синоним для WITH GRANT OPTION

Для таблицы можно указать только следующие значения priv_type: SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, GRANT OPTION, INDEX и ALTER.

Для столбца можно указать только следующие значения priv_type (при использовании оператора column_list): SELECT, INSERT и UPDATE.

Глобальные привилегии можно задать, воспользовавшись синтаксисом **ON *.***, а привилегии базы данных - при помощи синтаксиса **ON db_name.***.

Если указать **ON * при открытой текущей базе данных**, то привилегии будут заданы для этой базы данных. (**Предупреждение:** если указать ON * при отсутствии открытой текущей базы данных, это повлияет на глобальные привилегии)

Примечание. В операторе GRANT шаблонные символы '%' не допускаются в определении имени баз данных.

Имя пользователя (user_name) можно указать в форме **user@host**. Если необходимо указать строку **user**, в которой содержатся специальные символы (такие как '-') или строку **host**, в которой содержатся специальные или групповые символы (такие как '%'), необходимо заключить имя удаленного компьютера или пользователя в кавычки (например, 'test-user'@'test-hostname').

В имени удаленного компьютера также можно указывать групповые символы. Например, user@'%.loc.gre' относится к user всех удаленных компьютеров домена loc.gre, а user@'144.155.166.%' относится к user всех удаленных компьютеров подсети 144.155.166.

Простая форма user является синонимом для user@"%".

Привилегии для таблицы или столбца формируются при помощи логического оператора OR из привилегий каждого из четырех уровней. Например, если в таблице mysql.user указано, что у пользователя есть глобальная привилегия SELECT, эта привилегия не отменяется на уровне базы данных, таблицы или столбца.

Привилегии для столбца могут быть вычислены следующим образом:

глобальные привилегии
OR привилегии базы данных
OR привилегии таблицы
OR привилегии столбца

В большинстве случаев права пользователя определяются только на одном уровне привилегий, поэтому обычно эта процедура не настолько сложна, как описано выше.

Если привилегии предоставляются сочетанию пользователь@удаленный компьютер, которое отсутствует в таблице mysql.user, то в последнюю добавляется запись, которая остается в таблице до тех пор, пока не будет удалена при помощи команды DELETE.

Т.е. команда GRANT может создавать записи user в таблице, но команда REVOKE не может их удалить. Это необходимо делать при помощи команды DELETE.

Пароль пользователя будет назначаться оператором IDENTIFIED BY, если он указан. Если у пользователя уже есть пароль, то этот пароль будет заменен новым.

Если при создании нового пользователя не указать оператор IDENTIFIED BY, будет создан пользователь без пароля. Это ненадежно с точки зрения безопасности.

Пароли также можно задавать при помощи команды SET PASSWORD.

```
SET PASSWORD FOR root@localhost =
PASSWORD('new_password');
```

Если у пользователя нет никаких привилегий для таблицы, то таблица не отображается, когда пользователь запрашивает список таблиц (например, при помощи оператора SHOW TABLES).

Оператор **WITH GRANT OPTION** предоставляет пользователю возможность наделять других пользователей любыми привилегиями, которые он сам имеет на указанном уровне привилегий. При предоставлении привилегии GRANT необходимо проявлять осмотрительность, так как два пользователя с разными привилегиями могут объединить свои привилегии!

Нельзя предоставить другому пользователю привилегию, которой нет у вас самого. Привилегия GRANT позволяет предоставлять только те привилегии, которыми вы обладаете.

Учтите, что если пользователю назначена привилегия GRANT на определенном уровне привилегий, то все привилегии, которыми этот пользователь уже обладает (или которые будут ему назначены в будущем!) на этом уровне, также могут назначаться этим пользователем.

Предположим, пользователю назначена привилегия INSERT в базе данных. Если потом в базе данных назначить привилегию SELECT и указать WITH GRANT OPTION, пользователь сможет назначать не только привилегию SELECT, но также и INSERT. Если затем в базе данных предоставить пользователю привилегию UPDATE, пользователь сможет после этого назначать INSERT, SELECT и UPDATE.

Не следует назначать привилегии ALTER обычным пользователям. Это дает пользователю возможность разрушить систему привилегий путем переименования таблиц!

Изменения в таблицах назначения привилегий, которые осуществляются при помощи команд GRANT и REVOKE, обрабатываются сервером немедленно. Если изменять таблицы назначения привилегий вручную (используя команды INSERT, UPDATE и т.д.), необходимо выполнить оператор **FLUSH PRIVILEGES** или **mysqladmin flush-privileges**, чтобы указать серверу на необходимость перезагрузки таблиц назначения привилегий.

2. Добавление новых пользователей в MySQL

Пользователей можно добавлять тремя различными способами:

1) при помощи команды CREATE USER;
2) при помощи команды GRANT;
3) напрямую в таблицы назначения привилегий.

Предпочтительнее использовать команду GRANT - этот способ проще и дает меньше ошибок.

Существует также большое количество программ (таких как phpmyadmin), которые служат для создания и администрирования пользователей.

Команда CREATE USER

CREATE USER user [IDENTIFIED BY [PASSWORD] 'password'] [, user [IDENTIFIED BY [PASSWORD] 'password']] .

The CREATE USER statement creates new MySQL accounts. To use it, you must have the global CREATE USER privilege or the INSERT privilege for the mysql database.

For each account, CREATE USER creates a new row in the mysql.user table that has no privileges. ***An error occurs if the account already exists***.

Each account is named using the same format as for the GRANT statement; for example, 'jeffrey'@'localhost'. The user and host parts of the account name correspond to the User and Host column values of the user table row for the account.

Команда GRANT

```
mysql> GRANT ALL PRIVILEGES ON *.* TO monty@localhost
    -> IDENTIFIED BY 'some_pass' WITH GRANT OPTION;

mysql> GRANT ALL PRIVILEGES ON *.* TO monty@"%"
    -> IDENTIFIED BY 'some_pass' WITH GRANT OPTION;

mysql> GRANT RELOAD,PROCESS ON *.*
    -> TO admin@localhost;

mysql> GRANT USAGE ON *.* TO dummy@localhost;
```

Эти команды GRANT создают **трех** новых пользователей.

monty

Полноценный суперпользователь - он может подсоединяться к серверу откуда угодно, но должен использовать для этого пароль some_pass. Обратите внимание на то, что мы должны применить операторы GRANT как для monty@localhost, так и для monty@"%". Если не добавить запись с localhost, запись анонимного пользователя для localhost, которая создается при помощи mysql_install_db, будет иметь преимущество при подсоединении с локального компьютера, так как в ней указано более определенное значение для поля Host, и она расположена раньше в таблице user.

admin

Пользователь, который может подсоединяться с localhost без пароля; ему назначены административные привилегии RELOAD и PROCESS. Эти привилегии позволяют пользователю запускать команды mysqladmin reload, mysqladmin refresh и mysqladmin flush-*, а также mysqladmin processlist. Ему не назначено никаких привилегий, относящихся к базам данных (их можно назначить позже, дополнительно применив оператор GRANT).

dummy

Пользователь, который может подсоединяться к серверу без пароля, но только с локального компьютера. Все глобальные привилегии установлены в значение 'N'-тип привилегии USAGE, который позволяет создавать пользователей без привилегий. Предполагается, что относящиеся к базам данных привилегии будут назначены позже.

Непосредственное создание пользователя при помощи команды INSERT

```
mysql> INSERT INTO user VALUES('localhost','monty',
    -> PASSWORD('some_pass'),
    ->  'Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y');

mysql> INSERT INTO user VALUES('%','monty',PASSWORD('some_pass'),
    -> 'Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y');

mysql> INSERT INTO user SET Host='localhost',User='admin',
    -> Reload_priv='Y', Process_priv='Y';

mysql> INSERT INTO user (Host,User,Password)
    -> VALUES('localhost','dummy','');

mysql> FLUSH PRIVILEGES;
```

Для пользователя admin может использоваться более удобочитаемый **расширенный синтаксис команды** INSERT.

Чтобы создать администратора, нет необходимости задавать значения в записях таблиц db или host.

Пример добавления пользователя

Добавляется пользователь custom, который может подсоединяться с компьютеров localhost, server.com и whitehouse.ru. Он должен получать доступ к базе данных bank только с компьютера localhost, к базе данных exp - только с whitehouse.ru, и к базе данных customer - со всех трех компьютеров, а также использовать пароль student при подсоединении со всех трех компьютеров.

```
mysql> GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,DROP
    ->     ON bank.*
    ->     TO custom@localhost
    ->     IDENTIFIED BY ' student';

mysql> GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,DROP
    ->     ON exp.*
    ->     TO custom@whitehouse.ru
    ->     IDENTIFIED BY ' student';

mysql> GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,DROP
    ->     ON customer.*
    ->     TO custom@'%'
    ->     IDENTIFIED BY 'student';
```

Примечание. Команда SHOW PRIVILEGES показывает список системных привилегий, которые поддерживаются сервером MySQL.

```
mysql> show privileges;
+-----------+---------------------------+---------------------------------
| Privilege | Context                   | Comment
+-----------+---------------------------+---------------------------------
| Select    | Tables                    | To retrieve rows from
| Insert    | Tables                    | To insert data into
| Update    | Tables                    | To update existing
| Delete    | Tables                    | To delete existing
| Index     | Tables                    | To create or drop
| Alter     | Tables                    | To alter the
| Create    | Databases,Tables,Indexes  | To create new databases and
| Drop      | Databases,Tables          | To drop databases and tables
| Grant     | Databases,Tables          | To give to other users those
                                          privileges you possess
| References| Databases,Tables          | To have references on tables
| Reload    | Server Admin              | To reload or refresh tables,
                                          logs and privileges
| Shutdown  | Server Admin              | To shutdown the
| Process   | Server Admin              | To view the plain text of
                                          currently executing queries
| File      | File access on server     | To read and write files on the
                                          server
+-----------+---------------------------+---------------------------------
14 rows in set (0.00 sec)
```

3. Администрирование базы данных

3.1. ИСПОЛЬЗУЕМЫЕ ТАБЛИЦЫ

Таблица 1. Продавцы

SNUM	SNAME	CITY	COMM
1001	Peel	London	.12
1002	Serres	San Jose	.13
1004	Motika	London	.11
1007	Rifkin	Barcelona	.15
1003	Axelrod	New York	.10

snum - номер продавца "номер служащего").
sname- имя продавца.
city - местонахождение продавца (город).
comm - комиссионные продавцов в десятичной форме.

Таблица 2. Заказчики

CNUM	CNAME	CITY	RATING	SNUM
2001	Hoffman	London	100	1001
2002	Giovanni	Rome	200	1003
2003	Liu	SanJose	200	1002
2004	Grass	Berlin	300	1002
2006	Clemens	London	100	1001
2008	Cisneros	SanJose	300	1007
2007	Pereira	Rome	100	1004

cnum - номер заказчика.
cname - имя заказчика.
city - местонахождение заказчика (город).
rating - код, указывающий уровень предпочтения данного заказчика перед другими.
snum - номер продавца, назначенного этому заказчику (из таблицы Продавцов).

Таблица 3. Заказы

ONUM	AMT	ODATE	CNUM	SNUM
3001	18.69	10/03/1990	2008	1007
3003	767.19	10/03/1990	2001	1001
3002	1900.10	10/03/1990	2007	1004
3005	5160.45	10/03/1990	2003	1002
3006	1098.16	10/03/1990	2008	1007
3009	1713.23	10/04/1990	2002	1003
3007	75.75	10/04/1990	2004	1002
3008	4723.00	10/05/1990	2006	1001
3010	1309.95	10/06/1990	2004	1002
3011	9891.88	10/06/1990	2006	1001

onum - уникальный номер, данный каждому приобретению.
amt - значение суммы приобретений.
odate - дата приобретения.

3.2. КОМАНДА GRANT

Предположим, что пользователь **Diane** имеет таблицу Заказчиков и хочет разрешить пользователю **Adrian** выполнить запрос к ней. Diane должна в этом случае ввести следующую команду:

```
GRANT SELECT ON Заказчики TO Adrian;
```

Теперь Adrian может выполнить запросы к таблице Заказчиков. Но не может предоставить право SELECT другому пользователю: таблица еще принадлежит Diane (далее мы покажем, как Diane может дать право Adrian предоставлять SELECT другим пользователям).

Группы привилегий и пользователей.

Допустимыми являются списки привилегий или пользователей, разделяемых запятыми.

Stephen может предоставить и SELECT, и INSERT в таблице Заказов для **Adrian** и **Diane**:

```
GRANT SELECT, INSERT ON Заказы TO Adrian, Diane;
```

Ограничение привилегий на определенные таблицы и столбцы

Если Adrian хочет ограничить Diane в изменении, например, комиссионных, он может ввести:

GRANT UPDATE (city, comm) ON Продавцы TO Diane;

Здесь в круглых скобках после имени таблицы указаны конкретные столбцы, к которым привилегия UPDATE должна быть применена.

Имена нескольких столбцов таблицы могут указываться в любом порядке, отделяемые запятыми:

3.3. Использование аргумента

ALL используется вместо имён привилегий в команде GRANT, чтобы передать все привилегии в таблице.

Например, **Diane** может выдать **Stephen** весь набор привилегий в таблице Заказчиков с помощью такой команды:

GRANT ALL ON Продавцы TO Diane;

(при этом привилегия UPDATE применяется ко всем столбцам.)

Конечно, можно предоставить любые или все привилегии каждому, но это, очевидно, нежелательно. Все привилегии, за исключением SELECT, позволяют пользователю изменять содержание таблицы. Разрешение всем пользователям изменять содержание ваших таблиц создаст проблемы.

3.4. Предоставление привилегий с помощью предложения «WITH GRANT OPTION»

Иногда владельцу таблицы необходимо, чтобы **пользователи могли передавать свои привилегии на таблицы другим пользователям**. Для этого используется предложение **WITH GRANT OPTION**.

Если бы **Diane** хотела, чтобы **Adrian** имел право предоставлять привилегию SELECT в таблице Заказчиков другим пользователям, она дала бы ему привилегию SELECT с использованием предложения WITH GRANT OPTION:

```
GRANT SELECT ON Заказчики TO Adrian  WITH GRANT OPTION;
```

После этого **Adrian** получил право передавать привилегию SELECT третьим лицам. Например, он может выдать команду

```
GRANT SELECT ON Diane.Заказчики TO Stephen;
```

или даже

```
GRANT SELECT ON Diane.Заказчики TO Stephen WITH GRANT OPTION;
```

3.5. Отмена привилегий

Удаление привилегии выполняется командой REVOKE.

Чтобы удалить привилегию INSERT для Adrian в таблице Заказов, вы можете ввести

REVOKE INSERT ON Заказы FROM Adrian;

Использование списков привилегий и пользователей здесь также допустимы, как и в случае с GRANT, так что вы можете ввести следующую команду:

REVOKE INSERT, DELETE ON Заказчики

FROM Adrian, Stephen;

Привилегии отменяются пользователем, который их предоставил, и отмена будет каскадироваться, то есть она будет автоматически распространяться на всех пользователей, получивших от него эту привилегию.

3.6. Использование представлений для фильтрации привилегий

Действие привилегий можно сделать более точными, используя представления.

Чтобы создать представление, необходимо иметь привилегию SELECT во всех таблицах, на которые ссылается представление.

Ограничение привилегии для определённых столбцов

Предположим, вы хотите дать пользователю **Claire** способность видеть только столбцы **snum** и **sname** таблицы Продавцов. Вы можете сделать это, поместив имена этих столбцов в представление

```
CREATE VIEW Clairesview
    AS SELECT snum, sname
    FROM Продавцы;
```

и предоставить Claire привилегию SELECT в представлении, а не в самой таблице Продавцов:

```
GRANT SELECT On Clairesview to Claire;
```

Ограничение привилегий для определённых строк

Более полезный пример - такое представление, чтобы привилегия относилась только к определенным строкам.

Чтобы предоставить пользователю **Adrian** привилегию UPDATE в таблице Заказчиков для всех заказчиков, размещенных в Лондоне, можно создать такое представление:

```
CREATE VIEW Londoncust
  AS SELECT *
    FROM Заказчики
    WHERE city = 'London'
    WITH CHECK OPTION;
```

Затем необходимо передать привилегию UPDATE в этой таблице для **Adrian**:

```
GRANT UPDATE ON Londoncust TO Adrian;
```

Предложение **WITH CHECK OPTION** предохраняет Adrian от замены значения поля **city** на любое значение кроме **London**.

Предоставление доступа только к извлечённым данным

Другая возможность состоит в том, чтобы предлагать пользователям доступ к уже извлечённым данным, а не к фактическим значениям в таблице.

Пример. Вы можете создавать представление, которое выдаёт подсчёт, среднее и общее количество заказов на каждый день:

```
CREATE VIEW Datetotals
  AS SELECT odate, COUNT (*), SUM (amt), AVG (amt)
    FROM Заказы
    GROUP BY odate;
```

Теперь вы передаёте пользователю Diane привилегию SELECT в представлении Datetotals:

```
GRANT SELECT ON Datetotals TO Diane;
```