Изучение SQL с помощью phpMyAdmin

Автор: Марк Делисл
Перевод: Иван Шумилов

Данная обучающая статья от ведущего разработчика phpMyAdmin предназначена для веб-мастеров, которые только начинают знакомиться с основами SQL. В качестве инструментария используются сервер баз данных MySQL и интерфейс их администрирования - phpMyAdmin. Последний и будет использоваться в данной статье в качестве инструмента для изучения SQL.

Что такое SQL?

SQL (англ. Structured Query Language - язык структурированных запросов) - универсальный язык, применяемый для создания, модификации и управления данными в реляционных базах данных. Этот язык был разработан IBM (экспериментальная СУБД "System R") в 1974-1979 годах. Первый стандарт языка SQL был принят Американским национальным институтом стандартизации (ANSI) в 1987 (так называемый SQL level 1) и несколько уточнён в 1989 году (SQL level 2), 1992 и 1999 году. После этого, индустрия начала широко использовать SQL как язык реляционных баз данных, и на сегодняшний день практически каждая база данных основана на SQL.

Архитектура данных, к которой обращается SQL называется реляционной. В реляционных базах данных все данные представлены в виде простых таблиц, разбитых на строки и столбцы, на пересечении которых расположены данные. Запросы к таким таблицам возвращают таблицы, которые сами могут становиться предметом дальнейших запросов. Каждая база данных может включать несколько таблиц, которые, как правило, связаны друг с другом, откуда и произошло название реляционные.

В данной статье мы будем использовать MySQL, популярную открытую реализацию SQL, которая предоставляется абсолютным большинством хостинг-провайдеров.

Инструментарий для данной статьи

Для выполнения упражнений, приведенных в данной статье, потребуется доступ к MySQL-серверу. В качестве интерфейса для MySQL используется phpMyAdmin - PHP приложение, запущенное на Web-сервере. Для полного ознакомления с возможностями phpMyAdmin рекомендуется прочитать книгу "Mastering phpMyAdmin for effective MySQL Management".

Эта статья познакомит Вас с основами синтаксиса SQL, время от времени, предлагая Вам ввести выражения; покажет, как phpMyAdmin генерирует MySQL-запросы на основе ваших действий через интерфейс.

Создание таблиц в phpMyAdmin

В качестве примера, мы будем использовать географическую информационную систему. Допустим, мы решили, что нам необходима информация о городах и странах - таким образом, нам понадобятся две таблицы, которые будут частью базы данных `geodb`. Для создания таблиц можно использовать вкладку "Structure" на странице просмотра базы данных, или использовать блок SQL-запросов для ввода соответствующего выражения.

Чтобы создать таблицу, воспользуемся выражением CREATE TABLE, в котором мы зададим имя нашей новой таблицы. Выражение начинается с CREATE TABLE, после которой следует имя таблицы. Затем в скобках указывается список столбцов, и информация о ключах. Каждому столбцу дается имя, указывается тип данных, указывается атрибут NULL или NOT NULL (здесь, NOT NULL означает, что колонка не может иметь значение NULL), и значение по умолчанию, если оно уместно.

CREATE TABLE cities (id int(11) NOT NULL auto_increment, city_name varchar(50) NOT NULL default '', latitude varchar(15) NOT NULL default '', longitude varchar(15) NOT NULL default '', population int(11) NOT NULL default '0', country_code char(2) NOT NULL default '', PRIMARY KEY (id)) TYPE=MyISAM AUTO_INCREMENT=1 ;

Столбец id - это первичный ключ (primary key), колонка, которая уникально идентифицирует каждый город. Тип данных этого столбца - INT (целое число нормального размера), MySQL назначает уникальные значения для этого столбца, благодаря атрибуту auto_increment. Обратите внимание на то, что мы не можем использовать название городов в качестве первичного ключа, т.к. некоторые имена городов не уникальны в мире. Для отображения численности населения мы также используем целое число.

Другие столбцы используют в качестве типов данных строки фиксированной длины (CHAR) либо строки переменной длины (VARCHAR). Когда мы точно знаем длину строки, лучше использовать CHAR, задавая длину столбца как CHAR(2). В противном случае, мы используем в качестве типа данных строки переменной длины, указывая только максимальную длину строки, например: VARCHAR(15).

После списка столбцов, мы можем указать дополнительные параметры таблицы, например, её тип, первое значение для столбца автоинкремента. SQL-выражение заканчивается точкой с запятой. Создав таблицу для городов, мы делаем ту же операцию, но на этот раз для таблицы стран.

CREATE TABLE countries (country_code char(2) NOT NULL default '', country_name varchar(100) NOT NULL default '') TYPE=MyISAM;

Заметьте, что столбец `country_code` присутствует в обеих таблицах. Это отражает принцип связи: country_code в `cities` связан с одноименным столбцом в таблице `countries`. Таким образом, мы экономим на месте, указывая название страны в базе данных только однажды.

В другой статье (Migrating to InnoDB) техника связывания рассматривается более подробно. После того как таблицы созданы, следует ввести в них какие-нибудь данные.

Изменение табличных данных с помощью phpMyAdmin.
В данном разделе мы изучим базовый синтаксис выражений INSERT, UPDATE, DELETE, и SELECT.

Добавление данных с помощью INSERT

Для начала изучим выражение INSERT, на примере кода, который генерирует phpMyAdmin при выполнении операции INSERT. Для этого открываем вкладку Insert на странице просмотра таблицы `countries`, и вводим данные о стране:

После того как мы кликаем на Go, данные записываются в таблицу и phpMyAdmin показывает нам использованное выражение INSERT:

INSERT INTO `countries` (`country_code` , `country_name`) VALUES ('ca', 'Canada');

После части INSERT INTO, следует имя таблицы. В MySQL, мы можем заключать имена таблиц и имена столбцов в обратные галочки "`", если в именах используются спецсимволы, зарезервированные слова. Затем мы открываем первую скобку, перечисляем столбцы в которые будет осуществлена вставка, разделяя их друг от друга запятыми. После перечисления списка названий столбцов скобка закрывается и указывается зарезервированное слово VALUES, после которого в скобках перечисляются значения которые нужно вставить в таблицу, причем перечисляются в том же порядке, что

и названия столбцов. Если значения имеют символьный тип данных, необходимо заключать их в кавычки.

Давайте занесем в таблицу `cities` данные города:

INSERT INTO `cities` (`id` , `city_name` , `latitude` , `longitude` , `population`, `country_code`) VALUES ('', 'Sherbrooke', '45 23 59.00', '-71 46 11.00', 125000, 'ca');

Здесь, мы указываем пустое значение для id, потому что атрибут автоинкремента данного столбца обеспечивает автоматическое выставление уникального значения. Также следует обратить внимание, что значение `population` - числовое, поэтому не заключено в кавычки.

Давайте закончим этот раздел вставкой некоторых данных для нескольких других стран и городов, которые понадобятся нам позже.

INSERT INTO `countries` (`country_code` , `country_name`) VALUES ('zh', 'China'); INSERT INTO `cities` (`id` , `city_name` , `latitude` , `longitude` , `population` , `country_code`) VALUES ('', 'Shanghai', '31 13 58.00', '121 26 59.99', 11000000, 'zh');

Обновление данных с помощью UPDATE

Сначала кликните на `Browse` для таблицы `cities` в результате будет выведена пока единственная запись. Кликая по иконке в виде карандаша на бумаге (или ссылке Edit), мы переходим на панель редактирования данной строки. Изменим значение столбца `population` на 130000. После щелчка на `Save`, phpMyAdmin отображает следующее выражение:

UPDATE `cities` SET `population` = '130000' WHERE `id` = '1' LIMIT 1 ;

Ключевое слово в данном выражении - `UPDATE`, за которым следует название таблицы. Слово `SET` предваряет список модификаций (в нашем случае - только для столбца `population`) который записывается в формате "столбец = новое значение".

Мы видим, что в выражении присутствует условие: WHERE `id` = '1', в котором используется первичный ключ, чтобы ограничить изменение значения столбца `population` только данной строкой, т.е. только для данного города.

Часть limit 1 добавляется phpMyAdmin-ом и является гарантией, что если первичный ключ не задан, изменения не будут применены более чем к одной записи. За один запрос `UPDATE` могут быть изменены значения сразу нескольких столбцов:

UPDATE `cities` SET `city_name` = 'Sherbrooke, Quebec', `population` = '130001' WHERE `id` = '1' LIMIT 1 ;

Удаление данных с помощью DELETE

В режиме `Browse` (просмотр) таблицы `cities`, кликните по красной иконке корзины (или ссылке Delete) - будет сгенерирован запрос, запрашивающий подтверждение выполнения следующего выражения: DELETE FROM `cities` WHERE `id` = '1' LIMIT 1 ;

Синтаксис здесь очень прост, и включает только название таблицы, и условие при котором будет выполнена операция удаления. Исключение условия WHERE из запросов UPDATE или DELETE вполне допустимо в SQL, но в таком случае действие выражения будет применено к каждой записи таблицы!

Выборка данных с помощью SELECT

Извлечение информации из таблиц - вероятно наиболее часто используемый вид запроса. Например, запросы SELECT позволяют получить ответы на подобные вопросы: "какие города имеют численность населения большую, чем данное число?". Фактически, мы уже предварительно использовали SELECT, когда кликали на ссылку Browse для таблицы `cities`. Это сгенерировало простейшую форму выражения запроса SELECT:

```
SELECT * FROM `cities` LIMIT 0,30;
```

Звездочка здесь означает "все столбцы". Мы добавили FROM и имя таблицы, в которой будет выполнен запрос на выборку. LIMIT 0,30 означает что выборка начинается с записи номер 0 (самой первой), и содержит максимум 30 записей.

Вкладка Search позволяет увидеть большее количество опций для запроса SELECT. Выберем вкладку Search для таблицы cities, и выберем только те столбцы, которые нам нужны. Затем справа от списка столбцов мы выберем порядок сортировки полученной выборки по столбцу `population` по убыванию:

В результате phpMyAdmin сгенерирует следующий запрос:

```
SELECT `city_name` , `population`
FROM `cities`
WHERE 1
ORDER BY `population` DESC LIMIT 0,30;
```

Мы видим, что звездочка была заменена списком столбцов, разделенных запятыми. Условие WHERE 1, добавленное phpMyAdmin-ом, всегда истинно и выбирает все записи. Чуть позже мы увидим, что можно заменить его другим условием. Кроме того, появляется условие ORDER BY , после которого следует название столбца по которому мы хотим сортировать результат выборки, и ключевое слово DESC для сортировки по убыванию (мы могли также использовать ASC для сортировки по возрастанию).

Условия в SQL-запросах
Самым простым способом добавить условие - клик по SQL-query: Edit, на странице результатов, в результате которого будет открыто всплывающее окно "Query". Добавим условие для столбца `country`:

```
SELECT `city_name` , `population`
FROM `cities`
WHERE country_code = 'zh'
ORDER BY `population` DESC;
```

Это условие выберет все города, находящиеся в Китае. При обозначении условий может быть использовано богатое множество операторов и функций. Вот - два примера:

Найти канадские города с численностью населения более 100000:
WHERE population > 100000 AND country_code = 'ca';
Найти города, чьи названия начинаются с символа "A":
WHERE city_name like 'A%'

Функции группировки
Итоговая информация может быть сгенерирована в результате группировки по определенному столбцу. Давайте узнаем среднюю численность городского населения в стране:

```
SELECT country_code, AVG(population)
```

```
FROM cities
GROUP BY country_code
```

Другие возможные функции группировки - MIN(), MAX(), SUM() и COUNT(), которые вычисляют соответственно минимальное значение, максимальное значение, сумму значений, и число записей. Например, с помощью следующего запроса мы можем получить число городов в стране:

```
SELECT country_code, count(city_name) FROM cities
GROUP BY country_code;
```

Объединения

Обычно, реляционная база данных включает множество таблиц, связанных общими ключами. Часто возникает необходимость в запросах сразу для нескольких таблиц. Связать, или объединить, таблицы можно с помощью различных методов; мы сосредоточимся на самом простом методе, заключающемся в сравнении ключей.

В нижерассмотренном запросе, условие FROM содержит список названий таблиц, разделенных запятыми. В списке столбцов, мы используем названия таблиц и точку в качестве префикса перед каждым названием столбца (в этом нет необходимости в случае, если все столбцы из одной таблицы).

```
SELECT cities.city_name, cities.population, countries.country_name
FROM cities, countries
WHERE cities.country_code = countries.country_code LIMIT 0,30
```

Заключение

Те элементарные выражения, что были рассмотрены нами здесь не раскрывают полностью возможности SQL. Однако, эта статья поверхностно осветила основы SQL, и показала как использовать phpMyAdmin в качестве инструмента для углубления знаний в SQL.

Об авторе. Марк Делисл начал свое участие в развитии phpMyAdmin в декабре 1998 (именно он разработал первую многоязычную версию). С мая 2001, принимал активное участие в проекте как разработчик и проектный администратор.

Работает с 1980 в колледже Cegep de Sherbrook (Квебек, Канада), разработчиком программного обеспечения и системным администратором. Он также преподавал организацию сети, безопасность, серверы Linux, и прикладное программирование PHP/MySQL.

Издательство Packt Publishing выпустило первую книгу по phpMyAdmin: Mastering phpMyAdmin for effective MySQL Management, автором которой является Марк Делисл