

ХАНТИМ

УДОБНЫЙ СПОСОБ СЛЕДИТЬ
ЗА НОВЫМИ ВАКАНСИЯМИ В IT-ИНДУС[войти](#) [зарегистрироваться](#)[посты](#) [q&a](#) [события](#) [хабы](#) [компании](#)

ХАНТИМ

Несколько интересных приемов и особенностей работы с MySQL из песочницы

MySQL*

Я думаю, что в процессе изучения той или иной СУБД каждый из вас не раз изобретал велосипеды для решения своих задач, не зная о существовании той или иной функции или приема, которые бы могли в разы ускорить выполнение запросов и уменьшить объем кода. В данной статье я хочу поделиться с вами своим опытом работы с очень «добрым» и «отзывчивым» MySQL, часто позволяющему программисту делать вещи, которые другие СУБД переварить бы не смогли. Материал будет полезен скорее тем, кто только решил углубиться в чудесный мир запросов, но возможно и опытные программисты найдут тут что-то интересное.

Удаление дубликатов

Очень часто на различных специализированных ресурсах мне встречались вопросы о том, как быстрее и оптимальнее избавиться от дублирующихся записей в таблице. Сразу же в голову приходит то, что нужно создать еще одну таблицу, идентичную данной, создать в ней уникальный ключ и скопировать в нее данные из исходной таблицы, скажем, с помощью INSERT IGNORE. Но существует и более простой способ. Достаточно просто создать в таблице уникальный ключ с помощью такого вот запроса:

```
ALTER IGNORE TABLE table1 ADD UNIQUE (field1, field2);
```

После добавления ключа все дубликаты удалятся автоматически.

Преобразование строки в число

Допустим перед вами встала задача поиска в таблице адресов address дома с определенным номером. Причем номер дома хранится в текстовом поле num, содержащим значения типа '1', '1a', '1/б', '2ы', 'Зйцуken' и т.п. И мы хотим выбрать дома с номером, состоящим из 1 и еще каких-то символов. Думаю многие сразу кинутся искать решение с использованием LIKE или REGEXP. Но проще будет использовать следующую фишку MySQL:

```
SELECT *  
FROM address  
WHERE num + 0 = 1;
```

Встретив арифметическую операцию, MySQL автоматически приводит все аргументы к числовому типу. В случае со строками будут просто отсечены все символы, начиная с первого не числового.

Вот такой вот запрос тоже спокойно выполнится без ошибок:

Лучшее за 24 часа ↓

[Скончался знаменитый физик С.П.Капица](#)[Ключевое слово this в javascript — учимся определять контекст на практике](#)[RuTracker взломан](#)[Ким Дотком возвращается, намекает на возрождение MegaUpload](#)[Несколько интересных приемов и особенностей работы с MySQL](#)[Зашифрованный файл-страховка Wikileaks \(64 ГБ\)](#)[Рабочие места известных IT-руководителей](#)[Программа zxcvbn: реалистичная оценка надежности пароля](#)[Марсоход "Кьюриосити": объяснение выбора 2 Мп фотокамер](#)[Заморски иски читать бросай ты, как ни бесили. Секретны списки запретных сайтов теперь в России!](#)[« все лучшие](#)

```
SELECT '1qwe3s' + '2regt3g';
```

И в результате мы получим ответ: 3.

Использование переменных в запросах

Тут я сразу приведу пример решения задачи, в которой переменная облегчит нам жизнь.
Имеем следующую таблицу table1:

id	sum
1	35
2	25
3	10
4	55
5	12

Нужно вывести все эти поля и добавить к ним еще 2, onStart и total.
total = summ — onStart.
onStart равен значению total из предыдущей записи, для первой записи onStart = 0.
Т.е. в итоге мы должны получить такой вот результат:

id	sum	onStart	total
1	35	0	35
2	25	35	- 10
3	10	- 10	20
4	55	20	35
5	12	35	-23

Используя переменную, мы сможем при решении данной задачи избавиться от лишних JOIN'ов и подзапросов:

```
SELECT t1.id, t1.summ, @i AS onStart, @i := t1.summ - @i AS total
FROM table1 t1
JOIN (SELECT @i := 0) var;
```

Подсчет количества различных записей в таблице

Еще одна часто встречающаяся задача. И тут я тоже сразу приведу пример.
Дана таблица table1 (id, f1, f2). Нужно написать запрос, который бы вернул нам следующий результат:

общее количество записей	количество записей с f1 = 1	сумму значений f2 для f1 = 2
--------------------------	-----------------------------	------------------------------

Конечно можно получить результат вот так:

```
SELECT COUNT(1),
       (SELECT COUNT(1) FROM table1 WHERE f1 = 1),
       (SELECT SUM(f2) WHERE f1 = 2)
FROM table1;
```

Но очевидно, что это далеко не оптимальное решение. Придется для каждой записи выполнять еще два дополнительных подзапроса. И мы сделаем по-другому:

Похожие посты ↓

- [Парное программирование в аутсорсинге: достижение взаимопонимания с техническими специалистами заказчика](#) 31.07.2012
 - [SQL injection для начинающих. Часть 1](#) 20.07.2012
 - [База по языкам программирования: Синтаксический сахар или история развития языков](#) 11.07.2012
 - [База по языкам программирования: Принцип сохранения функционала](#) 11.07.2012
 - [База по языкам программирования: Как появлялись языки и зачем](#) 11.07.2012
 - [MySQL в миллион раз быстрее MemSQL](#) 27.06.2012
 - [Язык программирования Gentee](#) 18.06.2012
 - [Программирование под MeeGo для начинающих](#) 13.06.2012
 - [SQL Server 2012 — что нового для разработчика?](#) 13.05.2012
 - [Самый частый шаблон SQL инъекций в PHP — бесполезное экранирование символов](#) 23.04.2012
- Прямой эфир ↓
- [edogs → 8 успешных лет freelance'a, tips and tricks](#) 31
 - [iMedved → А слона то я и не заметил. Рассказ о обделенной вниманием Diem CMS](#) 18
 - [bugman → Защита от SQL-инъекций в PHP и MySQL](#) 91
 - [Karamax → Мультизагрузочный CD с использованием GRUB](#) 2
 - [disinvis → Apple предлагал Samsung купить лицензию на патенты по 30 долларов за смартфон](#) 110
 - [PiSaik → Точка зрения инди-разработчиков: как добиться успеха](#) 2
 - [max_posedon → Небольшой тест скорости QCoreApplication](#) 3
 - [equand → Несколько интересных приемов и особенностей работы с MySQL](#) 17
 - [dotsquid → Платформа для экспериментов на Three.js и WebGL](#) 7
 - [sexyhippo → «Простой бизнес» Версия 1.7. – Простой агент становится еще удобнее для управления Вашим бизнесом](#) 10

```
SELECT COUNT(1),
       SUM(f1 = 1),
       SUM(IF(f1 = 2, f2, 0))
FROM table1;
```

Теперь другое дело. Всё, что нам нужно, мы посчитали за один проход по таблице.

Column 'id' in group statement is ambiguous

В этой части статьи я хочу обратить ваше внимание на одну интересную особенность MySQL.

Имеем такой запрос:

```
SELECT t1.id, t2.id
FROM table1 t1
JOIN table2 t2 ON t1.id = t2.id_t1
GROUP BY id;
```

Видно, что в блоке GROUP BY мы забыли указать алиас у поля id, и соответственно при попытке выполнить запрос получили ошибку «Column 'id' in group statement is ambiguous». Казалось бы всё верно. Теперь изменим этот запрос:

```
SELECT t1.id, t2.f1
FROM table1 t1
JOIN table2 t2 ON t1.id = t2.id_t1
GROUP BY id;
```

Мы убрали из списка выводимых полей t2.id и, о чудо, запрос отработал, данные были сгруппированы по t1.id. Другие СУБД, такие как, например, MS SQL или PostgreSQL и во втором случае выдали бы ошибку, но для MySQL второй запрос полностью корректен.

Так что я рекомендую вам быть более внимательными и всегда использовать алиасы перед полями, иначе потом при небольшом изменении запроса можно нарваться на ошибку.

Поиск данных за последнюю дату

И напоследок хочу привести еще один пример решения одной типичной не сложной часто встречающейся задачи. Почему-то у многих она часто вызывает затруднения.

Дана таблица платежей payments (id INT, uid INT, pay_date DATE, amount DECIMAL(15, 2)).

id – первичный ключ

uid – идентификатор юзера

pay_date – дата платежа

amount – сумма платежа

Нужно написать запрос, который бы вывел для каждого юзера дату и сумму последнего платежа.

Я предлагаю вам следующее стандартное решение:

```
SELECT p.uid, p.amount
FROM payments p
JOIN
  (SELECT uid, MAX(pay_date) AS max_dt
   FROM payments
   GROUP BY uid) sel ON p.uid = sel.uid AND p.pay_date = sel.max_dt;
```

[mysql](#), [программирование](#), [sql](#)



14 августа 2012 в 17:43

360

AlexeyVD^{49,1}

Система защиты
виртуализованных
инфраструктур
от вредоносного кода

**Управление
безопасностью**
широкого ряда устройств
из единой консоли
администрирования

Достижение **более
высокой
производительности**
и консолидации

**Централизованные
средства**
менеджмента безопасности

Компания дня ?

Ivideon

Последний пост: [It happens — самое необычное применение Ivideon](#)

53 подписчика

Q&A ↓

[ipadm](#) → [Ubuntu 10.10 Обновить до LTS или откатить до LTS?](#) 5

[cjey](#) → [C# получение данных из SQL. Как реализовать?](#) 1

[ipadm](#) → [3G интернет в селе — самостоятельный подбор компонентов или готовое устройство?](#) 2

[Frag](#) → [Подскажите сайты с примерами кода на C#](#) 7

[sOlaris](#) → [Системы организации рассылок](#) 3

[opium](#) → [Какие курсы по автоматизации linux-администрирования есть в Москве?](#) 4

[opium](#) → [Как в Google Analytics создать фильтр по произвольному параметру ссылки?](#) 1


[VadikSS](#) → [Подскажите недорогой и хороший android tv box](#) 4

[stepio](#) → [Нужны ли хабру статьи о концепции "Интернет вещей"?](#) 2

комментарии (17)

[sunsey](#) → [PDO - полный отладочный запрос](#) 7


« [все вопросы](#)

**gibson** 14 августа 2012 в 18:34 # 0


спасибо, полезно — добавил пост в избранное

**nooze** 14 августа 2012 в 18:43 # -5


Интересные решения. Конечно не для рабочего варианта базы, а скорее как единичные запросы запускаемые вручную.

**parpalak** 14 августа 2012 в 19:11 # +1


Мне кажется, в подсчете количества различных записей в таблице вы предлагаете не оптимальный путь, потому что в нем не будут использоваться индексы. Другой способ — попробовать разбить один запрос на три и подобрать индексы. В конкретной задаче EXPLAIN SELECT подскажет, какой из способов быстрее.

**Shedal** 14 августа 2012 в 19:36 # ↑ +1


За что минусовали человека? Мне тоже кажется, что `SUM(IF(f1 = 2, f2, 0))` индексы использовать если и будет, то гораздо менее оптимально, т.к. `IF` — это функция, и в неё нужно будет подать **каждое** существующее значение `f1`. В то же время, `SELECT SUM(f2) WHERE f1 = 2` прекрасно использует индекс по условию, а потом уже просуммирует отфильтрованные данные.

**niga** 14 августа 2012 в 20:49 # ↑ -2

Насколько я помню, MySQL не использует индексы в подзапросах.

**kuber** 14 августа 2012 в 20:28 # ↑ 0

Вы правы.
При наличии индексов у полей `f1` и `f2` запрос предложенный автором в реальности (за исключением специально подогнанных случаев) будет работать заметно медленнее, чем запрос, который автор называет не оптимальным.

**Shedal** 14 августа 2012 в 19:41 # +2


Поиск данных за последнюю дату

```
...
SELECT p.uid, p.amount
FROM payments p
JOIN
  (SELECT uid, MAX(pay_date) AS max_dt
   FROM payments
   GROUP BY uid) sel ON p.uid = sel.uid AND p.pay_date =
sel.max_dt;
```

В результатах будут дубликаты, если один пользователь совершил несколько платежей в один день.

**niga** 14 августа 2012 в 20:38 # ↑ -3

По моему GROUP BY не допустит этого.

**kuber** 14 августа 2012 в 21:14 # ↑ 0

Вы ошибаетесь. Будут там дубликаты, при условии, что пользователь совершил несколько платежей в последний день.

**equand** 14 августа 2012 в 21:19 # ↑ 0

`distinct` не допустил бы этого, `group by` группирует тупо

**easterism** 14 августа 2012 в 19:53 # +3

О, работа! ↓

- [PHP программист](#)
- [Senior PHP developer](#)
- [TEAM LEADER группы разработчиков \(API, Web Services\)](#)
- [Ведущий программист/Ведущий разработчик \(API, Web Services\)](#)
- [Программист/Разработчик \(API, Web Services\)](#)
- [Ведущий разработчик C#/ .NET](#)
- [Руководитель отдела маркетинга и интернет-продаж](#)
- [Веб-разработчик](#)
- [Flash разработчик](#)
- [Разработчик на Python и Perl](#)
- [Веб-дизайнер](#)
- [Программист в научно-исследовательский департамент](#)
- [Разработчик драйверов](#)
- [PHP разработчик](#)
- [Java-разработчик](#)
- [Замечательный Linux администратор](#)
- [Web/UI дизайнер](#)
- [PHP разработчик \(крупная соцсеть\)](#)
- [Программист 1С-Битрикс](#)
- [PHP разработчик](#)
- « [все вакансии](#)

Ближайшие события

- 15** авг Обновленный курс по блейд-системам HP запланирован к проведению с 15 августа
- 16** авг Конференция для мобильных разработчиков mdday#ua
- 16** авг Вебинар "Оценка устойчивости к атакам типа DoS/DDoS"
- 16** авг Развитие облачных технологий: российский взгляд
- 16** авг Viber iOS meetup – для настоящих гурю iOS разработки. Он-лайн трансляция.
- « [все события](#)

Column 'id' in group statement is ambiguous


Я не понял «о чудо» этого абзаца. Очевидно же что для GROUP, id двусмысленный. Так добавьте ему смысла.

```
SELECT t1.id, t2.id
FROM table1 t1
JOIN table2 t2 ON t1.id = t2.id_t1
GROUP BY t1.id;
```

 **DoctorGester** 14 августа 2012 в 20:36 #

0

Плохо разбираюсь в MySQL, не подскажите, что за ключевое слово ON? Гугл плохо ищет по коротким словам.

 **Shedal** 14 августа 2012 в 20:40 # ↑

0

ON используется в совокупности с JOIN'ами:
dev.mysql.com/doc/refman/5.0/en/join.html

 **niga** 14 августа 2012 в 20:43 # ↑

0

После ON следует условие для выборки из связанных таблиц в случае JOIN.

 **doxa** 14 августа 2012 в 20:45 # ↑

0

ON — аналог WHERE, используемый при JOIN.

 **DoctorGester** 14 августа 2012 в 20:55 # ↑

0

Всем спасибо, все понял.

 **thebestzorro** 14 августа 2012 в 20:59 #

-1

Спасибо, хорошие рецепты.

Только зарегистрированные пользователи могут оставлять комментарии. [Войдите](#), пожалуйста.

User experience design: как построить сайт для клиентов, а не для себя

Автоматическая подсветка лестницы с помощью Arduino

Сеть на кристалле — мини-интернет внутри процессора

Войти	Разделы	Посты	Инфо	Услуги	<div>© 2006–2012 «Тематические Медиа»</div> <div>Служба поддержки: support@habrahabr.ru</div> <div>Мобильная версия</div>
Регистрация	Q&A	Лучшие	О сайте	Реклама	
	Хабы	Тематические	Правила	Корпоративные пакеты	
	События	Корпоративные	Помощь	Семинары	
	Компании	Песочница	Соглашение		
	Работа		Статистика		
	Люди				