

## Использование команд объединения таблиц базы данных MySQL

*Кочитов Михаил Евгеньевич*

*Приамурский государственный университет им. Шолом-Алейхема*

*студент*

### Аннотация

В данной статье рассматриваются команды объединения, которые используются для объединения нескольких таблиц в одну большую для предоставления более подробной информации. Также будет разработан пример с созданием двух таблиц, к которым будет применено объединение, используя запросы MySQL.

**Ключевые слова:** база данных, объединение, таблица, MySQL

## Using MySQL Database Join Tables Commands

*Kochitov Mikhail Evgenevich*

*Sholom-Aleichem Priamursky State University*

*student*

### Abstract

This article discusses the join commands that are used to join multiple tables into one large table to provide more detailed information. An example will also be developed to create two tables to which a join will be applied using MySQL queries.

**Keywords:** database, join, table, MySQL

Объединение в базах данных используется для того, чтобы определенные таблицы собрать в одну, указывая определенные столбцы, которые нуждаются в объединении. Структурированный язык запросов MySQL позволяет решить данную задачу путем внедрения запросов объединения на таблицы. Это поможет из определенных таблиц взять необходимые столбцы с данными и объединить их в одну большую таблицу, которая предоставит значительно больше данных, взятых из разных таблиц.

В статье С.Д. Точилина, Д.С. Точилина рассматривается производительность Internet-приложений поиска в данных MySQL и PostgreSQL [1]. Рассматривая статью О.В. Пастушенко, Е.Н. Сухарева можно увидеть проектирование базы данных корпоративной информационной системы кафедры в MySQL Workbench [2]. Р.Р. Алькаев, Д.Э. Водяков, В.А.Варюхин в своей статье рассмотрели анализ возможностей систем управления базами данных MS Access, MySQL, MS SQL Server [3]. В статье А.Е. Литвинова, Е.В. Измайловой рассматривается использование полнотекстового поиска в MySQL для определения похожих статей [4].

Рассматривая статью В.А. Чистова, А.В. Лукьянченко можно заметить автоматизацию масштабирования высоконагруженных баз данных MySQL [5].

Целью данной статьи является использование команд объединения таблиц базы данных, используя запросы языка MySQL. Также будет разработан пример с созданием двух таблиц и применен метод объединения самих таблиц в одну для предоставления подробной информации.

Базы данных работают на сервере, поэтому нужно использовать локальный веб-сервер OpenServer [6], который распространяется в интернете в свободном доступе. Также нужна программа для работы с базами данных, для примера будет использоваться СУБД Heidi SQL [7], которая также доступна в интернете в свободном доступе.

Теперь приступим к добавлению новой базы данных под названием «test» и в ней создадим для примера две таблицы «workers» (сотрудники) и «jobs» (профессии). Далее перейдем к настройке первой таблицы «workers» и добавим следующие столбцы

#	Имя	Тип данных	Длина/Знач...	Беззна...	Разреш...	Zerofill	По умолчанию
1	id	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	AUTO_INCREME...
2	name	VARCHAR	50	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	'0'
3	job	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL

Рисунок 1 – Таблица «workers»

На рисунке 1 изображена структура таблицы «workers» в ней видно добавленные три столбца: id – идентификатор сотрудника, name – имя сотрудника, job – номер профессии. Первый столбец id должно быть с первичным ключом и параметром авто инкремента. Последний столбец job должен иметь параметр разрешения пустых значений NULL. Теперь перейдем к настройке второй таблицы «jobs» и добавим в нее следующие столбцы

#	Имя	Тип данных	Длина/Знач...	Беззна...	Разреш...	Zerofill	По умолчанию
1	id	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	AUTO_INCREME...
2	name	VARCHAR	255	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	'0'

Рисунок 2 – Таблица «jobs»

На рисунке 2 показана структура таблицы «jobs» с двумя столбцами: id – идентификатор профессии (этот столбец должен иметь также первичный ключ и параметр авто инкремента), name – наименование профессии. Далее перейдем непосредственно к заполнению этих обеих таблиц


 id	name	job
1	Александр	1
2	Андрей	2
3	Михаил	(NULL)
4	Юлия	3
5	Светлана	(NULL)
6	Сергей	5
7	Наталья	3
8	Евгений	4
9	Мария	4
10	Анатолий	2

Рисунок 3 – Таблица «workers» с добавленными данными сотрудников

На рисунке 3 изображены данные из таблицы «workers». В этой таблице содержится 10 сотрудников, но сотрудники под номером 3 и 5 имеют пустые значения в столбце «job» это означает, что данные сотрудники без профессии. Значения NULL нужны обязательно, чтобы показать некоторые особенности объединения в MySQL. Далее заполним данными вторую таблицу профессий


 id	name
1	Менеджер
2	Программист
3	Повар
4	Бухгалтер
5	Писатель
6	Учитель

Рисунок 4 – Таблица «jobs» с добавленными данными профессий

На рисунке 4 изображена таблица «jobs» с добавленными данными из шести профессий. Эта таблица создана исключительно для того, чтобы в предыдущей таблице «workers» в столбце «job» с помощью команд объединения отобразить название самих профессий для каждого сотрудника, а не их номера. Теперь после того как были созданы две таблицы с внесенными данными, то можно на локальном сервере создать PHP файл «index.php» и вписать в него следующий код

```
<?php
$link = new mysqli("localhost", "root", "root", "test");
$query_text = "
    SELECT * FROM workers
";
$query = mysqli_query( $link, $query_text );

echo '<table border=1>';
echo '<tr>';
echo '<th>Номер</th>';
echo '<th>Имя</th>';
echo '<th>Профессия</th>';
echo '</tr>';
while ( $row = mysqli_fetch_row( $query ) ) {
    echo '<tr>';
    echo '<td>'.$row[0].</td>';
    echo '<td>'.$row[1].</td>';
    echo '<td>'.$row[2].</td>';
    echo '</tr>';
}
echo '</table>';
echo '<br><br>';
echo '<table border=1>';
echo '<tr>';
echo '<th>Номер</th>';
echo '<th>Профессия</th>';
echo '</tr>';

$query_text = "
    SELECT * FROM jobs
";
$query = mysqli_query( $link, $query_text );

while ( $row = mysqli_fetch_row( $query ) ) {
    echo '<tr>';
    echo '<td>'.$row[0].</td>';
    echo '<td>'.$row[1].</td>';
    echo '</tr>';
}
echo '</table>';
?>
```

Рисунок 5 – PHP код подключения к базе данных и отображения на веб-странице двух таблиц с данными

На рисунке 5 продемонстрирован PHP код, который изначально подключается к базе данных «test», используя логин «root» и пароль «root». Далее формируется MySQL запрос в переменной \$query\_text, который позволяет отобразить все данные из всех столбцов таблицы «workers». После этого переменная с запросом передается в функцию «mysqli\_query» и в переменную \$query вносится полученный после запроса результат, который в дальнейшем в цикле в функции «mysqli\_fetch\_row» будет перебирать все данные каждого столбца из таблицы «workers». В PHP коде используется метод «echo», который на веб-страницу добавляет HTML код двух таблиц «workers» и «jobs». Теперь глянем результат работы данного PHP кода на браузере и рассмотрим на веб-странице созданные две таблицы с данными

Номер	Имя	Профессия
1	Александр	1
2	Андрей	2
3	Михаил	
4	Юлия	3
5	Светлана	
6	Сергей	5
7	Наталья	3
8	Евгений	4
9	Мария	4
10	Анатолий	2

Номер	Профессия
1	Менеджер
2	Программист
3	Повар
4	Бухгалтер
5	Писатель
6	Учитель

Рисунок 6 – Отображенные с данными две таблицы «workers» и «jobs» на веб-странице

На рисунке 6 отображены две таблицы «workers» и «jobs», в которых содержатся данные такие же, что были на рисунках 3 и 4. Далее перейдем уже к объединению этих двух таблицы и рассмотрим, какие есть виды объединений.

Первое объединение называется INNER JOIN (внутреннее объединение), оно используется для объединения таблиц, но только если в значениях выбираемых столбцов имеются данные, то есть значения, содержащие пустые значения NULL будут не рассматриваться внутренним объединением INNER JOIN. Теперь вернемся к файлу «index.php» и немного изменим код

```
<?php
$link = new mysqli("localhost", "root", "root", "test");

$query_text = "
SELECT
    workers.id,
    workers.name,
    jobs.name AS job
FROM workers
JOIN jobs
    ON workers.job = jobs.id
";

$query = mysqli_query( $link, $query_text );

echo '<table border=1>';
echo '<tr>';
echo '<th>Номер</th>';
echo '<th>Имя</th>';
echo '<th>Профессия</th>';
echo '</tr>';
while ( $row = mysqli_fetch_row( $query ) ) {
    echo '<tr>';
    echo '<td>'.$row[0]. '</td>';
    echo '<td>'.$row[1]. '</td>';
    echo '<td>'.$row[2]. '</td>';
    echo '</tr>';
}
```

Рисунок 7 – PHP код с запросом внутреннего объединения INNER JOIN

На рисунке 7 показан PHP код, в котором используется запрос внутреннего объединения таблиц INNER JOIN. Теперь рассмотрим данный запрос более подробно. В запросе используется команда «SELECT», которая указывает, какие столбцы нужно использовать для отображения их на веб-странице. Далее оператор «JOIN» используется для выбора таблицы, которая будет объединяться. Оператор «ON» указывает в запросе, какие столбцы из таблицы «jobs» нужны для объединения в таблицу «workers». Данный запрос позволяет в третьем столбце «workers» отобразить не сами номера, а название самих профессий, то есть третьим столбцом будет объединенный столбец «name» из таблицы «jobs», а внутреннее объединение исключит сотрудников, у которых нет профессий. Теперь перейдем к веб-странице и рассмотрим результат сформированной таблицы с помощью внутреннего объединения

Номер	Имя	Профессия
1	Александр	Менеджер
2	Андрей	Программист
10	Анатолий	Программист
4	Юлия	Повар
7	Наталья	Повар
8	Евгений	Бухгалтер
9	Мария	Бухгалтер
6	Сергей	Писатель

Рисунок 8 – Таблица сотрудников с профессиями, используя внутреннее объединение INNER JOIN

Как видно на рисунке 8 изображена таблица, в которой в третьем столбце отображены названия профессий, но как можно заметить были исключены из содержимого таблицы сотрудники под номером 3 и 5, у которых не указаны профессии, то есть у них были пустые значения NULL.

Далее перейдем к следующему типу объединения это к внешнему левому и правому объединению LEFT JOIN и RIGHT JOIN. Внешнее объединение отличается от внутреннего тем, что оно будет отображать данные столбцов с пустыми значениями, то есть сотрудников, не имеющих профессию. Теперь рассмотрим следующий PHP код и его измененный запрос

```
<?php
$link = new mysqli("localhost", "root", "root", "test");

$query_text = "
SELECT
    workers.id,
    workers.name,
    jobs.name AS job
FROM workers
LEFT JOIN jobs
    ON workers.job = jobs.id
";

$query = mysqli_query( $link, $query_text );

echo '<table border=1>';
echo '<tr>';
echo '<th>Номер</th>';
echo '<th>Имя</th>';
echo '<th>Профессия</th>';
echo '</tr>';
while ( $row = mysqli_fetch_row( $query ) ) {
    echo '<tr>';
    echo '<td>'.$row[0]. '</td>';
    echo '<td>'.$row[1]. '</td>';
    echo '<td>'.$row[2]. '</td>';
    echo '</tr>';
}
```

Рисунок 9 – PHP код с запросом внешнего левого объединения LEFT JOIN

На рисунке 9 изображен PHP код, в котором указан запрос внешнего левого объединения LEFT JOIN, по предыдущему запросу внутреннего объединения INNER JOIN в содержимом запросе просто изменилось одно слово с INNER на LEFT, все остальное в запросе осталось таким же. Внешнее левое объединение позволяет объединить левую таблицу «jobs» к таблице «workers» с отображением всех названий профессий, даже у тех сотрудников, у которых ее нет. Теперь рассмотрим результат работы построенной таблицы на веб-странице с применением внешнего левого объединения

Номер	Имя	Профессия
1	Александр	Менеджер
2	Андрей	Программист
3	Михаил	
4	Юлия	Повар
5	Светлана	
6	Сергей	Писатель
7	Наталья	Повар
8	Евгений	Бухгалтер
9	Мария	Бухгалтер
10	Анатолий	Программист

Рисунок 10 – Таблица сотрудников с профессиями, используя внешнее левое объединение LEFT JOIN

На рисунке 10 представлена таблица, созданная на веб-странице с помощью левого внешнего объединения. В самом содержимом таблицы можно увидеть, что имеются все 10 сотрудников и даже с номером 3 и 5, у которых профессия имеет пустое поле. Теперь рассмотрим последний тип объединения это правое внешнее объединение RIGHT JOIN. Рассмотрим следующий PHP код

```
<?php
$link = new mysqli("localhost", "root", "root", "test");
$query_text = "
SELECT
    workers.id,
    workers.name,
    jobs.name AS job
FROM jobs
RIGHT JOIN workers
    ON workers.job = jobs.id
";
$query = mysqli_query( $link, $query_text );

echo '<table border=1>';
echo '<tr>';
echo '<th>Номер</th>';
echo '<th>Имя</th>';
echo '<th>Профессия</th>';
echo '</tr>';
while ( $row = mysqli_fetch_row( $query ) ) {
    echo '<tr>';
    echo '<td>'.$row[0]. '</td>';
    echo '<td>'.$row[1]. '</td>';
    echo '<td>'.$row[2]. '</td>';
    echo '</tr>';
}
```

Рисунок 11 – PHP код с запросом внешнего правого объединения RIGHT JOIN



На рисунке 11 показан PHP код, в котором используется запрос внешнего правого объединения RIGHT JOIN. Оно отличается от запроса левого внешнего объединения LEFT JOIN тем, что в запросе указано отображение таблицы «jobs», но объединением будет использоваться уже таблица «workers», однако результат будет тот же, что и с внешним левым объединением. Теперь рассмотрим результат сформированной таблицы

Номер	Имя	Профессия
1	Александр	Менеджер
2	Андрей	Программист
3	Михаил	
4	Юлия	Повар
5	Светлана	
6	Сергей	Писатель
7	Наталья	Повар
8	Евгений	Бухгалтер
9	Мария	Бухгалтер
10	Анатолий	Программист

Рисунок 12 – Таблица сотрудников с профессиями, используя правое внешнее соединение RIGHT JOIN

Как можно заметить на рисунке 12 результат работы запроса сформировал ту же самую таблицу, что и на рисунке 10, поэтому для примера рассматриваемых двух таблиц достаточно внутреннего объединения INNER JOIN и внешнего левого объединения LEFT JOIN.

Таким образом, были рассмотрены команды объединения таблиц базы данных MySQL, которые позволяют несколько таблиц объединить в одну для предоставления более подробной информации. Также были созданы в базе данных две тестируемые таблицы, подвергнувшиеся к различным видам объединений.

### Библиографический список

1. Точилин С.Д., Точилин Д.С. Производительность internet-приложений поиска в данных MySQL и PostgreSQL // Вестник Херсонского национального технического университета. 2011. № 1 (40). С. 204-207.
2. Пастушенко О.В., Сухарев Е.Н. Проектирование базы данных корпоративной информационной системы кафедры в MySQL Workbench // Решетневские чтения. 2010. Т. 2. С. 513-514.
3. Алькаев Р.Р., Водяков Д.Э., Варюхин В.А. Анализ возможностей систем управления базами данных MS Access, MySQL, MS SQL Server // Apriori. Серия: Естественные и технические науки. 2015. № 6. С.

- 4.
5. Литвинов А.Е., Измайлова Е.В. Использование полнотекстового поиска в MySQL для определения похожих статей // *Juvenis Scientia*. 2016. № 2. С. 14-15.
6. Чистов В.А., Лукьянченко А.В. Автоматизация масштабирования высоконагруженных баз данных MySQL // *Современные наукоемкие технологии*. 2016. № 6-2. С. 315-319.
7. Open Server Panel – Локальный веб-сервер для Windows URL: <https://ospanel.io/> (дата обращения 23.08.2020)
8. Heidi SQL URL: <https://www.heidisql.com/> (дата обращения 23.08.2020)
9. Поговорим о JOIN URL: <https://anton-pribora.ru/articles/mysql/mysql-join/> (дата обращения 23.08.2020)
10. SQL Соединение таблиц и виды JOIN URL: <https://office-menu.ru/uroki-sql/92-sql-join> (дата обращения 23.08.2020)