## **Изучение SQL с помощью phpMyAdmin**

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 блок "Создать новую таблицу в базе данных" на вкладке "Структура" 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)

) Engine=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 ''

) ENGINE=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.  
    
  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 "Вставить" 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 "Обзор" для таблицы `cities` в результате будет выведена пока единственная запись. Кликая по иконке в виде карандаша, мы переходим на панель редактирования данной строки. Изменим значение столбца `population` на 130000. После щелчка на "Сохранить", 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 "Обзор" таблицы `cities`, кликните по красному крестику напротив записи - 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 "Обзор" 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 "Поиск" 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: Изменить, на странице результатов, в результате которого будет открыто всплывающее окно "SQL". 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