

Создание, модификация объектов баз данных

Артамонов Ю.Н.

Типы данных

Стандарт языка SQL определяет ряд типов данных. Ниже перечислены наиболее часто используемые. Некоторые СУБД могут не поддерживать часть из перечисленных типов или наоборот - добавлять свои типы данных.

INTEGER, SMALLINT целые числа со знаком. Точность представления определяется реализацией, точность SMALLINT не должна превышать точность INTEGER. Многие СУБД позволяют сокращать запись INTEGER до INT.

NUMERIC (p,s), DECIMAL (p, s) задают в общем виде формат точного числа, целого или с дробной частью. NUMERIC представляет число с точностью p (число знаков) и масштабом s (число знаков после запятой). Если пропущен масштаб, то он полагается равным 0 (т. е. число целое), а если опущена точность, то ее значение по умолчанию определяется реализацией. Типы данных NUMERIC и DECIMAL реализованы в MySQL как один и тот же тип - это разрешается стандартом SQL92. Они используются для величин, для которых важно сохранить повышенную точность, например для денежных данных.

Величины типов DECIMAL и NUMERIC хранятся как строки, а не как двоичные числа с плавающей точкой, чтобы сохранить точность представления этих величин в десятичном виде. При этом используется по одному символу строки для каждого разряда хранимой величины, для десятичного знака (если масштаб > 0) и для знака минус (для отрицательных чисел).

FLOAT (p), REAL, DOUBLE PRECISION числа в представлении с плавающей точкой (приближенные числовые типы). Точность REAL определяется реализацией, точность DOUBLE PRECISION должна быть больше точности REAL. Точность FLOAT задается параметром p , если параметр отсутствует определяется реализацией. Примеры допустимых значений: 34, -11.23, 23.45E-15.

CHARACTER (n) строка из n символов, допускается сокращение CHAR (n). Запись «CHAR» соответствует CHAR (1). Если длина сохраняемого значения меньше n , в конце будет добавлено соответствующее число пробелов. Тип NATIONAL CHARACTER (n) хранит данные в кодировке UNICODE (2 байта на символ для поддержки национальных алфавитов). Допустима запись NCHAR. Строковые константы заключаются в одинарные или двойные кавычки.

CHARACTER VARYING (n) — строка переменной длины не более чем из n символов, допускается сокращение VARCHAR (n). Если длина сохраняемого значения m , где $m < n$, будет сохраняться ровно m символов. Аналогично предыдущему случаю, существует тип NATIONAL CHARACTER VARYING (n) с синонимом NVARCHAR (n).

CHARACTER LARGE OBJECT - символьный тип, позволяющий хранить большие объемы текста. Допустима запись CLOB. Также существует тип NCLOB. XML документ в формате XML (тип данных введен в версии стандарта SQL:2003). BIT (n) битовая строка длиной n бит. Запись «BIT» аналогична BIT (1). BIT VARYING (n) – битовая строка переменной длины не более n бит. BINARY LARGE OBJECT большой двоичный объект, допустима запись BLOB. DATE дата в формате “YYYY-MM-DD” TIME – тип данных для хранения отметок времени, включающих поля <часы><минуты>:<секунды>:<доли секунды>. TIMESTAMP тип данных Для совместного хранения даты и времени. Особенности реализации разных типов данных в MySQL представлены по ссылке http://www.mysql.ru/docs/man/Column_types.html

Создание базовых таблиц

Отметим две важные особенности таблиц SQL по сравнению с отношениями реляционной модели:

- в таблицах SQL допустимы идентичные строки, поэтому нет требования обязательного наличия потенциальных ключей;
- в таблицах SQL столбцы рассматриваются в порядке слева направо, тогда как в отношении порядок атрибутов неважен.

Базовые таблицы создаются с помощью оператора `CREATE TABLE`, формат которого:

```
CREATE TABLE <table name> (table-element commalist)
```

<table name> — имя создаваемой базовой таблицы;

table-element-commalist список через запятую определений столбцов или ограничений уровня таблицы. В списке элементов должно быть хотя бы одно определение столбца.

Определение столбца должно включать название столбца и указание на базовый тип или домен, на котором столбец определен. Также оно может включать указание значения по умолчанию и ограничения уровня столбца.

Ограничение `NOT NULL` требует, чтобы столбец не мог содержать значение `NULL` (все значения должны быть определены). По умолчанию или при явном указании `NULL` в определении столбца, неопределенные значения допускаются.

Создание базовых таблиц

Ключевое слово `PRIMARY KEY` позволяет указать первичный ключ, а `UNIQUE` — альтернативный. Оба эти ограничения требуют уникальности значений, но `PRIMARY KEY` дополнительно включает ограничение `NOT NULL`. Таким образом, если для столбца задано ограничение `UNIQUE`, в этом столбце может встречаться значение `NULL` не более одного раза.

Если указано ограничение `CHECK (<условие>)`, будет выполняться проверка условия на значение. Попытка создания строки рассматривается как нарушение проверочного условия, если в результате вычисления условного выражения получено значение «ложь».

Ограничение внешнего ключа, если его задавать как ограничение уровня столбца, описывается в соответствии с форматом:

```
REFERENCES <table name> [(column) ] [ON DELETE option] [ON UPDATE option]
```

где `<table name>` имя таблицы, на которую ссылается внешний ключ; в скобках может быть указан столбец, если он опущен, то объект ссылки — первичный ключ указанной таблицы; `option` определяет поведение при попытке удалить или изменить строку «родительской» таблицы, на которую ссылается внешний ключ. Может принимать значения `NO ACTION` (запретить изменение), `CASCADE` (каскадировать изменение или удаление), `SET DEFAULT` (установить значение по умолчанию), `SET NULL` (установить значение `NULL`).

Примеры создания таблиц

The screenshot shows the MySQL Workbench interface with the following components:

- Left Panel (MANAGEMENT, INSTANCE, PERFORMANCE, SCHEMAS):** The SCHEMAS section is expanded, showing the database 'class' and its table 'T1'. The table structure shows columns 'id' and 'sum'.
- Query Editor (Query 1):** Contains the following SQL statements:

```
1 use class;  
2 create table T1 (id INT primary key, sum real not null);
```
- Action Output:** A table showing the execution results of the queries.

| # | Time | Action | Message |
|---|----------|---|-------------------|
| 1 | 05:16:27 | use class | 0 row(s) affected |
| 2 | 05:17:41 | use class | 0 row(s) affected |
| 3 | 05:17:41 | create table T1 (id INT primary key, sum real not null) | 0 row(s) affected |

Query Completed

Примеры создания таблиц

The screenshot shows the MySQL Workbench interface with the following components:

- Top Bar:** MySQL Model, EER Diagram, localhost.
- Menu Bar:** File, Edit, View, Query, Database, Server, Tools, Scripting, Help.
- Left Panel (MANAGEMENT, INSTANCE, PERFORMANCE, SCHEMAS):**
 - MANAGEMENT:** Server Status, Client Connections, Users and Privileges, Status and System Variables, Data Export, Data Import/Restore.
 - INSTANCE:** Startup / Shutdown, Server Logs, Options File.
 - PERFORMANCE:** Dashboard, Performance Reports, Performance Schema Setup.
 - SCHEMAS:** Filter objects, T10, t1_id, Indexes, Foreign Keys, Triggers, T1.
- Query Editor (Query 1):**

```
1 use class;  
2 create table Students(id int primary key,  
3                       fio varchar(20) not null,  
4                       t1_id int references T1(id)  
5                       on delete no action  
6                       on update cascade);  
7  
8  
9
```
- Action Output:**

| | # | Time | Action | Message |
|---|----|----------|--|---|
| ✓ | 14 | 05:46:05 | use class | 0 row(s) affected |
| ✗ | 15 | 05:46:05 | create table Students(id int primary key, fio var... | Error Code: 1050. Table 'Students' already exists |
| ✓ | 16 | 05:46:34 | DROP TABLE `class`.`Students` | 0 row(s) affected |
| ✓ | 17 | 05:46:37 | use class | 0 row(s) affected |
| ✓ | 18 | 05:46:37 | create table Students(id int primary key, fio var... | 0 row(s) affected |
- Object Info:** Column: t1_id, Definition: t1_id int(11).

Ограничения уровня таблицы

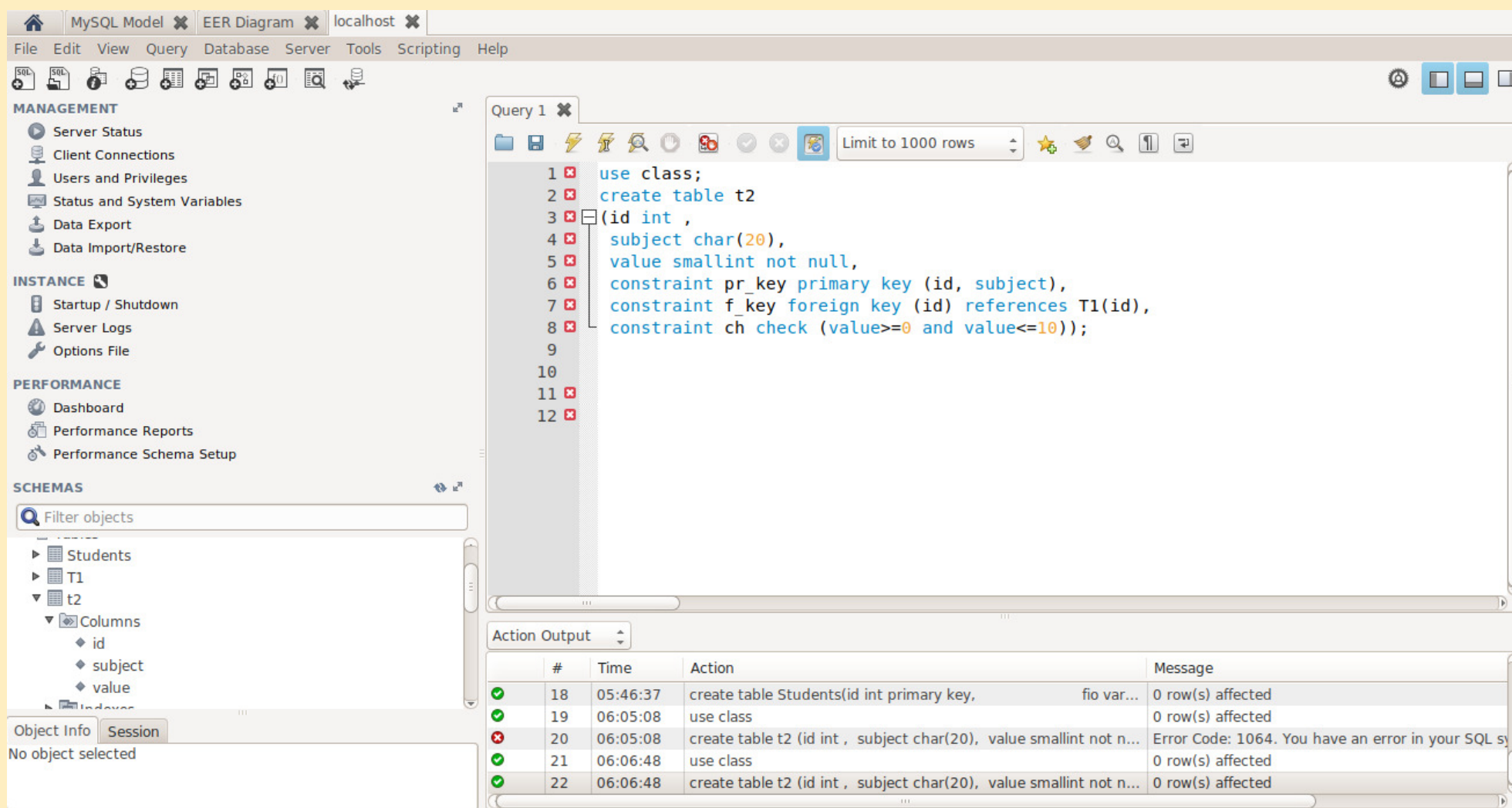
Рассмотрим теперь задание ограничений уровня таблицы. С их помощью можно, например, определить составной первичный или альтернативный ключ, что с помощью ограничения уровня столбца сделать не удастся (хотя на практике составные первичные ключи редко используются и их обычно заменяют суррогатными простыми).

Ограничения уровня таблицы описываются в операторе CREATE TABLE после описания столбцов. Используется следующий формат:

```
[CONSTRAINT <constraint name>] PRIMARY KEY | UNIQUE (column_commalist)
| FOREIGN KEY (column_commalist) references_definition | CHECK (conditional_expression)
```

Рассмотрим пример.

Примеры создания таблиц



The screenshot shows the MySQL Modeler interface. The main window displays a SQL query for creating a table named 't2' with columns 'id' and 'subject'. The query includes a primary key constraint for 'id', a foreign key constraint referencing 'T1(id)', and a check constraint for 'subject' values between 0 and 10. The 'Action Output' panel at the bottom shows the execution results of the query, including an error message for the check constraint.

```
1 use class;
2 create table t2
3 (id int ,
4  subject char(20),
5  value smallint not null,
6  constraint pr_key primary key (id, subject),
7  constraint f_key foreign key (id) references T1(id),
8  constraint ch check (value>=0 and value<=10));
9
10
11
12
```

| # | Time | Action | Message |
|----|----------|---|--|
| 18 | 05:46:37 | create table Students(id int primary key, fio var... | 0 row(s) affected |
| 19 | 06:05:08 | use class | 0 row(s) affected |
| 20 | 06:05:08 | create table t2 (id int , subject char(20), value smallint not n... | Error Code: 1064. You have an error in your SQL sy |
| 21 | 06:06:48 | use class | 0 row(s) affected |
| 22 | 06:06:48 | create table t2 (id int , subject char(20), value smallint not n... | 0 row(s) affected |

Явное указание имен ограничений может быть полезно, например, при их изменении. В то же время, если ключевое слово **CONSTRAINT** и название ограничения пропустить, СУБД автоматически сгенерирует имя ограничения, которое можно будет узнать с помощью инструментов администрирования.

Изменения базовых таблиц

Базовая таблица может быть изменена оператором ALTER TABLE. Могут быть сделаны следующие изменения: - добавление и удаление столбцов; определение для существующего столбца значения по умолчанию и удаление ранее определенного значения по умолчанию; создание нового ограничения целостности для таблицы и удаление ранее определенного.

Рассмотрим использование данной команды на примерах.

Добавим в таблицу T1 новый столбец.

The screenshot shows a SQL IDE interface. On the left, the 'SCHEMAS' panel shows a tree view with 'Students' and 'T1'. Under 'T1', the 'Columns' list includes 'id', 'sum', and 'col1'. The main query editor shows the following SQL code:

```
1 use class;  
2 alter table T1 add col1 varchar(100);
```

Below the query editor, the 'Action Output' panel displays a log of executed actions:

| # | Time | Action | Message |
|----|----------|---|--|
| 20 | 06:05:08 | create table t2 (id int , subject char(20), value smallint not n... | Error Code: 1064. You have an error in your SQL sy |
| 21 | 06:06:48 | use class | 0 row(s) affected |
| 22 | 06:06:48 | create table t2 (id int , subject char(20), value smallint not n... | 0 row(s) affected |
| 23 | 06:18:39 | use class | 0 row(s) affected |
| 24 | 06:18:39 | alter table T1 add col1 varchar(100) | 0 row(s) affected |

Изменения базовых таблиц

Базовая таблица может быть изменена оператором ALTER TABLE. Могут быть сделаны следующие изменения: - добавление и удаление столбцов; определение для существующего столбца значения по умолчанию и удаление ранее определенного значения по умолчанию; создание нового ограничения целостности для таблицы и удаление ранее определенного.

Рассмотрим использование данной команды на примерах.

Добавим в таблицу T1 новый столбец.

The screenshot shows a SQL IDE interface. On the left, the 'SCHEMAS' panel shows a tree view with 'Students' and 'T1'. Under 'T1', there are columns 'id', 'sum', and 'col1'. The main query editor shows two lines of SQL code:

```
1 use class;  
2 alter table T1 add col1 varchar(100);
```

Below the query editor, the 'Action Output' panel displays a log of executed actions:

| # | Time | Action | Message |
|----|----------|---|--|
| 20 | 06:05:08 | create table t2 (id int , subject char(20), value smallint not n... | Error Code: 1064. You have an error in your SQL sy |
| 21 | 06:06:48 | use class | 0 row(s) affected |
| 22 | 06:06:48 | create table t2 (id int , subject char(20), value smallint not n... | 0 row(s) affected |
| 23 | 06:18:39 | use class | 0 row(s) affected |
| 24 | 06:18:39 | alter table T1 add col1 varchar(100) | 0 row(s) affected |

Изменения базовых таблиц

А теперь удалим этот столбец:

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' panel shows the database structure for 'T1', including columns 'id' and 'sum'. The main query editor contains the following SQL code:

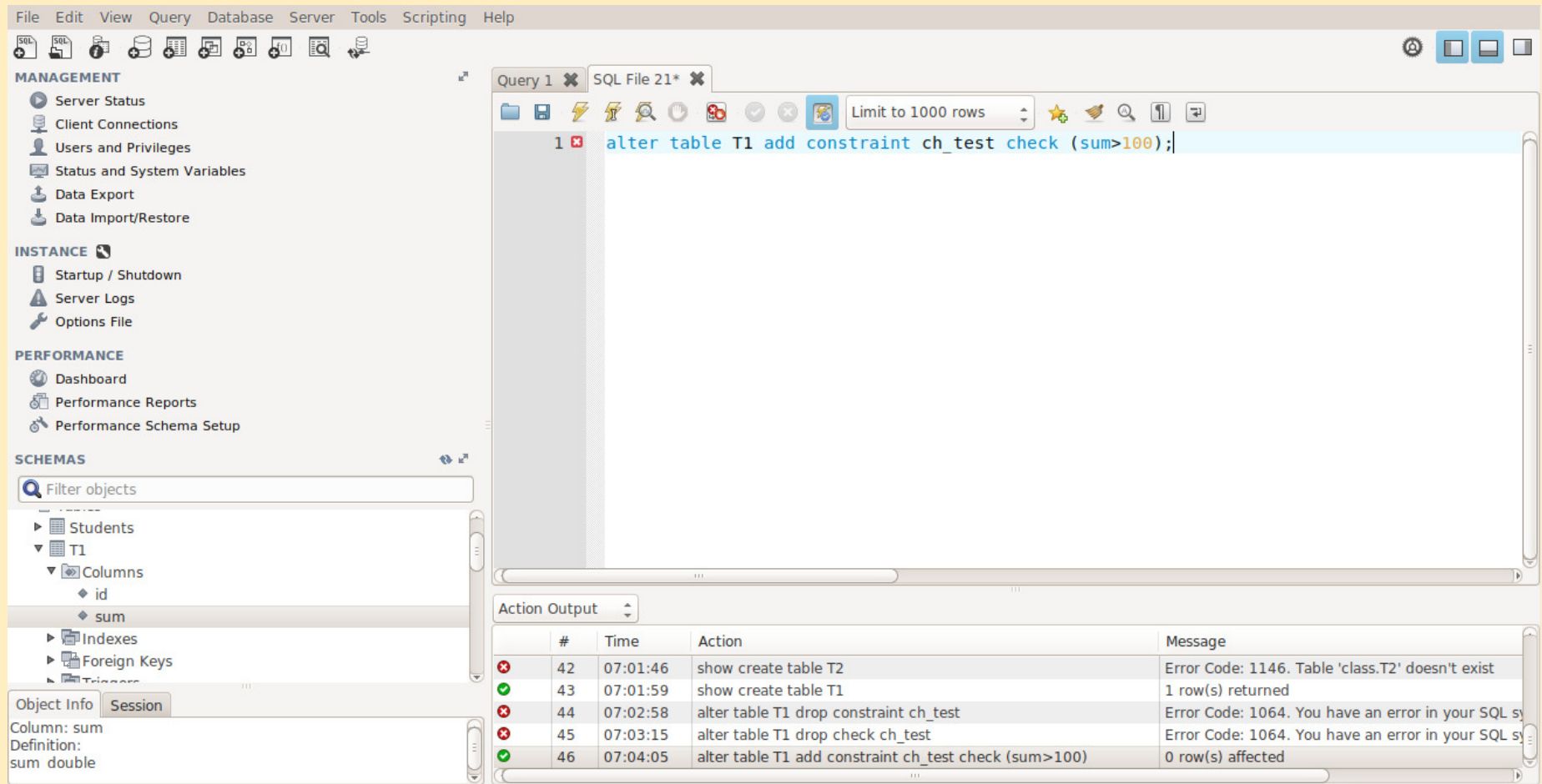
```
1 use class;  
2 alter table T1 drop column col1;
```

Below the query editor, the 'Action Output' panel displays a log of database actions. The log shows that the column 'col1' was successfully dropped from table 'T1'.

| # | Time | Action | Message |
|----|----------|---|-------------------|
| 22 | 06:06:48 | create table t2 (id int , subject char(20), value smallint not n... | 0 row(s) affected |
| 23 | 06:18:39 | use class | 0 row(s) affected |
| 24 | 06:18:39 | alter table T1 add col1 varchar(100) | 0 row(s) affected |
| 25 | 06:41:42 | use class | 0 row(s) affected |
| 26 | 06:41:42 | alter table T1 drop column col1 | 0 row(s) affected |

Изменения базовых таблиц

Добавим новое ограничение:



The screenshot displays a database management interface with a menu bar (File, Edit, View, Query, Database, Server, Tools, Scripting, Help) and a toolbar. The left sidebar contains sections for MANAGEMENT, INSTANCE, PERFORMANCE, and SCHEMAS. Under SCHEMAS, the 'T1' table is selected, showing its columns: 'id' and 'sum'. The main query editor shows a single query: `alter table T1 add constraint ch_test check (sum>100);`. The bottom pane, titled 'Action Output', displays a table of executed queries and their results.

| | # | Time | Action | Message |
|---|----|----------|---|--|
| ✗ | 42 | 07:01:46 | show create table T2 | Error Code: 1146. Table 'class.T2' doesn't exist |
| ✓ | 43 | 07:01:59 | show create table T1 | 1 row(s) returned |
| ✗ | 44 | 07:02:58 | alter table T1 drop constraint ch_test | Error Code: 1064. You have an error in your SQL sy |
| ✗ | 45 | 07:03:15 | alter table T1 drop check ch_test | Error Code: 1064. You have an error in your SQL sy |
| ✓ | 46 | 07:04:05 | alter table T1 add constraint ch_test check (sum>100) | 0 row(s) affected |

Изменения базовых таблиц

Удаление внешнего ключа:

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'T1' selected. The main editor window shows a SQL query: `alter table t2 drop FOREIGN KEY f_key;`. The 'Action Output' pane at the bottom shows a log of database actions:

| # | Time | Action | Message |
|----|----------|---|--|
| 45 | 07:03:15 | alter table T1 drop check ch_test | Error Code: 1064. You have an error in your SQL sy |
| 46 | 07:04:05 | alter table T1 add constraint ch_test check (sum>100) | 0 row(s) affected |
| 47 | 07:07:20 | alter table t2 drop constraint f_key | Error Code: 1064. You have an error in your SQL sy |
| 48 | 07:11:13 | alter table t2 drop FOREIGN KEY id | Error Code: 1091. Can't DROP 'id'; check that colu |
| 49 | 07:11:44 | alter table t2 drop FOREIGN KEY f_key | 0 row(s) affected |

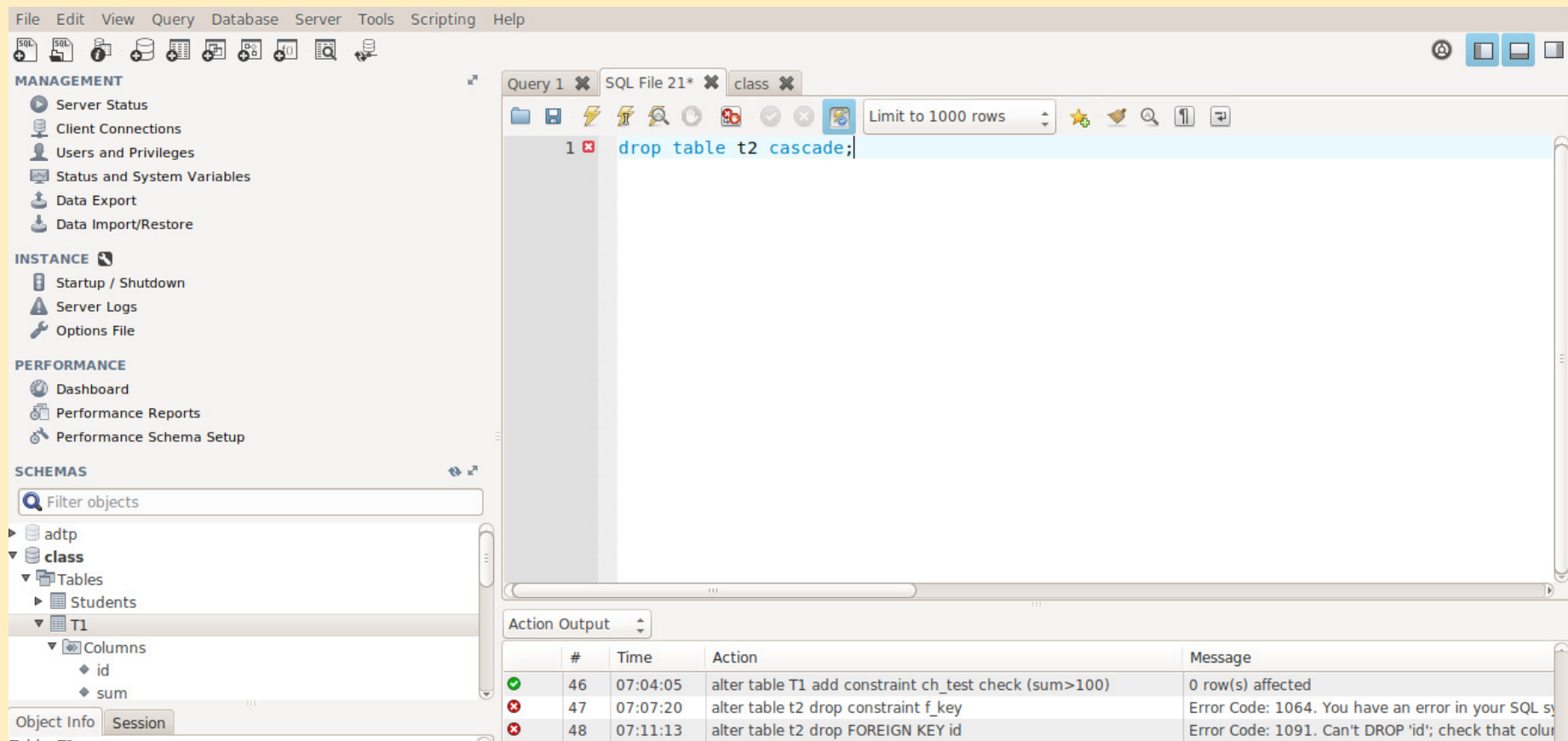
Удаление базовой таблицы

Базовая таблица может быть удалена оператором DROP TABLE.

DROP TABLE <table name> (RESTRICT | CASCADE)

где <table name> название существующей базовой таблицы; опция RESTRICT не позволит удалить таблицу, если она используется при определении какого-либо представления или ограничения целостности;

при использовании опции CASCADE оператор выполняется в любом случае, определения представлений и ограничений целостности, включающие указания на данную таблицу, также будут удалены.



Ряд полезных команд

SHOW DATABASES; - список баз данных

SHOW TABLES [FROM db_name]; - список таблиц в базе

SHOW COLUMNS FROM таблица [FROM db_name]; - список столбцов в таблице

SHOW CREATE TABLE table_name; - показать структуру таблицы в формате "CREATE TABLE"

SHOW INDEX FROM tbl_name; - список индексов

SHOW GRANTS FOR user [FROM db_name]; - привилегии для пользователя.

SHOW VARIABLES; - значения системных переменных

SHOW [FULL] PROCESSLIST; - статистика по mysqld процессам

SHOW STATUS; - общая статистика

SHOW TABLE STATUS [FROM db_name]; - статистика по всем таблицам в базе

Операции добавления, обновления и удаления данных

Язык обработки данных (DML) включает три операции обновления: INSERT (вставка), UPDATE (изменение) и DELETE (удаление). Рассмотрим синтаксис этих операторов.

Вставка строк производится с помощью оператора INSERT, формат которого приведен ниже.

```
INSERT INTO <table name> [(column_list)] {VALUES (value_list) | <query>}
```

где <table name> — имя базовой таблицы или обновляемого представления, column list необязательный параметр, позволяющий указать обновляемые столбцы; если он не указан, то порядок столбцов берется таким же, как в определении таблицы.

Рассмотрим ряд примеров:

Добавление данных в таблицу

The screenshot shows the MySQL Workbench interface with the following components:

- MANAGEMENT**: Server Status, Client Connections, Users and Privileges, Status and System Variables, Data Export, Data Import/Restore.
- INSTANCE**: Startup / Shutdown, Server Logs, Options File.
- PERFORMANCE**: Dashboard, Performance Reports, Performance Schema Setup.
- SCHEMAS**: Filter objects, adtp, class, Tables, Students, T1, Columns (id, sum).
- Object Info**: Table: T1, Columns: id int(11) PK, sum double.
- Query 1**:

```
1 • insert into T1 value (1,10.5);
2 • insert into T1 (id,sum) value (2,132);
```
- Result Grid**:

| # | id | sum |
|---|------|------|
| 1 | 1 | 10.5 |
| 2 | 2 | 132 |
| * | NULL | NULL |
- Action Output**:

| # | Time | Action | Message |
|----|----------|---------------------------------------|--|
| 53 | 07:27:53 | insert into T1 (id) value (2) | Error Code: 1364. Field 'sum' doesn't have a default value |
| 54 | 07:28:00 | SELECT * FROM class.T1 LIMIT 0, 1000 | 1 row(s) returned |
| 55 | 07:28:22 | SELECT * FROM class.T1 LIMIT 0, 1000 | 1 row(s) returned |
| 56 | 07:28:45 | insert into T1 (id,sum) value (2,132) | 1 row(s) affected |
| 57 | 07:28:50 | SELECT * FROM class.T1 LIMIT 0, 1000 | 2 row(s) returned |

Добавление данных в таблицу

При добавлении записи, значения отдельных атрибутов могут быть опущены. В этом случае, столбец получит или значение по умолчанию, если оно для него было определено, или значение NULL, Если такой столбец находится в середине списка, пропуск значения явно указывается, если в конце - можно ничего не указывать.

С помощью оператора INSERT также можно добавить в таблицу набор строк, полученных в результате выполнения запроса на выборку. В этом случае, вместо ключевого слова VALUES и перечисления значений, должен стоять оператор SELECT

The screenshot shows the MySQL Model interface. The left sidebar contains sections for MANAGEMENT, INSTANCE, PERFORMANCE, and SCHEMAS. The SCHEMAS section shows a tree view with 'Students' and 'T1'. The 'T1' table is selected, showing its columns: 'id' (int(11) PK) and 'sum' (double). The main window displays a query in the 'Query 1' tab:

```
1 insert into T1 (id, sum)
2 select `Табельный № учителя`, `Классная комната` from `Классы`;
```

The 'Result Grid' shows the results of the query:

| # | id | sum |
|---|------|------|
| 1 | 1 | 10.5 |
| 2 | 2 | 132 |
| 3 | 100 | 212 |
| 4 | 101 | 203 |
| * | NULL | NULL |

The 'Action Output' section shows the execution log:

| # | Time | Action | Message |
|----|----------|--|--|
| 60 | 07:38:32 | insert into T2 select `Табельный № учителя`, `Классная ком... | Error Code: 1146. Table 'class.T2' doesn't exist |
| 61 | 07:38:39 | select `Табельный № учителя`, `Классная комната` from `К... | 2 row(s) returned |
| 62 | 07:39:31 | insert into T2 (id, sum) select `Табельный № учителя`, `Кла... | Error Code: 1146. Table 'class.T2' doesn't exist |
| 63 | 07:39:41 | insert into T1 (id, sum) select `Табельный № учителя`, `Кла... | 2 row(s) affected |
| 64 | 07:39:51 | SELECT * FROM class.T1 LIMIT 0, 1000 | 4 row(s) returned |

Изменение данных в таблице

Значения полей существующих записей таблицы можно изменить с помощью оператора UPDATE, формат которого следующий:

```
UPDATE <table name> SET column_1 = value_1 [, column_2 = value_2 ...] [ WHERE <predicate>]
```

Здесь <table name> название обновляемой таблицы; column_1 название первого из обновляемых столбцов, value_1 присваиваемое ему значение (константа или результат вычислений). Обновляемых столбцов может быть несколько.

Выражение <predicate> обозначает логическое условие. Если необязательная секция WHERE пропущена, обновляются все записи таблицы. Если эта секция присутствует, то обновляется только те записи, для которых <predicate> будет истинным. Составное условие формируется с помощью логических связок AND (логическое «и»), OR (логическое «или»), NOT (отрицание). Если необходимо обновить только одну конкретную запись, это можно сделать, указав в условии значение первичного ключа.

Изменение данных в таблице

The screenshot shows the MySQL Workbench interface with the following components:

- Top Bar:** MySQL Model, EER Diagram, localhost.
- Menu Bar:** File, Edit, View, Query, Database, Server, Tools, Scripting, Help.
- Left Sidebar:**
 - MANAGEMENT:** Server Status, Client Connections, Users and Privileges, Status and System Variables, Data Export, Data Import/Restore.
 - INSTANCE:** Startup / Shutdown, Server Logs, Options File.
 - PERFORMANCE:** Dashboard, Performance Reports, Performance Schema Setup.
 - SCHEMAS:** Filter objects, Students, T1 (expanded showing Columns: id, sum; Indexes; Foreign Keys; Triggers).
- Query Editor:** Query 1, T1, Students, Классы, T1, T1. The query is: `update T1 set sum=sum*1.1 where id>10;`. A "Limit to 1000 rows" dropdown is visible.
- Result Grid:** Shows the result of the query. The table has columns #, id, and sum.

| # | id | sum |
|---|------|---------------------|
| 1 | 1 | 10.5 |
| 2 | 2 | 132 |
| 3 | 100 | 233.200000000000002 |
| 4 | 101 | 223.3 |
| * | NULL | NULL |
- Action Output:** Shows the execution log.

| # | Time | Action | Message |
|----|----------|--|--|
| 62 | 07:39:31 | insert into T2 (id, sum) select `Табельный № учителя`, `Кла... | Error Code: 1146. Table 'class.T2' doesn't exist |
| 63 | 07:39:41 | insert into T1 (id, sum) select `Табельный № учителя`, `Кла... | 2 row(s) affected |
| 64 | 07:39:51 | SELECT * FROM class.T1 LIMIT 0, 1000 | 4 row(s) returned |
| 65 | 07:47:14 | update T1 set sum=sum*1.1 where id>10 | 2 row(s) affected |
| 66 | 07:47:21 | SELECT * FROM class.T1 LIMIT 0, 1000 | 4 row(s) returned |

Удаление данных из таблицы

Удаление строк из таблицы производится оператором DELETE.

DELETE FROM <table name> [WHERE <predicate>]

Здесь <table name> - название таблицы, из которой удаляются данные, <predicate> логическое условие. Если секция WHERE присутствует, удаляются только те записи, для которых <predicate> будет истинным.

Выполнение приведенного ниже выражения, удалит все данные из таблицы, но в отличие от оператора DROP TABLE, сама таблица удалена не будет.

DELETE FROM Students

Надо учитывать, что определенные для базы данных ограничения целостности (например, ограничение внешнего ключа) могут не позволить обновить или удалить значения некоторых записей.

The screenshot shows a database management interface with a menu bar (File, Edit, View, Query, Database, Server, Tools, Scripting, Help) and a toolbar. The left sidebar contains sections for MANAGEMENT, INSTANCE, PERFORMANCE, and SCHEMAS. The SCHEMAS section shows a tree view with 'Students' and 'T1' (selected). The main window displays a query editor with the text: `1 • delete from T1 where id=1;`. Below the query editor is a 'Result Grid' showing the following data:

| # | id | sum |
|---|------|--------------------|
| 1 | 2 | 132 |
| 2 | 100 | 233.20000000000002 |
| 3 | 101 | 223.3 |
| * | NULL | NULL |

Below the result grid is an 'Action Output' table showing the execution of the query:

| # | Time | Action | Message |
|----|----------|---------------------------------------|-------------------|
| 64 | 07:39:51 | SELECT * FROM class.T1 LIMIT 0, 1000 | 4 row(s) returned |
| 65 | 07:47:14 | update T1 set sum=sum*1.1 where id>10 | 2 row(s) affected |
| 66 | 07:47:21 | SELECT * FROM class.T1 LIMIT 0, 1000 | 4 row(s) returned |

Управление правами доступа к объектам базы данных

В MySQL пользователь характеризуется двумя параметрами: именем и хостом, с которого он может обращаться. По умолчанию доступ разрешен только с локальной машины, т.е. для пользователя `user@localhost`. Права на доступ пользователям даются с помощью команды `GRANT`. Команда выполняется под рутом. Например, если я хочу создать юзера, который сможет коннектиться с любого хоста с полными правами, то следует выполнить следующую команду:

```
GRANT ALL PRIVILEGES ON 'имя_базы'.* TO myuser@% IDENTIFIED BY 'пароль';
```

Примечание. Обратите внимание, что данная команда дает доступ пользователю `myuser` со всех IP кроме `127.0.0.1`, соответствующего `localhost`.

Для пользователя `myuser@localhost` необходимо давать права отдельной командой `GRANT`.

Второй пример показывает как дать право читать таблицу `time_zone` в базе `mysql` пользователю `myuser` с машины `192.168.0.76` с паролем `mypassy`:

```
GRANT SELECT ON mysql.time_zone TO myuser@192.168.0.76 IDENTIFIED BY 'mypassy';
```

Управление правами доступа к объектам базы данных

Создание нового пользователя

```
CREATE USER 'user'@'localhost' IDENTIFIED BY 'secret';
```

Добавим выбранные привилегии для всех таблиц БД dbname пользователю 'user'@'localhost'

```
GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,DROP,ALTER,INDEX ON dbname.*  
TO 'user'@'localhost';
```

Добавить все привилегии для всех таблиц БД dbname пользователю 'user'@'localhost':

```
GRANT ALL PRIVILEGES ON dbname.* TO 'user'@'localhost';
```

Удаление прав пользователя 'user'@'localhost' для БД dbname:

```
REVOKE ALL ON dbname.* FROM 'user'@'localhost';
```

Перезагрузка привилегий:

```
FLUSH PRIVILEGES;
```

Посмотреть список пользователей:

```
SELECT User,Host FROM mysql.user;
```

Посмотреть список привилегий пользователя:

```
SHOW GRANTS FOR 'user'@'localhost';
```

Удалить все привилегии пользователя:

```
REVOKE ALL PRIVILEGES, GRANT OPTION FROM 'user'@'localhost';
```

Удалить пользователя:

```
DROP USER 'user'@'localhost';
```

Понятие представления

Представление (VIEW) — объект базы данных, являющийся результатом выполнения запроса к базе данных, определенного с помощью оператора SELECT, в момент обращения к представлению. Представления иногда называют «виртуальными таблицами».

Для создания представления можно использовать следующую команду:

```
CREATE VIEW view_name (список полей) AS select_statement;
```