Построение таблиц и обращение к их содержимому — настолько простая и понятная вещь, что многие, освоив основные приемы, попросту перестают изучать синтаксис построения и оптимизации таблиц более глубоко.

А между тем, одна только индексация таблиц порой поднимает производительность сайта в несколько раз.

Что же такое — индексация? Попробую объяснить на примере.

Вспомните любую публичную библиотеку. Пусть детскую или даже школьную. Помните зал со стеллажами книг? И даже если вы были очень давно в библиотеке, вы прекрасно знаете, что все книги в этом уважаемом заведении расставлены не абы как, не в порядке их поступления в библиотеку (как поступают данные в базу), а по каким-то правилам. Обычно, книги разносят по темам, авторам и по алфавиту.

Я думаю, излишне объяснять, зачем все это делается, и почему библиотекари так ревностно следят за порядком размещения книг на стеллажах. Но я позволю себе обратить ваше внимание на сравнительную эффективность поиска в такой структурированной системе, которой, кстати, обычно пренебрегают при построении и использовании компьютерных баз данных.

Предположим, вы в библиотеке ищите какую-то книгу.

Если вы начнете тупо перебирать все книги в библиотеке, то у вас на это уйдет не один день или даже не один месяц, если это крупная библиотека.

Но если вы знаете автора или год или тему книги, то, подойдя к соответствующим стеллажам, вы найдете издание за несколько минут, а то и секунд. Как, собственно, это и бывает в библиотеке.

Так какого же хрена, прошу прощения за резкость, мы заставляем наши компьютеры искать необходимые нам данные методом тупого перебора? Только потому, что они это делают быстро?

Да, быстро. Но это если поиск надо произвести в тысячах записей. А если речь идет о миллионах? Или вы думаете, что ваша база данных не так велика, чтобы заниматься ее оптимизацией? Ошибаетесь, дорогие мои. Как только к вашей базе обратятся сотни человек, так ваши тысячи записей тут же превратятся для движка баз данных в миллионы! И ваш провайдер совершенно резонно сделает вам замечание.

Итак. В том, что базы данных надо индексировать — не сомневается ни один здравомыслящий программист. Правильно построенные индексы позволяют находить нужную информацию "в одно касание".

Как это происходит технически — нам знать не обязательно. Достаточно иметь в виду, что MySQL, как и любой другой движок баз данных, использует дополнительное место на диске для хранения индексных файлов. Это надо знать, только для того, чтобы не плодить ненужных индексов. Во всем надо знать меру. Даже в таком полезном деле, как индексация таблиц.

Так давайте сразу и определимся с тем, что нам надо индексировать.

А индексировать надо те поля таблицы, по которым происходит поиск или сортировка данных.

Например, у нас есть таблица books и таблица authors. В первой мы храним информацию о книгах, а во второй — информацию об их авторах.

Разумеется, самое логичное для подобной базы — искать в ней книги по названию и по автору.

Не имея индексации такой базы, примитивный поиск по первым буквам произведения вынудит компьютер просмотреть все записи в таблице, чтобы выдать полный результат. Если база большая, на это уйдет время.

Если же мы добавим в таблицу books индекс по полю book_name (название книги), то MySQL создаст индекс этой таблицы. То есть, отсортирует таблицу по указанному полю и расставит метки и ссылки на ячейки в реальной таблице.

Другими словами, индекс — это копия таблицы без данных, но отсортированная по определенным правилам, где каждая ячейка — есть ссылка на родительскую ячейку в основной таблице. Надеюсь, что выразился достаточно ясно.

Построив индекс по полю book_name, наш компьютер найдет по первым буквам нужные нам произведения практически мгновенно, ведь ему теперь не надо перебирать всю таблицу, а достаточно посмотреть на ту ее часть, где хранятся записи на нужную букву.

Еще проще говоря, если мы ищем книги на букву "М", то компьютер не станет перебирать записи, начинающиеся на другие буквы, прекрасно зная, что там нет записей, удовлетворяющих запросу.

И если в нашей книжной базе находится 100.000 книг, среди которых только 30 на букву Ж, то по запросу "найти все книги на букву Ж", компьютер переберет только 30 записей при наличии индекса, или переберет 100.000 записей при его отсутствии.

По-моему, польза очевидна.

Кстати, индексация текстовых полей — занятие чуть более сложное, чем индексация полей другого типа. Поясню. Цифровые, буквенные, булевы, поля дат, времени и другие — индексируются, как правило, без каких либо дополнительных размышлений.

Скажем, если в таблице книг есть поле book_date, хранящее дату публикации книги, то добавление индекса к такому полю будет выглядеть примерно так:

alter table tbl_books add index i_date (book_date);

Эта директива указывает MySQL создать индекс по полю book_date.

Теперь компьютеру не составит труда найти все книги 1993-го года или выстроить все найденные книги в порядке даты их публикации.

Точно так же можно создать индексы для других полей, по которым нам интересно производить поиск или сортировку.

Создание же индекса по текстовому полю осложняется только тем, что надо указать количество символов от начала записи, по которым надо построить индекс.

И тут вы уже сами должны оценить размер базы, похожесть первых символов разных записей и так далее.

Например, если мы строим индекс по названиям улиц, то нам не надо индексировать их по всей длине, а можно предположить, что подавляющее большинство записей начнут различаться уже где-то на пятой букве.

А если мы индексируем коды шариковых подшипников, то индексировать лучше как можно больше символов, ибо большое количество изделий могут нести коды, отличающиеся только последними буквами или цифрами.

Иногда нам не надо специально строить индексы. Достаточно того, что MySQL сам построит индексы по полю, если при объявлении структуры таблицы мы зададим полю уникальность UNIQUE, говорящую о том, что данное поле не может хранить два одинаковых значения.

Другое ключевой слово, создающее индексы — KEY. Помните объявление primary key, которое я обязательно использую в каждой таблице для id-поля.

Если вы хотите увидеть, какие поля в таблице проиндексированы, дайте команду MySQL:

desc tbl_name;

и MySQL выведет всю информацию о таблице tbl_name, включая отметки об индексации полей.

Или более подробно, только об индексах:

show index from tbl_name;

Остается еще отметить, что в индексе может участвовать множество полей. Не обязательно одно.

Если в базе накладных чаще всего производится поиск по сумме с учетом диапазона дат, то логично создать индекс по этим двум полям: сумма и дата. Именно, создать один индекс по двум полям, а не два индекса по каждому полю!

alter table tbl_name add index i_name (field1, field2);

Поиск по уникальным индексам производится чуть быстрее, поэтому, если значение какого-то поля должно быть обязательно уникальным — не поленитесь отметить это в конструкции таблицы. MySQL отблагодарит вас скоростью своих ответов на ваши вопросы.

О синтаксисе создания индексов лучше всего написано тут: http://www.mysql.com/doc/A/L/ALTER_TABLE.html.

Прежде чем попрощаться, я бы хотел подсказать вам одну полезную директиву MySQL. Называется она explain.

Если вы поставите это слово перед любым запросом к базе данных, MySQL не станет выводить вам результат запроса, а покажет подробную информацию о том, какими средствами пришлось воспользоваться и сколько операций пришлось произвести, чтобы получить ответ на ваш запрос.

Это волшебное слово **explain** позволит оценить эффективность любого запроса и отрегулировать все индексы вашей базы.

Удачной индексации, и да прибудет с вами порядок!