Администрирование баз данных

Типы данных и таблицы

- Типы данных
- Создание/Изменение/Удаление таблиц

Создание таблицы

- Как выбрать "правильные" типы данных?
- Нужны ли все колонки?
- Какая самая короткая версия данных?

Типы таблиц

- MyISAM
- InnoDB
- MERGE
- MEMORY (HEAP)
- CSV
- REDERATED

MyISAM

Каждая **MyISAM** таблица хранится на диске в двух файлах:
.MYD — в этом файле содержатся данные таблицы;
.MYI — в этом файле содержатся индексы таблицы.

Таблицы **MyISAM** обладают рядом особенностей:

1. Данные хранятся в кроссплатформенном формате.

2. Максимальное количество индексов — 64, каждый индекс может быть максимум из 16 столбцов.

3. С версии MySQL 4.1, для каждого текстового столбца может быть задана своя кодировка.

4. Допускается индексирование текстовых столбцов, в том числе и переменной длины.

5. Поддерживается полнотекстовый поиск.

6. Таблицы имеют специальный флаг, указывающий на правильность закрытия таблицы.

InnoDB

Тип таблиц **InnoDB** разработан компанией Innobase. Таблицы такого типа предоставляют высокую производительность и устойчивое хранение данных в таблицах объёмом до 1 Тбайт и нагрузкой на сервер до 800 вставок/обновлений в секунду.

Особенности типа **InnoDB**:

1. Все таблицы хранятся в едином табличном пространстве, поэтому имена таблиц должны быть уникальны.

2. Хранение данных в едином табличном пространстве позволяет снять ограничение на объём таблиц. Файл с таблицами может быть разбит на несколько частей и распределён по нескольким дискам или даже хостам.

3. Таблицы поддерживают автоматическое восстановление после сбоя.

4. Поддерживаются транзакции.

5. Этот тип таблиц в MySQL единственный, который поддерживает каскадное удаление и внешние ключи.

6. Выполняется блокировка на уровне отдельных записей.

MERGE

Позволяет объединять несколько таблиц типа **MyISAM** в одну.
Таблицы **MyISAM** объединяемые в одну таблицу **MERGE**, должны иметь идентичную структуру, т.е. одинаковые столбцы, индексы и порядок их следования.

К объединенной таблице можно применять команды SELECT, DELETE и UPDATE. Если попытаться удалить таблицу **MERGE** при помощи команды DROP TABLE, то будет уничтожена именно **MERGE** таблица, исходные таблицы MyISAM не будут затронуты.

```
create table summ_table (
id int not null auto_increment,
name char(20), index(id))
engine=merge union=(table_one,table_two) insert_method=last;
```

Параметры INSERT_METHOD
FIRST — При вставке новой записи в таблицу **MERGE**, запись размещается в первой таблице из списка в параметре UNION.
LAST — Запись размещается в последней таблице списка.
NO — Вставка в **MERGE** таблицу невозможна, а использования оператора INSERT приведёт к ошибке.

MEMORY

MEMORY (HEAP) хранится в оперативной памяти, из-за чего все запросы к таким таблицам выполняются очень быстро.

Недостаток: полная потеря данных в случае сбоя работы сервера. В связи с этим в таких таблицах хранят в основном временные данные, которые можно легко восстановить заново.

При создании таблицы типа **MEMORY**, создаётся один файл с расширением frm, в котором определяется структура таблицы.

При остановке или перезагрузке сервера, данные о структуре таблицы остаются, но вся информация содержащаяся в этой таблице теряется, поскольку хранится только в оперативной памяти.

При каждой перезагрузке сервера, пересоздавать таблицу не нужно, её структура остаётся.

CSV

Формат CSV, представляет собой обычный текстовый файл, записи в котором хранятся в строках, а поля разделены точкой с запятой. При создании таблицы формируются два файла. Один с расширением frm, содержащим структуру таблицы. Второй с расширением CSV — содержащим данные в CSV — формате.

FEDERATED

Тип **FEDERATED** позволяет хранить данные в удалённых таблицах, которые находятся на других машинах в сети. При создании таблицы создаётся только файл структуры с расширением frm, поскольку данные хранятся на удалённой машине.

Для создания таблицы **FEDERATED** необходимо сначала создать таблицу на удалённой машине, затем на рабочей машине с указанием на удалённую.

Пример создания таблицы **FEDERATED**:

```
    //Сначала создаём таблицу на удалённой машине...
    create table test1( field1 INT) engine=MyISAM;

    //создаём таблицу FEDERATED на рабочей машине...
    create table test2( field1 INT) engine=FEDERATED
CONNECTION='mysql:// user_name[:password]@host_name[:port_num]/db_name/tbl_name ';
```

Структура таблиц на машинах должна быть идентична.

В поле CONNECTION таблицы FEDERATED указывается строка подключения к таблице на удалённой машине.

В таблицах данного типа не поддерживаются транзакции.

Создание таблицы

Максимальная длина имен таблиц и столбцов составляет **64** знака и может включать литеры, цифры и символы ‘_’ и ‘$’. Имя может начинаться с цифры, но не должно полностью состоять из цифр.

Для создания таблицы используется оператор **CREATE TABLE**

Общий синтаксис:

```
CREATE [TEMPORARY] TABLE [IF NOT EXISTS] table_name
(спецификация, ...)
[опция, ...]
```

Формат спецификации:

```
column_name data_type[(length)] [NOT NULL | NULL][DEFAULT значение][AUTO_INCREMENT][KEY]
```

```
mysql>create table if not exists table_test_1 (
-> id int(5) auto_increment primary key,
-> name varchar(30) not null,
-> status varchar(10) not null default 'active’,
-> reg_date date
-> ) ENGINE=INNODB;

Query OK, 0 rows affected (0.35 sec)
```

Описание таблицы

```
mysql>describe table_test_1;
+----------+-------------+------+-----+---------+----------------+
| Field    | Type        | Null | Key | Default | Extra          |
+----------+-------------+------+-----+---------+----------------+
| id       | int(5)      | NO   | PRI | NULL    | auto_increment |
| name     | varchar(30) | NO   |     | NULL    |                |
| status   | varchar(10) | NO   |     | active  |                |
| reg_date | date        | YES  |     | NULL    |                |
+----------+-------------+------+-----+---------+----------------+
4 rows in set (0.01 sec)
```

Проверка наличия созданной таблицы в базе

```
show tables;
```

Просмотр описания таблицы

```
describe table_test_1;
```

удаление таблицы

```
drop table_test_1;
```

Типы данных

MySQL поддерживает несколько типов столбцов, которые можно разделить на три категории:

- числовые типы данных,
- типы данных для хранения даты и времени
- символьные (строковые) типы данных.

Числовые типы данных

Тип	Байт	От	До
TINYINT [(M)]	1	-128	127
SMALLINT [(M)]	2	-32768	32767
MEDIUMINT [(M)]	3	-8388608	8388607
INT [(M)]	4	-2147483648	2147483647
BIGINT[(M)]	8	-9223372036854775808	9223372036854775807
BOOLEAN[(M)]	1	0 или 1	
DECIMAL[(M, [D])]	M+2байта	Повышенная точность. Зависит от параметров M и D	
FLOAT[(M,D)]	4 байта	-1,175494351E-38	3,402823466E+38
DOUBLE[(M,D)]	8 байт	-1,7976931348623157E+308	1,7976931348623157E+308

Целочисленные типы данных

Тип	Байт	От	До
TINYINT [(M)]	1	-128	127
SMALLINT [(M)]	2	-32768	32767
MEDIUMINT [(M)]	3	-8388608	8388607
INT [(M)]	4	-2147483648	2147483647
BIGINT[(M)]	8	-9223372036854775808	9223372036854775807
BOOLEAN[(M)] (синоним TINYINT(1))	1	Значения 1 или 0 соответствуют true или false	

```
create table numeric_format (
   num1 TINYINT(3),
   num2 TINYINT(3) UNSIGNED,
   num3 INT(5) UNSIGNED ZEROFILL
);
```

Целые типы данных могут быть объявлены положительными. При объявлении поля использовать ключевое слово `UNSIGNED`

При объявлении целого типа задается количество отводимых под число символов M (от 1 до 255).

Если дополнительно указан необязательный атрибут **ZEROFILL**, свободные позиции по умолчанию заполняются нулями слева

Целочисленные типы

```
mysql> create table numeric_format (num1 TINYINT(3), num2 TINYINT(3) UNSIGNED, num3 INT(5) UNSIGNED ZEROFILL);
Query OK, 0 rows affected (0.36 sec)

mysql> insert into numeric_format values (127, 200, 3);
Query OK, 1 row affected (0.07 sec)

mysql> insert into numeric_format values (-127, 200, 3);
Query OK, 1 row affected (0.07 sec)

mysql> insert into numeric_format values (-127, -200, 3);
ERROR 1264 (22003): Out of range value for column 'num2' at row 1

mysql> insert into numeric_format values (200, -200, 3);
ERROR 1264 (22003): Out of range value for column 'num1' at row 1

mysql> select * from test1;
+------+------+-------+
| num1 | num2 | num3  |
+------+------+-------+
|  127 |  200 | 00003 |
| -127 |  200 | 00003 |
+------+------+-------+
2 rows in set (0.00 sec)
```

Счетчик auto_increment

```
create table autoincrement
    (num1 int(5) zerofill not null auto_increment primary key,
     num2 TINYINT(3));
```

```
create table autoincrement1
    (num1 int(5) zerofill not null auto_increment,
         num2 TINYINT(3),
         primary key(num1));
```

```
insert into autoincrement values (null, 15);
```

```
insert into autoincrement (num2) values (25);
```

```
select * from autoincrement;
```

```
insert into autoincrement (num1, num2) values (7777, 50);
```

Вещественные числовые типы

Тип	Байт	От	До
DECIMAL[(M, [D])]	M+2байта	Повышенная точность. Зависит от параметров M и D	
FLOAT[(M,D)]	4 байта	-1,175494351E-38	3,402823466E+38
DOUBLE[(M,D)]	8 байт	-1,7976931348623157E+308	1,7976931348623157E+308

```
create table decimal_float (
dec_num decimal(5,2),
float_num float(5,2)
);

insert into decimal_float values (5.23, 5.23);

select * from decimal_float;
```

DECIMAL – предназначен для повышенной точности. Требуемая точность задается при объявлении столбца.

```
salary decimal(6,2)
```

6 – общее количество символов для числа
2 – количество знаков после точки

Первый параметр может иметь максимальное значение 64, второй – 30.

FLOAT, DOUBLE – является приблизительным (хранит неточное число).

```
select * from decimal_float;
+---------+-----------+
| dec_num | float_num |
+---------+-----------+
|    5.23 |      5.23 |
+---------+-----------+
1 row in set (0.00 sec)
```

```
select dec_num*1000000, float_num*1000000 from decimal_float;
```

```
select dec_num*1000000, float_num*1000000 from decimal_float;
+-----------------+-------------------+
| dec_num*1000000 | float_num*1000000 |
+-----------------+-------------------+
|      5230000.00 |        5230000.02 |
+-----------------+-------------------+
1 row in set (0.00 sec)
```

Строковые типы данных (CHAR, VARCHAR, BLOB, TEXT, ENUM, SET**)**

CHAR, VARCHAR

CHAR(M) - длина поля постоянна и задается при создании таблицы. M от 1 до 65535. Величины типа CHAR при хранении дополняются справа пробелами до заданной длины. Эти концевые пробелы удаляются при извлечении хранимых величин.

VARCHAR(M) - строки переменной длины. M от 1 до 65535. При данных используется только то количество символов, которое необходимо, плюс один байт для записи длины. Хранимые величины пробелами не дополняются, наоборот, концевые пробелы при хранении удаляются.

Величина	CHAR(4)	Требуемая память	VARCHAR(4)	Требуемая память
''	' '	4 байта	''	1 байт
'ab'	'ab '	4 байта	'ab'	3 байта
'abcd'	'abcd'	4 байта	'abcd'	5 байтов
'abcdefgh'	'abcd'	4 байта	'abcd'	5 байтов

Если задаваемая в столбце CHAR или VARCHAR величина превосходит максимально допустимую длину столбца, то эта величина соответствующим образом усекается.

BLOB, TEXT

Тип столбца	Требуемая память
TINYBLOB, TINYTEXT	L+1 байт, где L < 2^8
BLOB, TEXT	L+2 байт, где L < 2^16
MEDIUMBLOB, MEDIUMTEXT	L+3 байт, где L < 2^24
LONGBLOB, LONGTEXT	L+4 байт, где L < 2^32

BLOB - двоичный объект большого размера, который может содержать переменное количество данных.

Тип TEXT обычно используется для больших объемов текста, в то время как BLOB – для двоичных объектов, таких как, электронные документы, изображения, звуки и т.д.

SET, ENUM

Строки этих типов принимают значения из заранее заданного списка допустимых значений. Основное различие между ними заключается в том, что значение типа ENUM должно содержать одно значение из указанного множества, тогда как столбцы SET могут содержать любой или все элементы заранее заданного множества одновременно.
SET максимум 64 значения.
ENUM максимум 65535 значений.

```
CREATE TABLE users(
...
        region ENUM('EU', 'USA', 'ASIA') NOT NULL
...
);
```

Преимущество - записывается номер значения вместо самого значения в каждую строку. Этим обеспечивается огромная экономия места.

Типы данных даты и времени (*DATETIME, DATE, TIMESTAMP, TIME и YEAR)*

DATETIME

Используется для величин, содержащих информацию как о дате, так и о времени.

Формат: 'YYYY-MM-DD HH:MM:SS‘

Диапазон величин от '1000-01-01 00:00:00' до '9999-12-31 23:59:59'

DATE

Используется для величин с информацией только о дате, без части, содержащей время.

Формат: 'YYYY-MM-DD'.

Диапазон величин от '1000-01-01' до '9999-12-31'.

TIMESTAMP

Обеспечивает тип представления данных, который можно использовать для автоматической записи текущих даты и времени при выполнении операций INSERT или UPDATE. При наличии нескольких столбцов типа TIMESTAMP только первый из них обновляется автоматически.
Величины типа TIMESTAMP могут принимать значения от начала 1970 года до некоторого значения в 2037 году с разрешением в одну секунду.

```
mysql>CREATE TABLE timestamp_test (
name varchar(30),
puttime1 TIMESTAMP NULL DEFAULT CURRENT_TIMESTAMP,
puttime2 TIMESTAMP NULL ON UPDATE CURRENT_TIMESTAMP
);
```

```
mysql> describe timestamp_test;
+----------+-------------+------+-----+-------------------+-----------------------------+
| Field    | Type        | Null | Key | Default           | Extra                       |
+----------+-------------+------+-----+-------------------+-----------------------------+
| name     | varchar(30) | YES  |     | NULL              |                             |
| puttime1 | timestamp   | YES  |     | CURRENT_TIMESTAMP |                             |
| puttime2 | timestamp   | YES  |     | NULL              | on update CURRENT_TIMESTAMP |
+----------+-------------+------+-----+-------------------+-----------------------------+
3 rows in set (0.00 sec)
```

```
mysql> insert into timestamp_test (name) values ('John');
Query OK, 1 row affected (0.05 sec)

mysql> select * from timestamp_test;
+------+---------------------+----------+
| name | puttime1            | puttime2 |
+------+---------------------+----------+
| John | 2018-10-11 23:58:09 | NULL     |
+------+---------------------+----------+
1 row in set (0.00 sec)
```

```
mysql> update timestamp_test set name='Alex';
Query OK, 1 row affected (0.07 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from timestamp_test;
+------+---------------------+---------------------+
| name | puttime1            | puttime2            |
+------+---------------------+---------------------+
| Alex | 2018-10-11 23:58:09 | 2018-10-11 23:59:14 |
+------+---------------------+---------------------+
1 row in set (0.00 sec)
```

TIME

Формат: 'HH:MM:SS' (или в формате 'HHH:MM:SS' для больших значений часов).

Величины TIME могут изменяться в пределах от '-838:59:59' до '838:59:59'.

YEAR

Однобайтный тип данных для представления значений года.

Диапазон возможных значений - от 1901 до 2155

Тип данных NULL

Используется когда информации недостаточно и для части данных нельзя определить, какое значение они примут

```
CREATE TABLE users
(
  id_user INT NOT NULL,
  name VARCHAR(250) NOT NULL,
  email VARCHAR(200) NULL
)
```

Передача значения NULL первым двум столбцам таблицы приведет к возникновению ошибки. Если при создании новой записи поля не инициируются и им будут присвоены значения по умолчанию, то id_user получит значение 0, name – пустую строку