**Знакомство с SQLite с клиентом и базой данных**

Краткое описание программного обеспечения базы данных SQLite, в частности, как перемещаться по файлу базы данных SQLite с использованием клиента SQLite.

**Оглавление**

* + [**Быстрый старт и обучение**](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#quick-start-and-learn-by-doing)
  + [Загрузите и запустите DB Browser для клиента SQLite](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#download-and-launch-the-db-browser-for-sqlite-client)
  + [Загрузить файлы базы данных SQLite](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#download-sqlite-database-files)
  + [Открытие файла базы данных sqlite](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#opening-a-sqlite-database-file)
  + [Одна панель, чтобы управлять ими всеми](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#one-panel-to-rule-them-all)
  + [**Просмотр данных в базе данных**](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#browsing-data-in-a-database)
  + [Но что такое база данных?](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#but-whats-a-database)
  + [Просмотр данных с помощью клиента базы данных](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#viewing-the-data-with-a-database-client)
  + [**Запрос данных**](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#querying-the-data)
  + [Привет, SQL](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#hello-sql)
  + [Сидеть, ждать, желая, с подсказкой](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#sitting-waiting-wishing-with-the-prompt)
  + [Выполнить!](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#execute)
  + [Обратите внимание на свою грамматику и синтаксис](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#mind-your-grammar-and-syntax)
  + [Сообщения об ошибках хороши](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#error-messages-are-good)
  + [**Вихревой тур по SQL**](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#a-whirlwind-tour-of-sql)
  + [Покажите мне данные](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#show-me-the-data)
  + [Выберите и покажите все строки, все столбцы](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#select-and-show-all-rows-all-columns)
  + [Выберите определенные столбцы](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#select-specific-columns)
  + [Сортировка строк по значениям столбцов](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#sorting-rows-by-column-values)
  + [Ограничение количества строк](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#limiting-the-number-of-rows)
  + [Вывод столбцов](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#deriving-columns)
  + [Имена псевдонимов](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#aliasing-column-names)
  + [фильтрация](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#filtering)
  + [Найти всех заключенных из определенного округа](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#find-all-inmates-from-a-specific-county)
  + [Скопления](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#aggregations)
  + [Получение количества строк](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#getting-a-count-of-rows)
  + [Получение подсчета по расе / полу](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#getting-a-count-by-racegender)
  + [Группировка и совокупность с любыми значениями, которые вы хотите](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#group-and-aggregate-with-whatever-values-you-want)
  + [**Портирование данных**](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#porting-data)
  + [Экспорт данных](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#exporting-data)
  + [**Другие клиенты**](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#other-clients)
  + [Поездка г-на Робота с помощью командной строки](http://2016.padjo.org/tutorials/hello-sqlite-and-sqlite-clients/#going-mr-robot-with-the-command-line)

**Быстрый старт и обучение**

Если вы новичок в SQLite и баз данных , в общем, это может быть легче учиться через *делать* . В этом разделе содержится руководство по быстрому запуску-щелчку-здесь-типа, которое *должно* работать на относительно современном компьютере и операционной системе, хотя это понятно и пользователю-новичку.

**Загрузите и запустите DB Browser для клиента SQLite**

Перейдите на домашнюю страницу [sqlitebrowser.org](http://sqlitebrowser.org/) и загрузите соответствующий исполняемый файл:

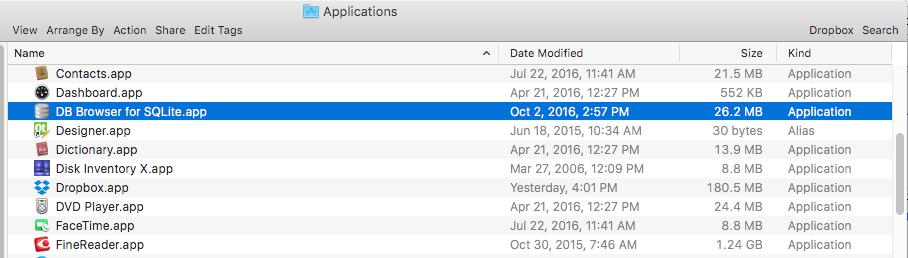


Теперь **установите** клиент, выполнив все, что вы обычно делаете, чтобы установить приложение. Это может включать переход к папке « **Загрузка** » и двойное нажатие на то, что вы только что загрузили.

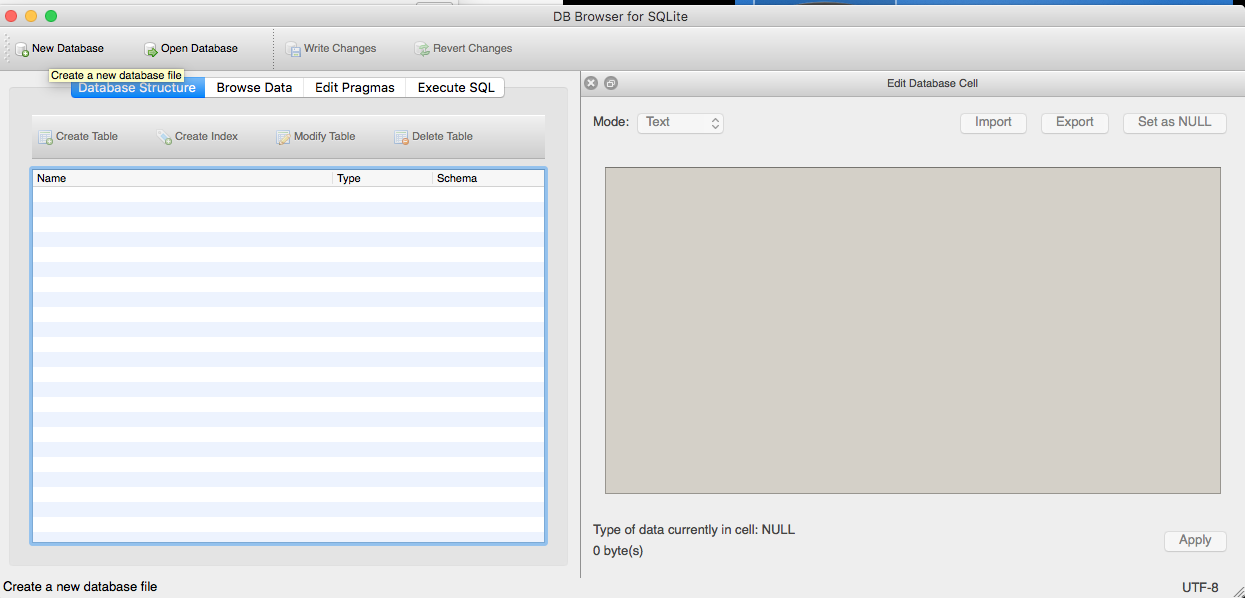
Для пользователей Mac вы получите стандартное всплывающее окно, в котором вы попросите загрузить забавный значок в папку « **Приложения** »:



Перейдите в папку «Приложения», найдите значок « **Браузер DB для SQLite»** и **дважды щелкните по**нему:



Если вам удалось установить последнюю версию, **3.9.1** , **DB Browser** будет выглядеть примерно так:



(Если вы используете версию 3.8, она должна выглядеть *аналогично,* а различия не важны)

Так же, как мы можем открыть **Microsoft Excel** без указания файла, мы можем открыть наш клиент базы данных и начать пуст. Но это не весело. Итак, **выйдите из** браузера DB и перейдите к следующему этапу загрузки файла базы данных pre-build SQLite.

**Загрузить файлы базы данных SQLite**

Я создал «стартовый пакет» баз данных SQLite, которые интересны и относительно легко изучаются: [SQLite Data Starter Packs](http://2016.padjo.org/tutorials/sqlite-data-starterpacks) .

В верхней части страницы вы можете получить краткий обзор того, что доступно. Но чтобы все было просто, давайте загрузим следующую базу данных:

* [Флорида](http://2016.padjo.org/files/data/starterpack/florida-deathrow/florida-deathrow.sqlite)

Нажав на приведенную выше ссылку, вы должны немедленно загрузить файл в папку «Загрузки», по умолчанию это имя файла:

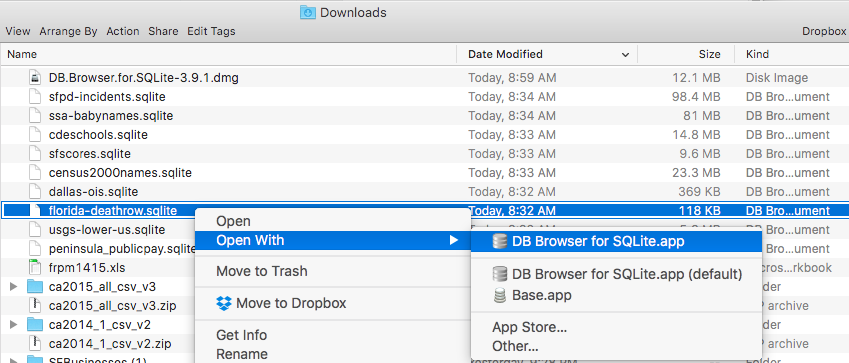
* флорида-deathrow.sqlite

**Открытие файла базы данных sqlite**

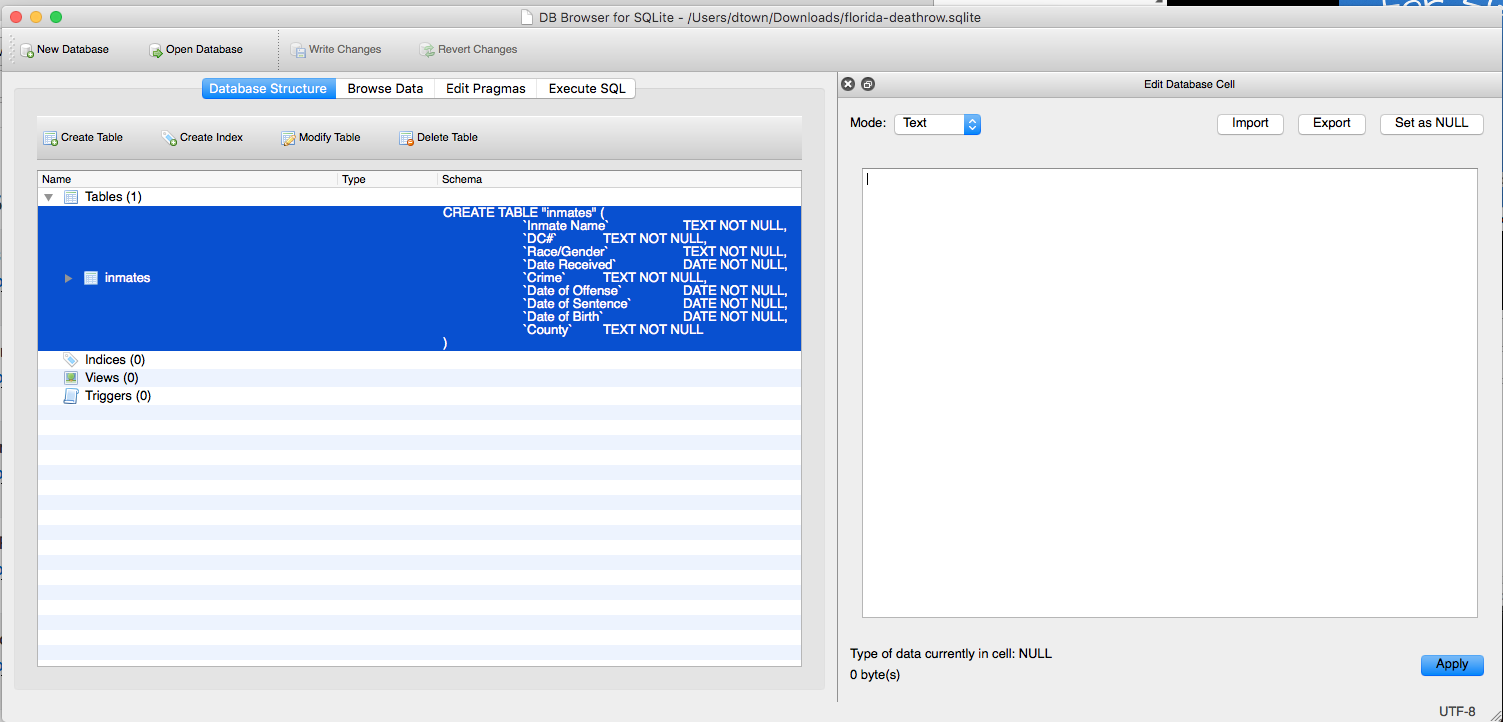
Что .sqliteозначает это расширение? Это просто читаемый человеком ярлык (как и все расширения и имена файлов), чтобы напомнить нам, что файл нельзя открыть в Microsoft Excel или текстовом редакторе.

По соглашению, когда вы установили **DB Browser для SQLite** , DB Browser должен был зарекомендовать себя в качестве программы по умолчанию для открытия .sqliteфайлов при двойном щелчке по ним. Если нет, вы можете щелкнуть правой кнопкой мыши файл и указать, какую программу использовать для открытия данного файла.

Попробуйте открыть florida-deathrow.sqliteфайл в DB Browser:

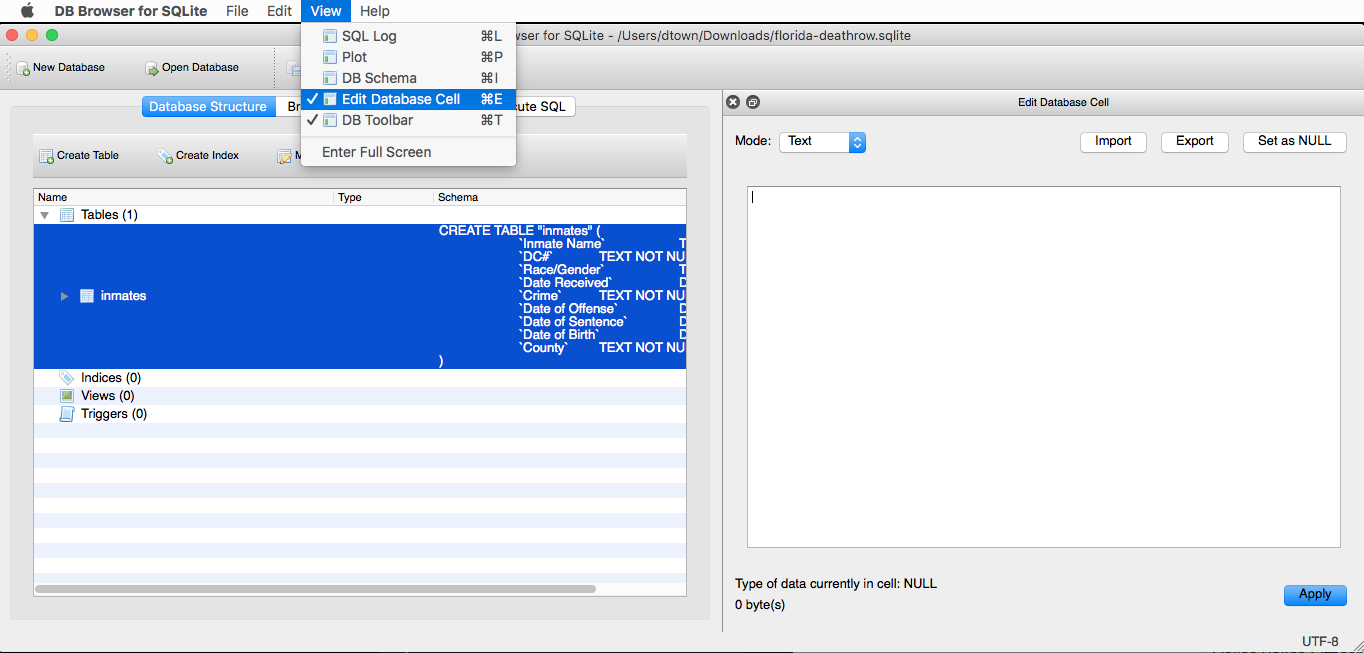


И вот как выглядит DB Browser, когда вы запустили его, указав файл базы данных, который нужно открыть:

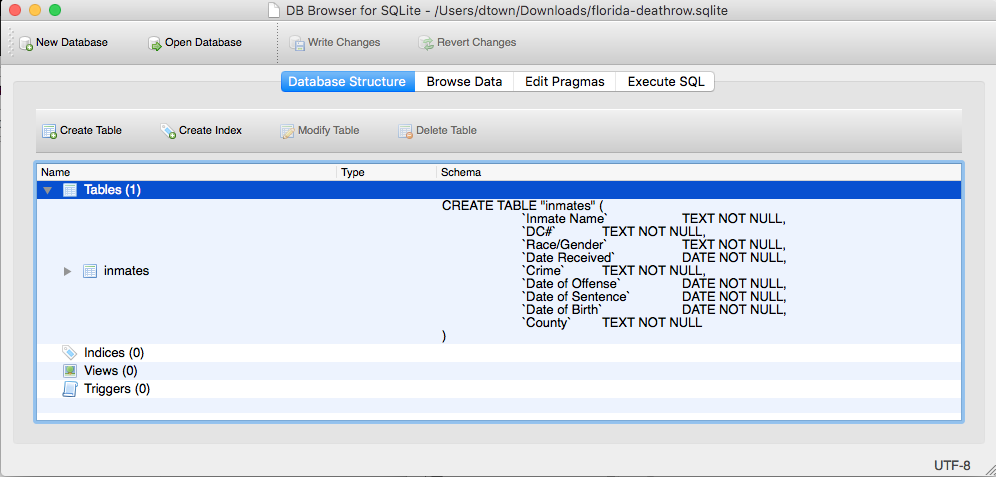


**Одна панель, чтобы управлять ими всеми**

Это просто быстрое косметическое действие, которое вы можете игнорировать или отменить позже: в большинстве программ мне нравится открывать только одно окно. По какой-либо причине браузер DB по умолчанию открывает две панели. Левая панель в порядке. Но правая панель под названием «Редактировать ячейку базы данных» не только пуста, но и не будет особенно полезна для подавляющего большинства наших SQLite-работ. Поэтому просто закройте его, нажав **X** в углу или переключив опцию в подменю **View** :



И теперь у нас только одна панель:



Гораздо чище!

**Просмотр данных в базе данных**

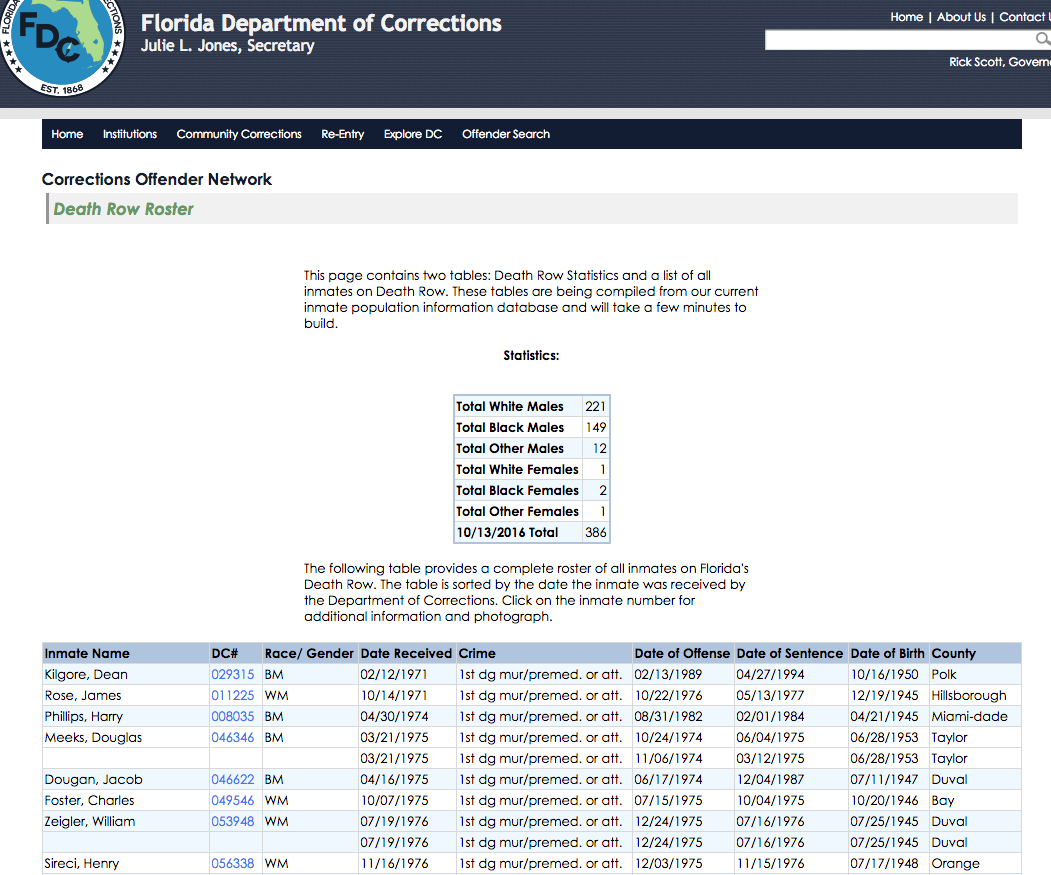
Я расскажу о функциях интерфейса, *специфичных* для **DB Browser,** в другом учебнике. На данный момент важно, чтобы мы познакомились с фундаментальными аспектами **базы данных** .

Если вы точно выполнили мои шаги, ваш клиент, т. **Е. DB Browser,** открыл базу данных [списка смертников](http://2016.padjo.org/files/data/starterpack/florida-deathrow/florida-deathrow.sqlite) в [Флориде](http://2016.padjo.org/files/data/starterpack/florida-deathrow/florida-deathrow.sqlite) и готов к ее исследованию.

**Но что такое база данных?**

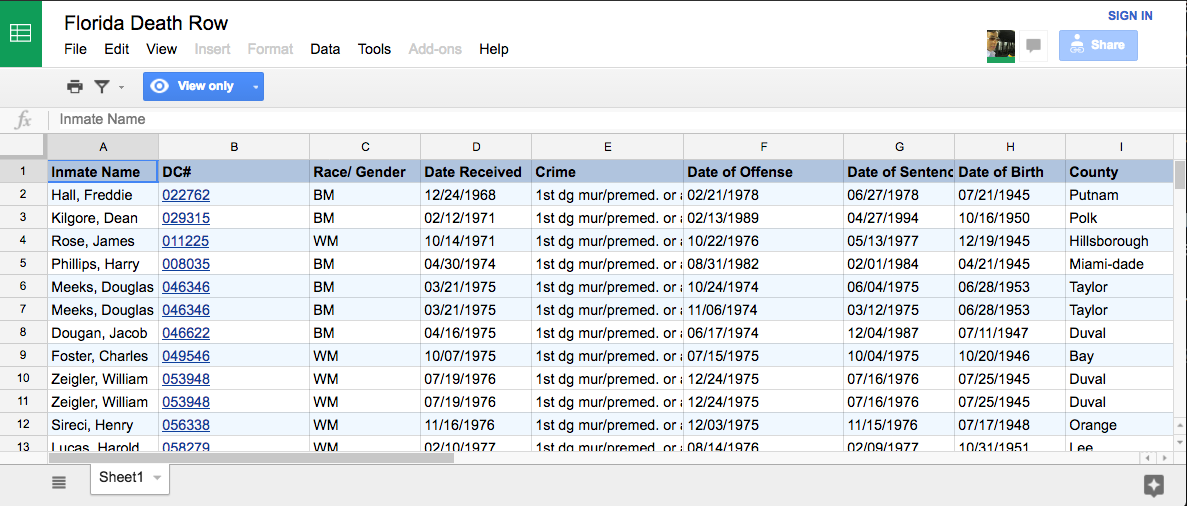
Но что *это* эта смерть строк данных? Откуда это? И что делает его «базой данных» вместо таблицы?

Данные пришли с этой [веб-страницы Департамента исправлений Флориды](http://www.dc.state.fl.us/activeinmates/deathrowroster.asp) - я написал быстрый веб-скребок, но вы можете просто предположить, что я скопировал его вручную в файл базы данных SQLite:



Но почему файл базы данных SQLite? Просто взглянув на веб-страницу, вы можете предположить, что данные таблицы смертной казни могут вписываться в стандартную электронную таблицу.

И вы были бы правы: [вот список смертных призов как Google Spreadsheet](https://docs.google.com/spreadsheets/d/12AFPLsTqILHGGL4TFbGCd_3PQ6KhDqZWtIQKaZ3D1Sg/) :



В этих данных действительно нет ничего, что требует, чтобы он находился в базе данных. Фактически, из-за своей простой структуры и небольшого размера данные этой строки смерти *должны* быть в электронной таблице, где мы можем использовать все мощные методы электронных таблиц, такие как сортировка, фильтрация, агрегация, что нам совершенно удобно.

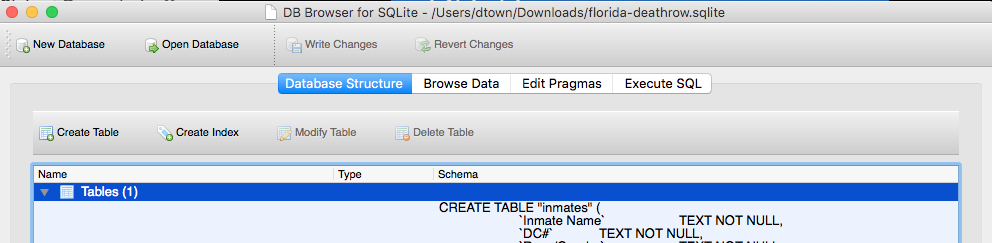
Но я импортировал эти данные в базу данных SQLite, чтобы было легче увидеть, как эти методы таблиц сопоставляются как методы базы данных. Кроме того, чтобы подтвердить, что данные *данных*являются его структурой, а не программой, с которой вы ее прочитали.

**Просмотр данных с помощью клиента базы данных**

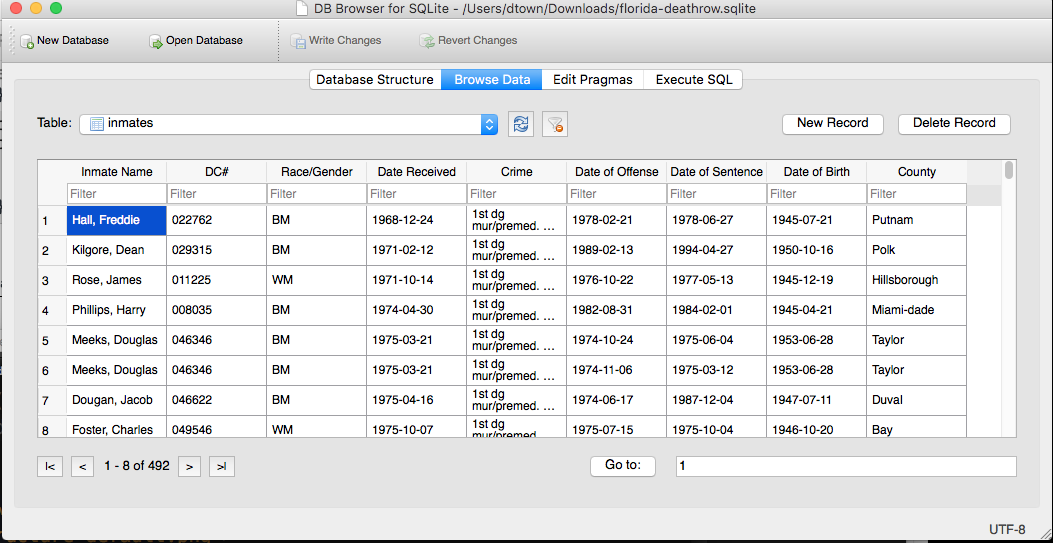
Позже в этом уроке мы увидим, что основная концепция «открытия базы данных» не дает нам права фактически *видеть* данные, как мы привыкли при открытии электронной таблицы. Но на данный момент давайте использовать **DB Browser** , который является клиентом, специально разработанным для того, чтобы сделать базы данных похожими на точки и щелчки.

Обратите внимание на панель навигации, расположенную поверх основного вида панели. В настоящее время необходимо выделить вкладку « **Структура базы данных** ».

Это представление **структуры базы данных** сейчас не интересно для нас, хотя вы, возможно, заметили, что оно, как представляется, перечисляет одну таблицу inmatesс именами, которые, по-видимому, являются именами полей, которые соответствуют [именам столбцов, которые мы видели в электронной](https://docs.google.com/spreadsheets/d/12AFPLsTqILHGGL4TFbGCd_3PQ6KhDqZWtIQKaZ3D1Sg/edit#gid=0) таблице [Google](https://docs.google.com/spreadsheets/d/12AFPLsTqILHGGL4TFbGCd_3PQ6KhDqZWtIQKaZ3D1Sg/edit" \l "gid=0) .



Без разницы. Перейдите на следующую вкладку навигации: **Обзор данных** :



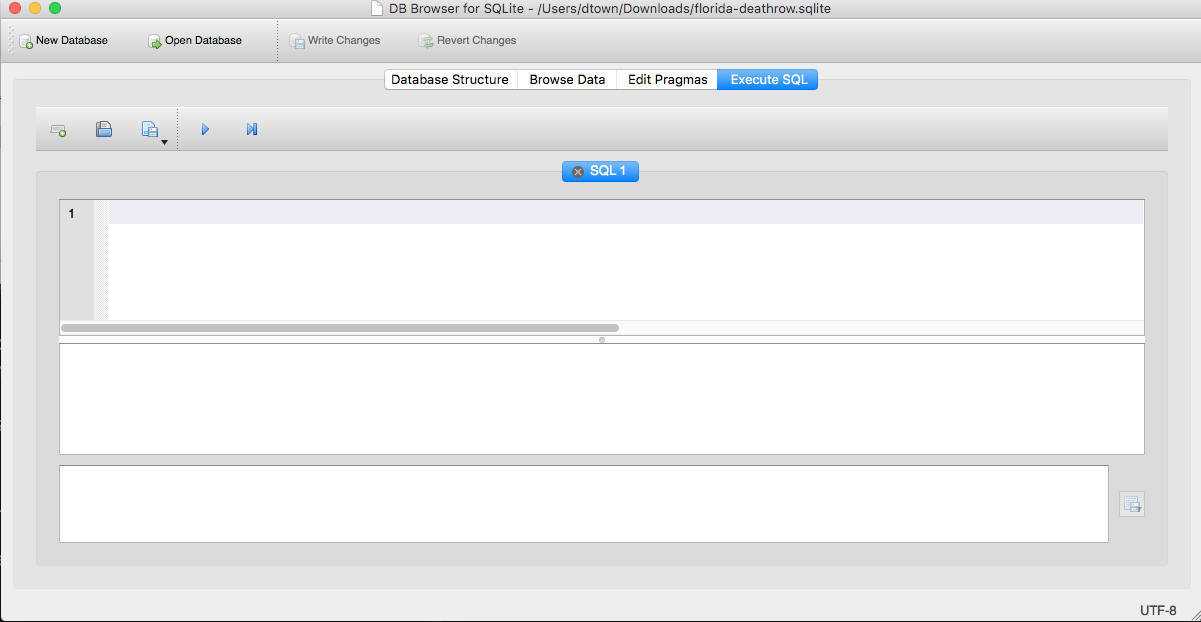
Это *похоже* на удобство просмотра в виде таблицы; мы можем прокручивать вверх и вниз список строк. У каждого столбца есть удобные **фильтры** . И если мы дважды щелкните поле, мы можем ... редактировать данные?

Если вы играете с этим видом просмотра данных более нескольких секунд, должно быть очевидно, что он *существенно* уступает тому, с чем мы привыкли, с современной электронной таблицей. Это связано с тем, что у [великих авторов с открытым исходным кодом браузера DB не](http://sqlitebrowser.org/) хватает навыков проектирования, а потому, что *базы данных не должны быть интерактивными,* как электронные таблицы.

Трудно понять эту «особенность» сейчас, пока мы не займемся более важными существенными данными. Другой способ подумать об этом: когда мы работаем с базами данных, у нас есть лучшие способы управления данными, а не вручную.

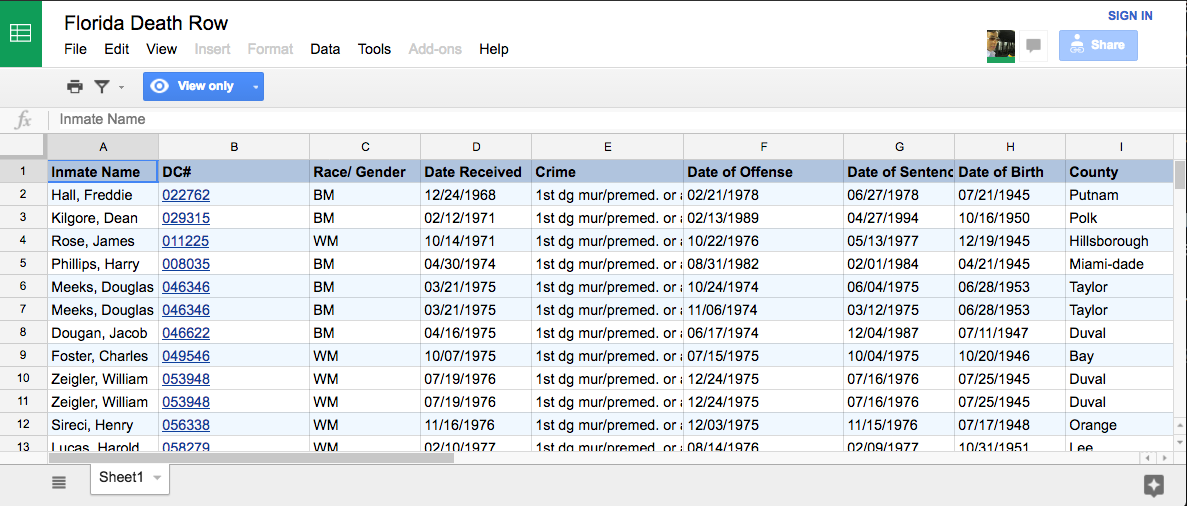
**Запрос данных**

Последняя остановка в интерфейсе браузера DB: в панели навигации щелкните вкладку « **Выполнение SQL** »:



Он не выглядит ничем, но это основной холст, на котором мы будем *писать программы,* чтобы командовать компьютером для извлечения, изучения, упорядочивания и вычисления данных из данных с гораздо большей изощренностью и масштабом, чем мы могли бы с помощью электронной таблицы.

Наличие пустого холста должно быть пугающим. Помните данные смертной казни как веб-страницу и [таблицу](https://docs.google.com/spreadsheets/d/12AFPLsTqILHGGL4TFbGCd_3PQ6KhDqZWtIQKaZ3D1Sg/) ?



По умолчанию электронная таблица показывает все, прежде чем вы сможете обратиться к другому. По умолчанию база данных ничего не показывает, пока вы не сообщите ей то, что вы хотите видеть.

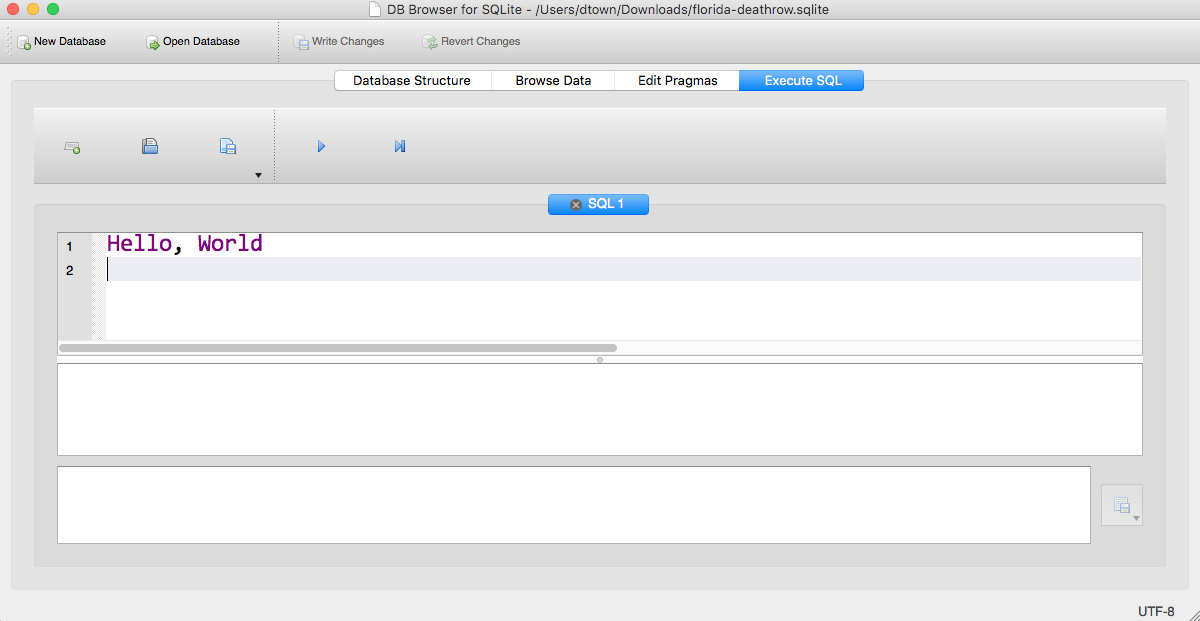
**Привет, SQL**

SQL - это язык программирования для SQLite и других баз данных в SQL-семействе. Не нужно заходить в более подробные сведения, поэтому давайте просто напишем какой-то код, начиная с [классической программы «Hello, World» Брайана Кернигана](http://blog.hackerrank.com/the-history-of-hello-world/) , правильной программы, когда мы не знаем, что мы делаем, или если что-то еще работает правильно.

Верхняя рамка в представлении **Execute SQL** представляет собой стандартное текстовое поле ввода. Тип:

Hello, World

Нажмите **Enter** .



Тогда *ждите* .

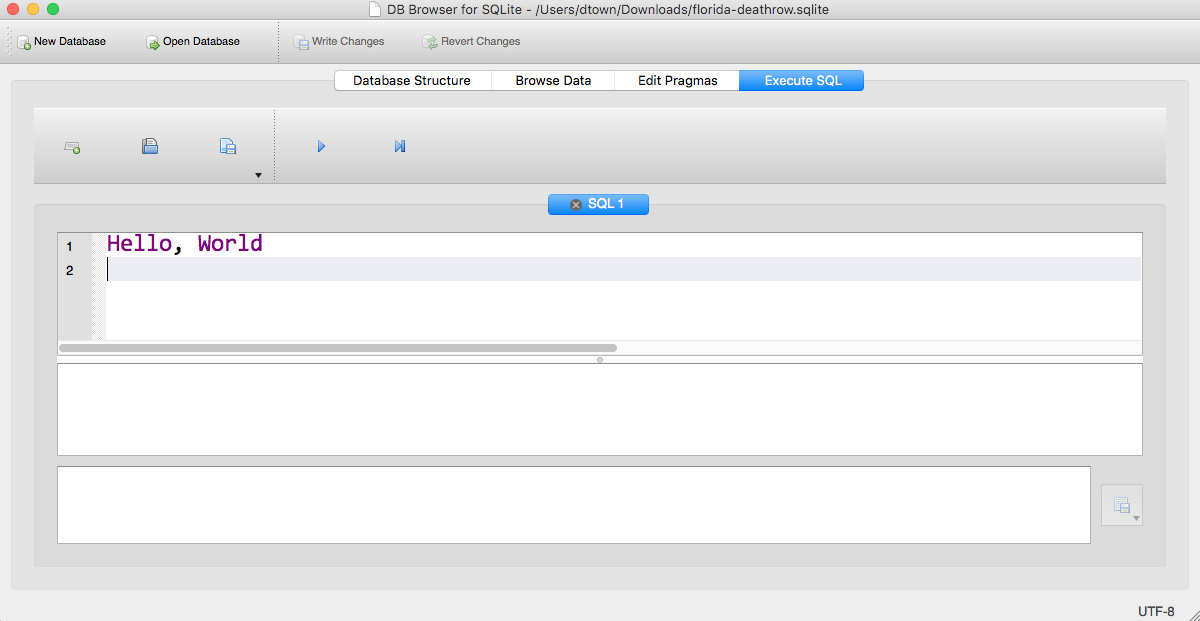
**Сидеть, ждать, желая, с подсказкой**

Если вы совершенно новичок в программировании, вы можете быть смущены тем, что браузер DB - и, действительно, механизм базы данных SQLite - кажется, *ничего не* делает, даже если вы набрали какой-то код и даже нажали **Enter** .

Это **совершенно рационально** , хотя для опытных программистов (включая меня) трудно понять, как даже начинающих можно смутить.

Это рационально, потому что, если вы используете современные компьютеры с любой регулярностью и даже профессионализмом, так как, скажем, 2010 год, тогда вы, возможно, либо забыли, никогда не осознали, либо никогда не испытывали, что компьютер просто *ждет* вас.

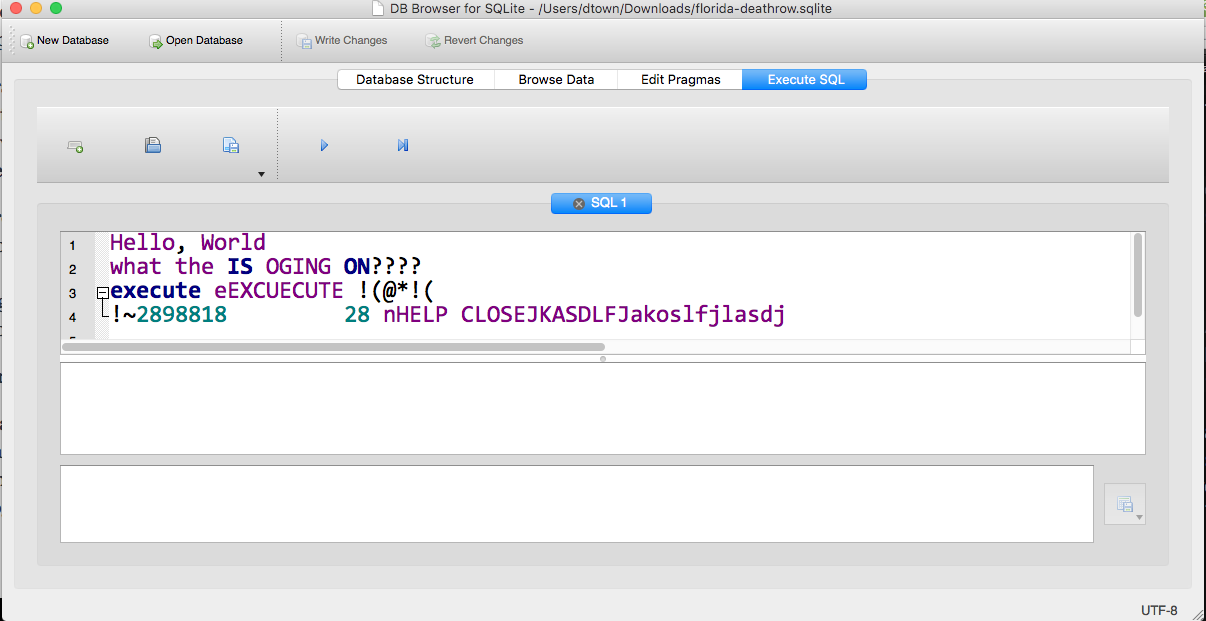
Как только вы откроете свой веб-браузер, ваш компьютер отправляет тысячи байтов данных, относящихся к тому, что вы просматривали в предыдущем сеансе, или о том, что вы ожидаете просматривать в ближайшие несколько минут. Как только вы введете пару персонажей в Google, у Google уже есть мнение о том, с чем вас путают:



[Система Facebook знает вас и ваших друзей так хорошо, что](https://www.facebook.com/help/122175507864081?helpref=uf_permalink) , как только вы загружаете фотографию, она может в основном пометить лица на фотографии даже до того, как вы увидите загруженную фотографию в своем браузере. Как только вы начнете вводить *что-* либо другу, серверы Facebook обрабатывают ваши слова, даже если вы [не отправляете сообщение](http://newsfeed.time.com/2013/12/16/facebook-is-keeping-track-of-every-post-you-write-and-dont-publish/)

Черт, вы не можете даже *загружать файл* в современную электронную таблицу без нее, программу электронных таблиц, накладывая свои мнения и бросая [сообщения об ошибках](http://superuser.com/questions/210027/why-does-excel-think-csv-files-are-sylk) или [искажая исходные данные](http://superuser.com/questions/413226/why-does-excel-treat-long-numeric-strings-as-scientific-notation-even-after-chan) .

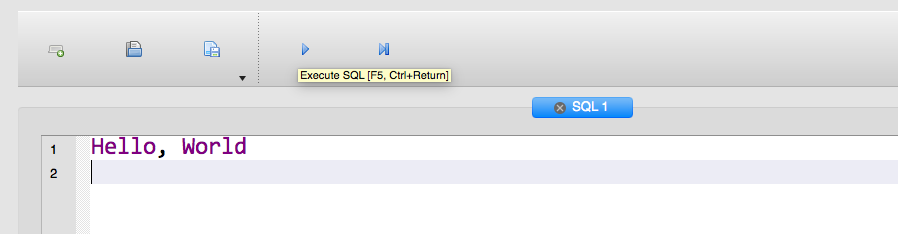
SQLite и клиент DB Browser столь же изощренны, как и любое современное программное обеспечение, которое вы используете изо дня в день. Но DB Browser, помимо создания очень узких и простых предложений автозаполнения при написании запроса, ничего *не* сделает, пока мы не **нажмем** кнопку **Execute** . Поэтому не стесняйтесь ошибаться, вас не судить ... пока:



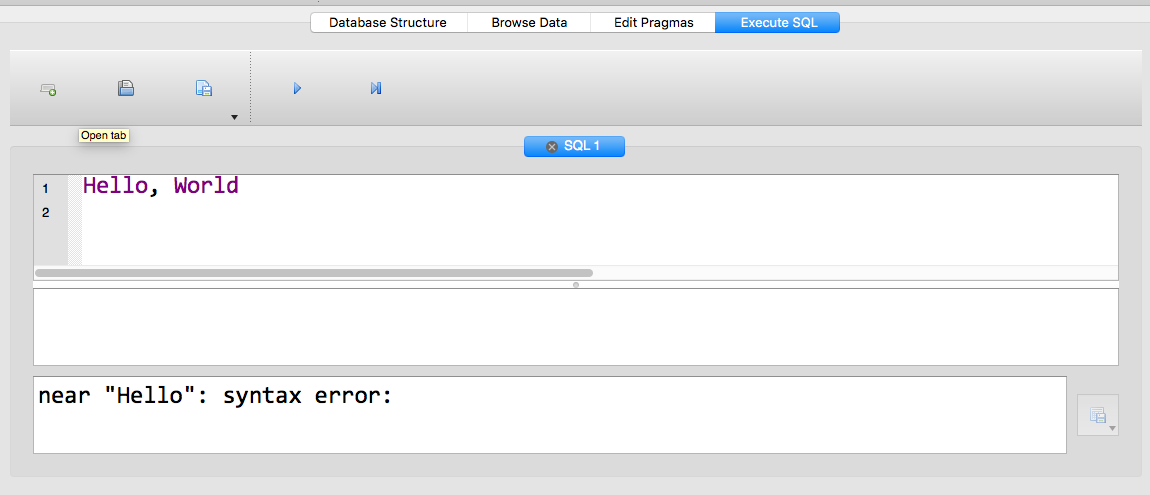
И сам движок SQLite, ну, он берет минимализм до крайности, как мы увидим позже.

**Выполнить!**

Хорошо, давайте вернемся к работе. В браузере DB кнопка, которая выглядит как кнопка воспроизведения на музыкальном проигрывателе, будет выполнять любой код, который мы написали:



Идите вперед и ударитесь ... И вот приходит суд:

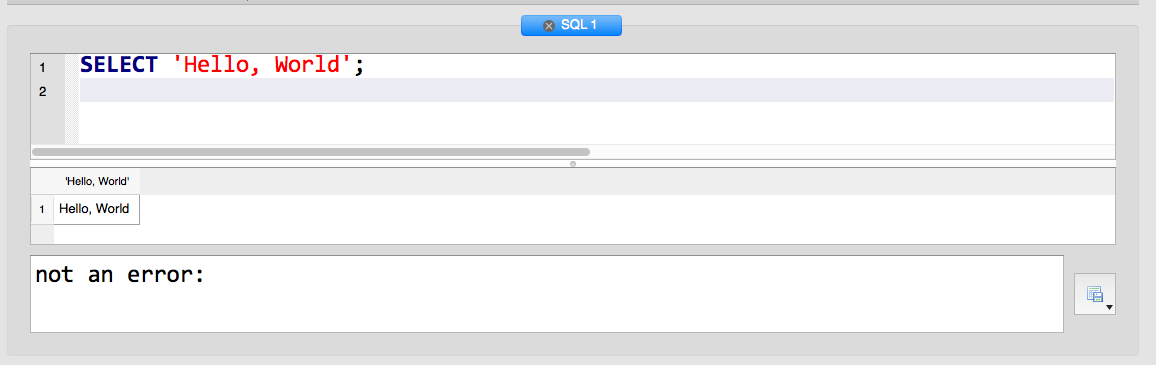


Hello, World не является правильным синтаксисом SQL. Итак, мы получаем синтаксическую ошибку. В DB Browser нижний ящик представляет собой **консоль результатов** и стоит обратить внимание на более поздние версии.

Вот один из способов сказать «Hello, World», следуя правилам SQL:

SELECT 'Hello, World';

Ура, поздравляю с первой программой SQL:



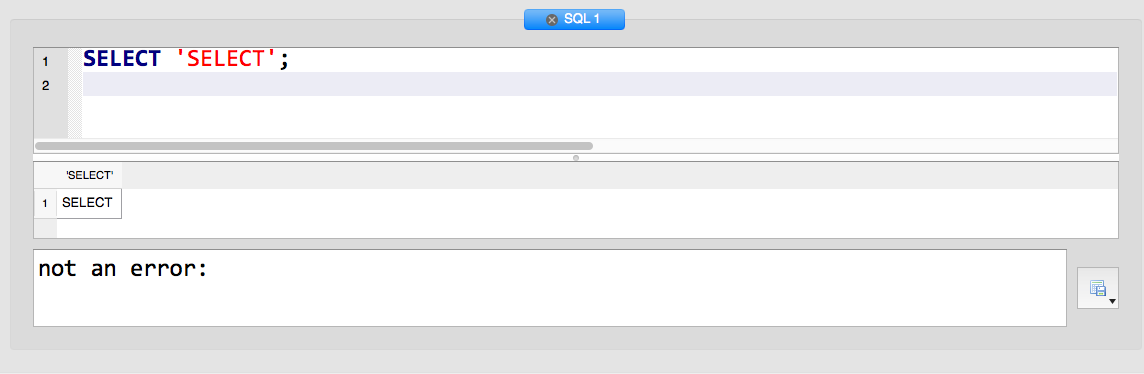
Мы подробно рассмотрим детали интерфейса DB Browser. Важное сообщение находится в консоли результатов:not an error

**Подсветка синтаксиса**

Когда вы набрали эту команду, вы можете заметить, что цвет текста изменяется. Это называется **подсветкой синтаксиса** : SQL является формальным и довольно простым языком. DB Browser знает, что SELECTв этом контексте это ключевое слово SQL, следовательно, жирный шрифт. И в SQL, когда текст заключен в одиночные кавычки, он имеет совершенно другое собрание.

Например, напишите и выполните:

SELECT 'SELECT';

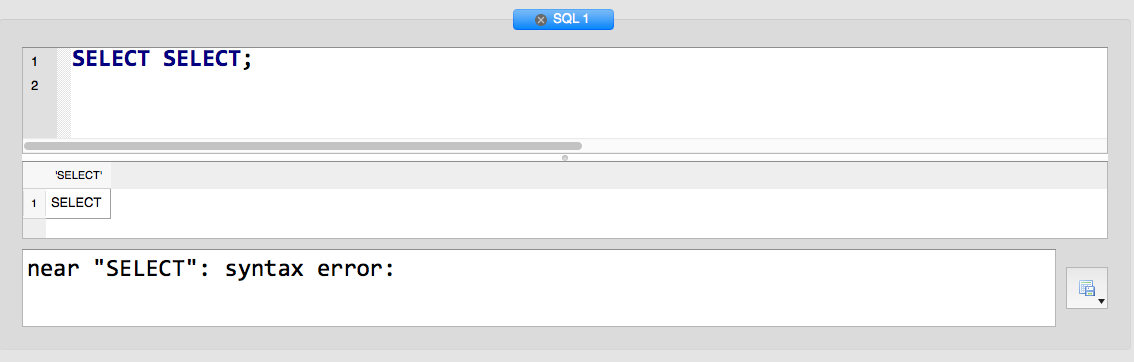


И сейчас самое время экспериментировать и преднамеренно *нарушать вещи* . На самом деле, это должен быть ваш менталитет в будущем, если вы надеетесь понять, как работают SQL и базы данных за короткое время, чтобы изучить их.

Что происходит, когда мы не цитируем эту секунду SELECT?

SELECT SELECT;

Другая синтаксическая ошибка:



**Обратите внимание на свою грамматику и синтаксис**

Опять же, мы рассмотрим это более подробно позже, но SQL - это *язык* . Грамматика и пунктуация имеют смысл так же, как и на человеческих языках, и отсутствие единого слова или знака препинания или правила может быть проблематичным:

Неправильно: «Она любит смех мужчин».

Правильно: «Она любит смех человека».

**Сообщения об ошибках хороши**

Трудно привыкнуть к тому, что вы *ошибаетесь* , и дело с компьютерной программой, базой данных или иным образом заключается в том, что он никогда не устает рассказывать вам, когда и почему вы ошибаетесь. Но это *актив* : мы *хотим* быстро потерпеть неудачу.

Спросите этого патентного тролля, [который только узнал о своей бедной пунктуации после потери миллионов долларов в иске](http://www.thisamericanlife.org/radio-archives/episode/496/transcript) .

Потому что сценарии гораздо хуже, чем *ошибочные* . Вы можете быть технически правильными, но абсолютно *ошибочными* :

Грамматически правильно: «Она любит непредумышленное убийство».

**Вихревой тур по SQL**

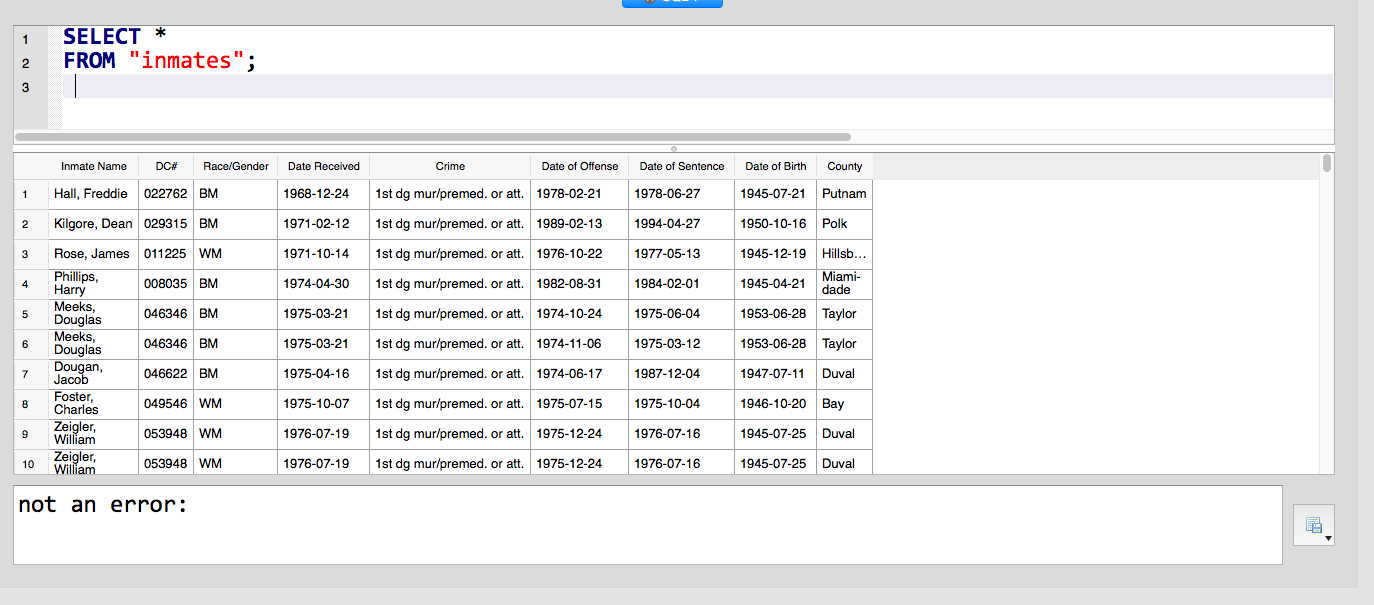
В последующих уроках мы изучим SQL, необходимые для повышения эффективности работы с данными. На данный момент я продемонстрирую, как SQL можно использовать для поиска информации, которую мы нашли с помощью электронных таблиц.

Предполагая, что вы следуете вместе с [Флоридой Death Row Roster в качестве базы данных SQLite](http://2016.padjo.org/files/data/starterpack/florida-deathrow/florida-deathrow.sqlite) , попробуйте просто ввести и выполнить тот же код, который я пишу ниже:

**Покажите мне данные**

**Выберите и покажите все строки, все столбцы**

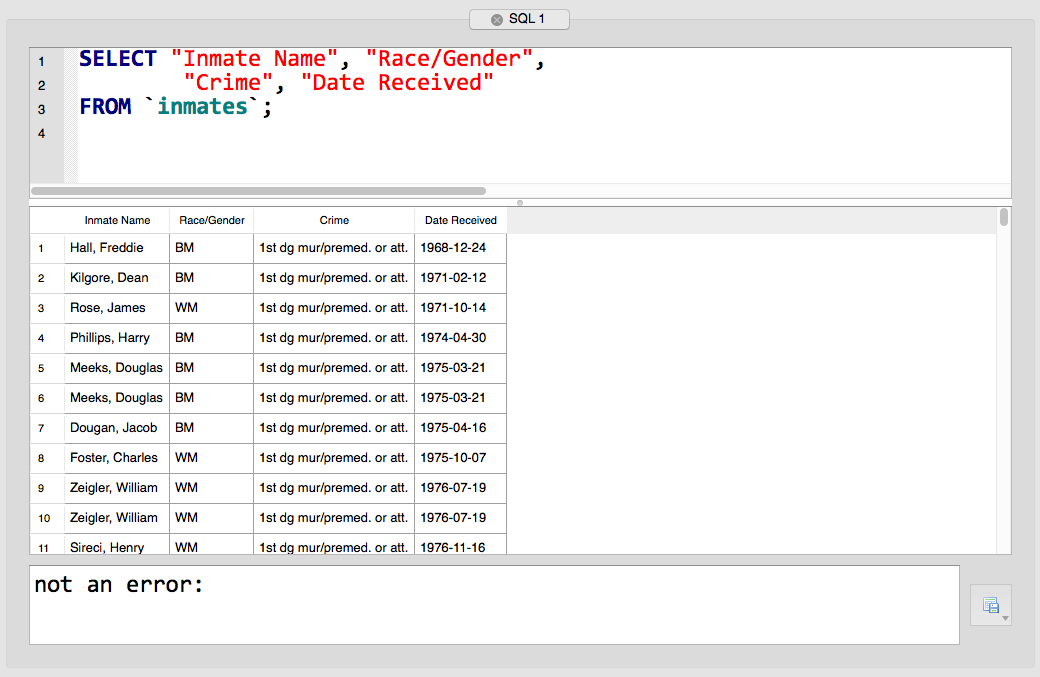
В отличие от электронных таблиц, мы не можем получить что-то, невзирая ни на что, без написания кода:

sql SELECT \* FROM "inmates";

**Выберите определенные столбцы**

Помните, как наборы данных в реальной жизни часто содержат ненужные или избыточные столбцы и как эти столбцы могут затруднить просто *чтение* электронной таблицы? И работу, которую требуется, чтобы вручную скрыть или удалить столбцы? Будучи явным в нашем SQL-запросе, мы объявляем именно то, что хотим, и только то, что хотим:

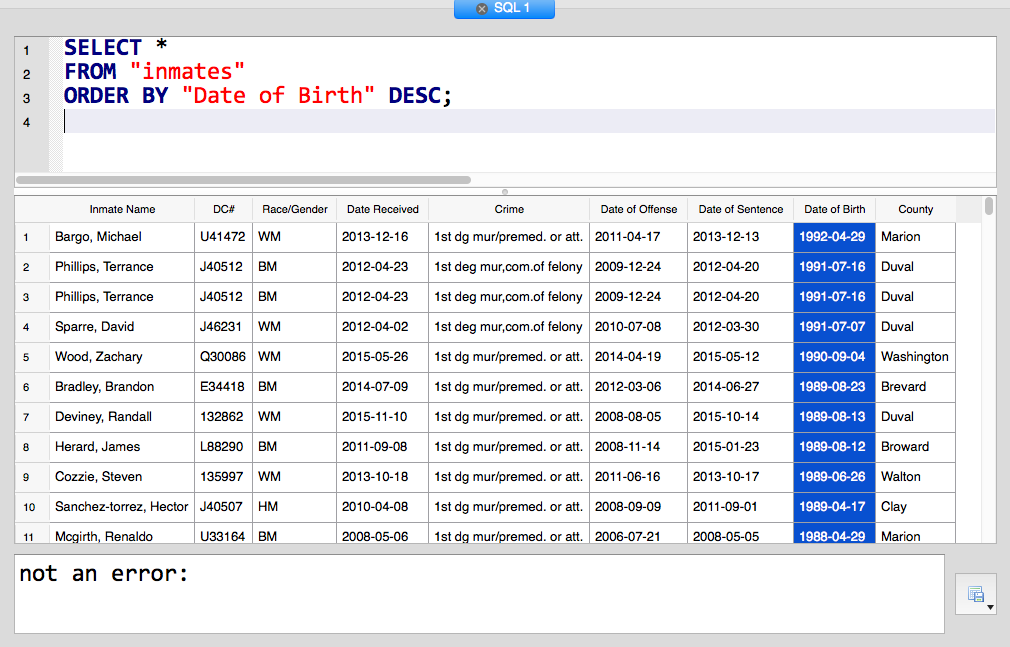
sql SELECT "Inmate Name", "Race/Gender", "Crime", "Date Received" FROM "inmates";



**Сортировка строк по значениям столбцов**

Не намного больше работы, чем щелчок заголовка столбца в электронной таблице - и *гораздо* более точный. Вот список заключенных, заказанных младшим сначала - то есть «дата рождения в порядке убывания»:

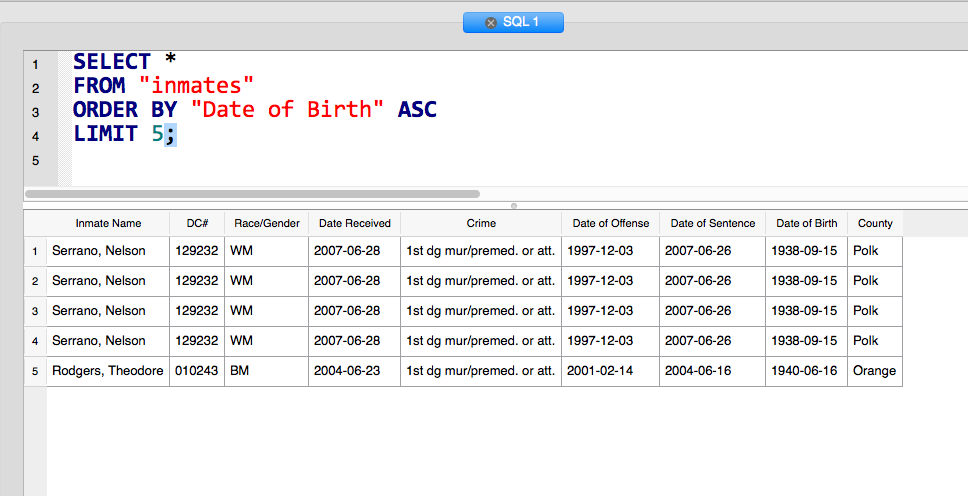
sql SELECT \* FROM "inmates" ORDER BY "Date of Birth" DESC;



**Ограничение количества строк**

После своего рода мы часто не заботимся обо всем наборе данных. Мы просто заботимся о том, что находится наверху. В электронной таблице мы должны перепрыгнуть через несколько обручей, чтобы скрыть шум. С SQL это просто другое предложение:

sql SELECT \* FROM "inmates" ORDER BY "Date of Birth" ASC LIMIT 5;



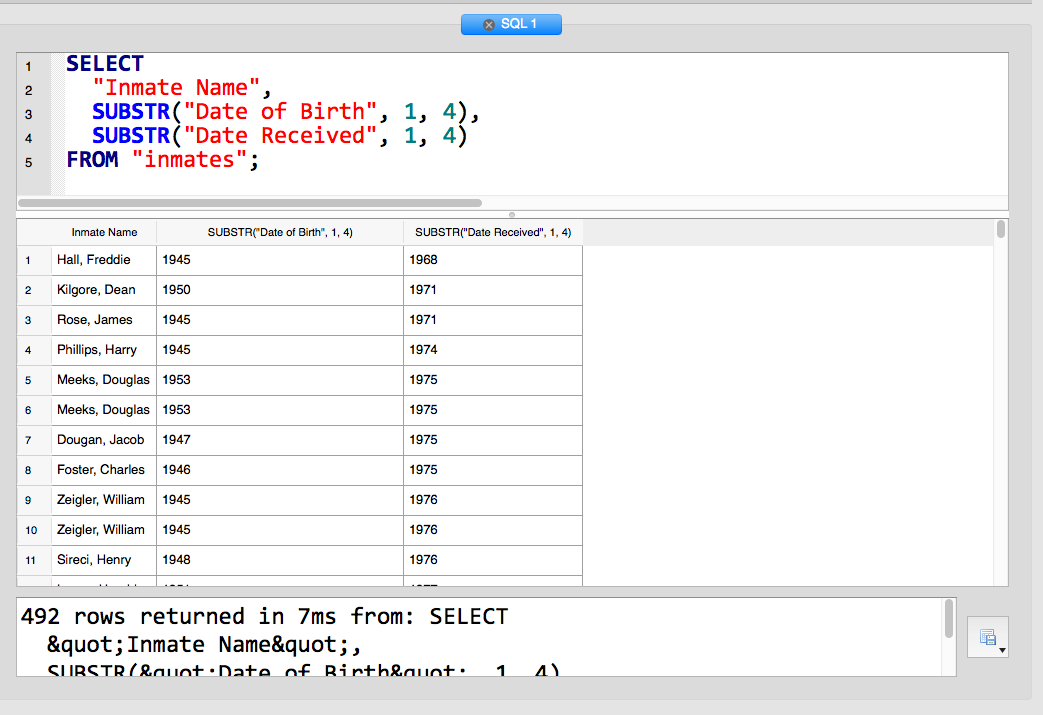
**Вывод столбцов**

Не намного отличается от создания нового столбца в электронной таблице, заполненного значениями, полученными из формулы, например:

=(B2+C2) / 100

Но в SQL нам не нужно запоминать произвольные буквенные буквы. Или щелкните мышью, чтобы вставить новый столбец и реплицировать значения. Нам даже не нужно *создавать* столбец вообще. Мы просто объявляем формулы и расчеты, которые хотим видеть:

sql SELECT "Inmate Name", SUBSTR("Date of Birth", 1, 4), SUBSTR("Date Received", 1, 4) FROM "inmates";

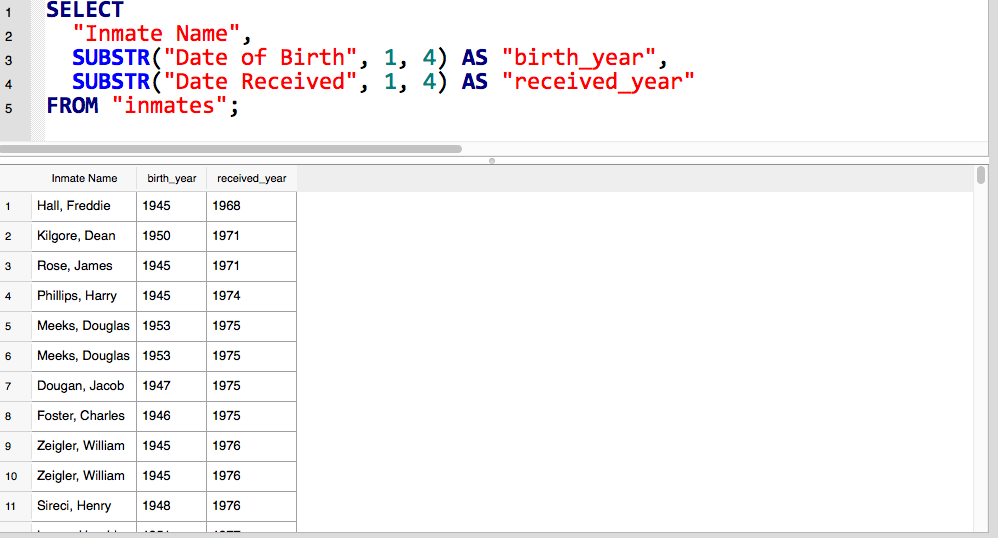


**Имена псевдонимов**

Еще один шаг, требуемый в версии электронной таблицы вставки данных: назначение новых столбцов.

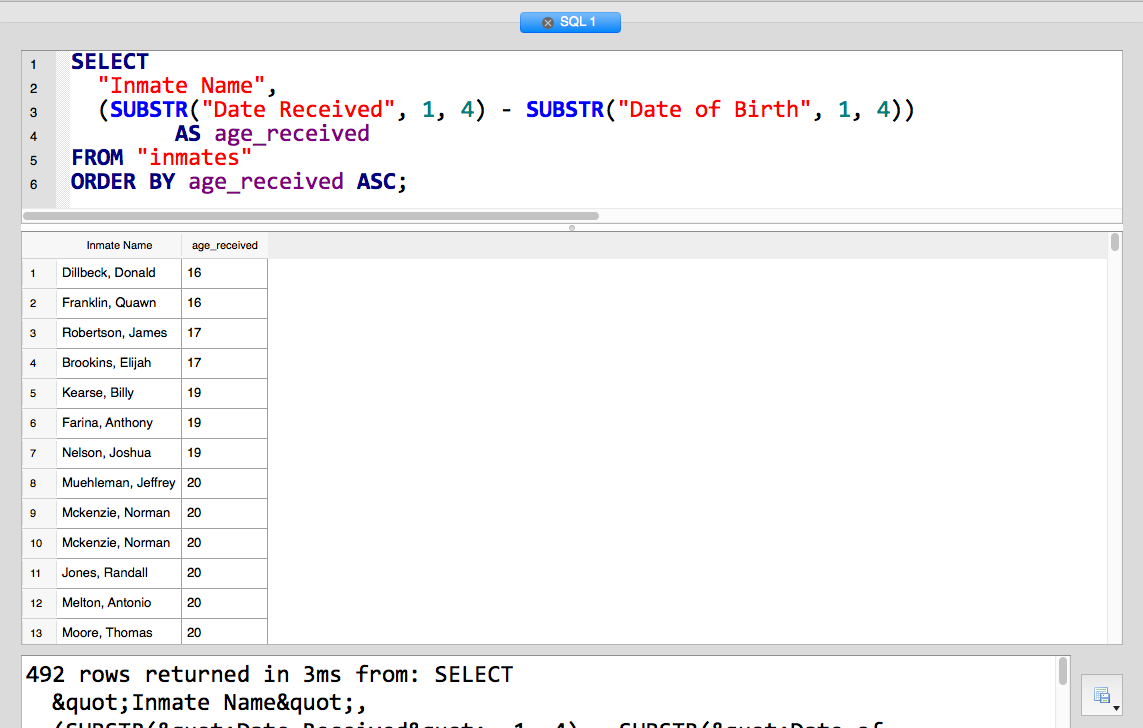
Нам не нужно делать это в SQL, как вы можете видеть на предыдущем скриншоте. Но если нам нужны более приятные, удобные для человека заголовки, нам просто нужно быть немного более откровенными:

sql SELECT "Inmate Name", SUBSTR("Date of Birth", 1, 4) AS "birth\_year", SUBSTR("Date Received", 1, 4) AS "received\_year" FROM "inmates";



И мы тоже можем сделать математику: какой самый молодой возраст, когда кто-то вошел в смертную казнь? Это функция различия между received\_yearи birth\_year:

sql SELECT "Inmate Name", (SUBSTR("Date Received", 1, 4) - SUBSTR("Date of Birth", 1, 4)) AS age\_received FROM "inmates" ORDER BY age\_received ASC;



Хороший побочный эффект всего этого наложения и перестановки столбцов: мы *никогда не изменяем базовые данные* . Напротив, практически невозможно сделать какую-либо таблицу, не изменяя, по сути, данные.

Иногда мы этого не хотим.

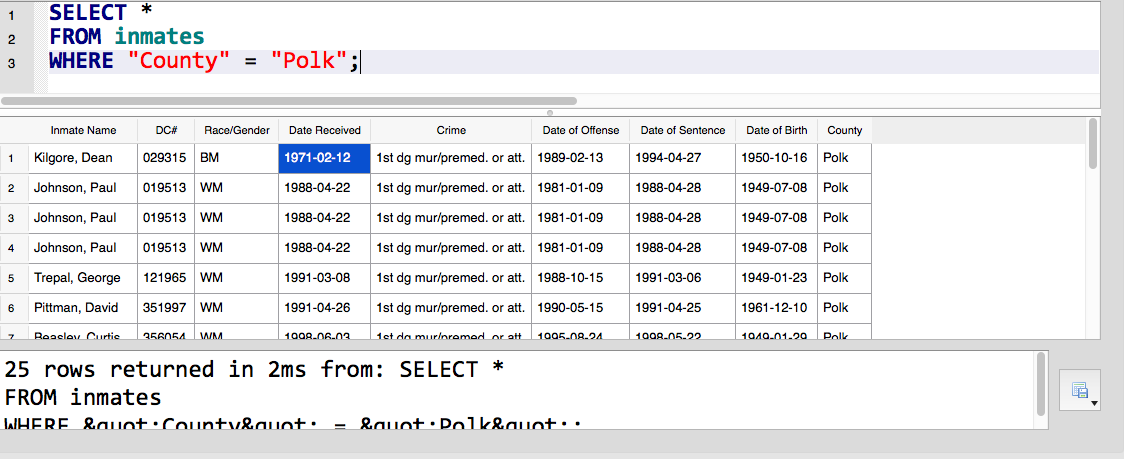
**фильтрация**

Для меня фильтрация данных в электронных таблицах не только требовала значительных вложений в работу по изучению душевных состояний в учебном меню, процесс всегда был причудливым и подвержен катастрофическим ошибкам человека. Вот почему я никогда не преподаю, как фильтровать данные, используя электронную таблицу.

Фильтрация в SQL использует одно и то же условное мышление - «покажите мне строки, в которых какой-то столбец равен / больше / меньше, чем какое-либо другое значение», - но мы можем выразить его в коде, а не в фиктивных флажках:

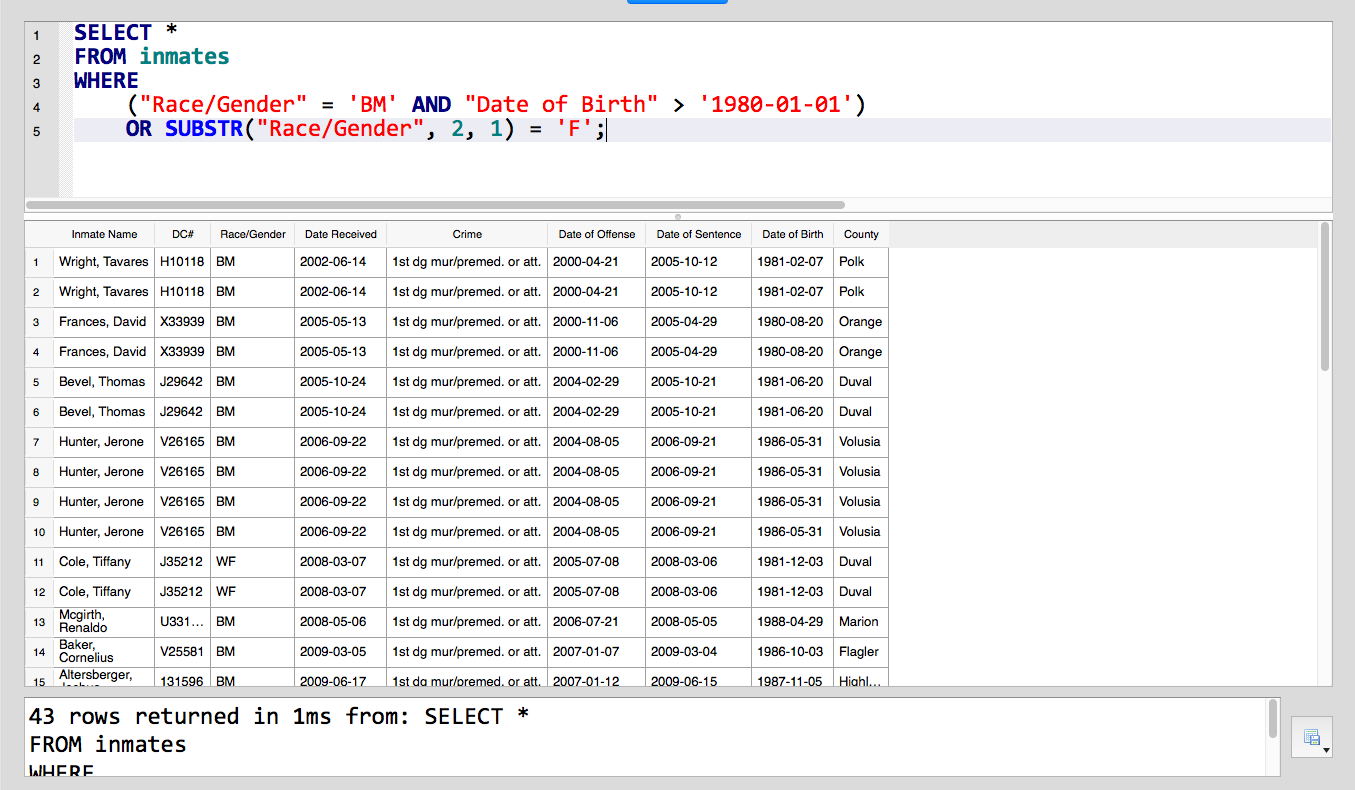
**Найти всех заключенных из определенного округа**

sql SELECT \* FROM inmates WHERE "County" = "Polk";



Условная логика может быть настолько гибкой, насколько нам это необходимо: черные мужчины, родившиеся после 1980 года, или женщины-заключенные:

sql SELECT \* FROM inmates WHERE ("Race/Gender" = 'BM' AND "Date of Birth" > '1980-01-01') OR SUBSTR("Race/Gender", 2, 1) = 'F';



**Скопления**

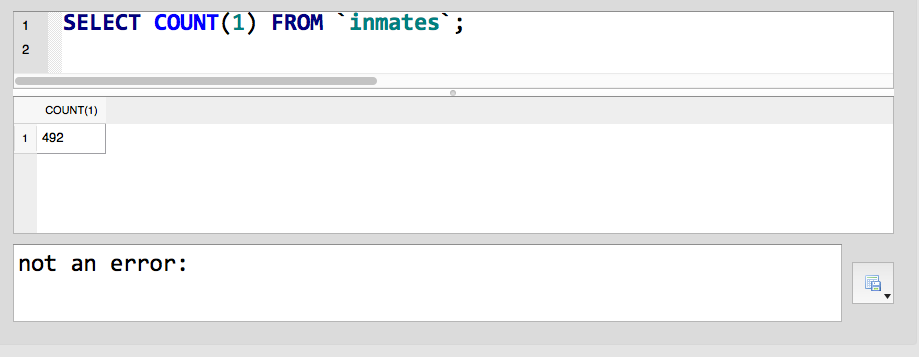
Сводные таблицы - это функция-убийца электронных таблиц, а SQL из-за жесткого характера баз данных не может выполнять все необходимые изменения данных, которые могут иметь сводная таблица.

Но с функциями агрегации и GROUP BYпредложением мы можем подсчитать, суммировать, усреднять по категориям в наборе данных.

**Получение количества строк**

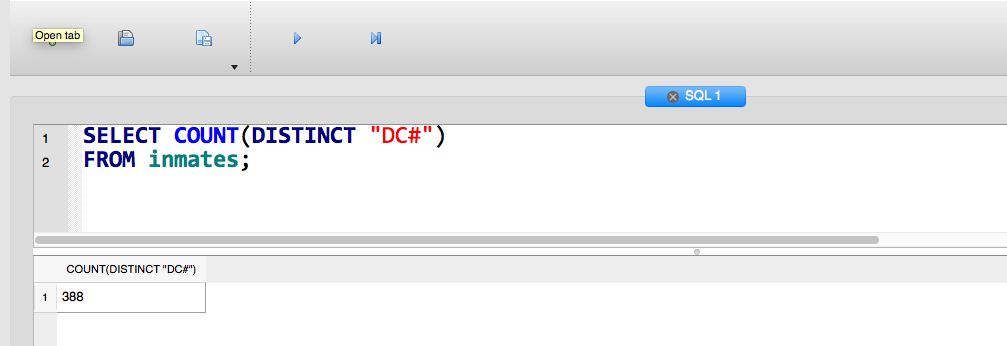
Во-первых, сколько строк в этом наборе данных?

sql SELECT COUNT(1) FROM `inmates`;



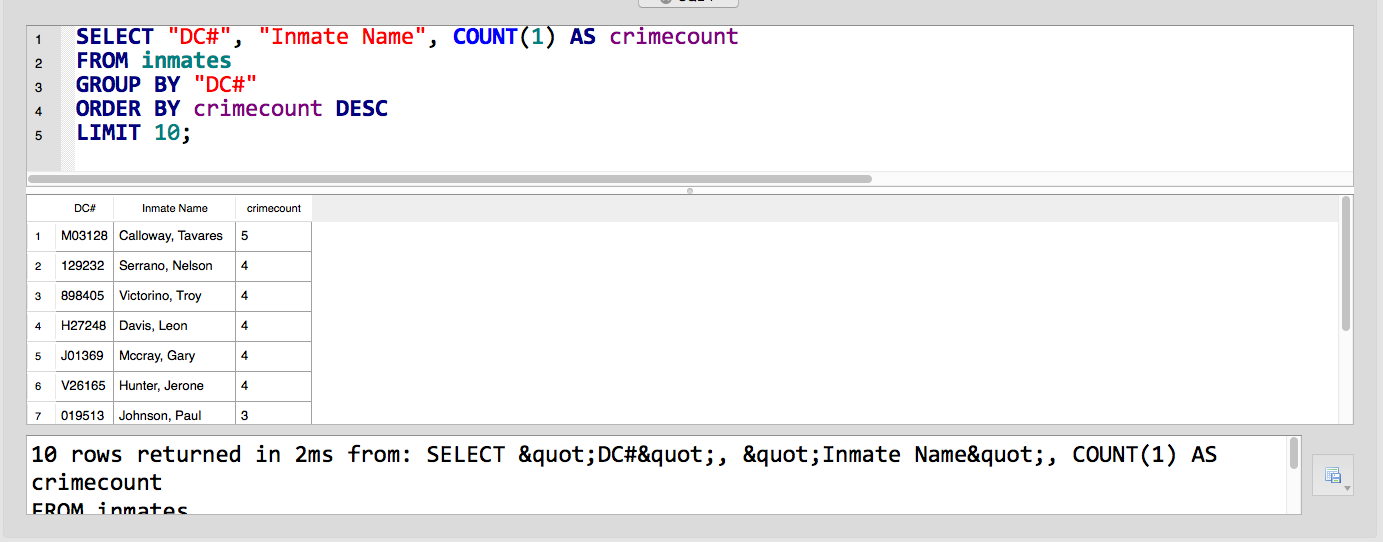
Но некоторые заключенные находятся в нескольких преступлениях. Поэтому нам нужно подсчитать *отдельные* идентификационные номера для заключенных:

sql SELECT COUNT(DISTINCT "DC#") AS totalinmates FROM inmates;



Но если мы заинтересованы в получении списка заключенных, отсортированных по количеству совершенных преступлений, мы используем GROUP BYдля граниту значение столбца:

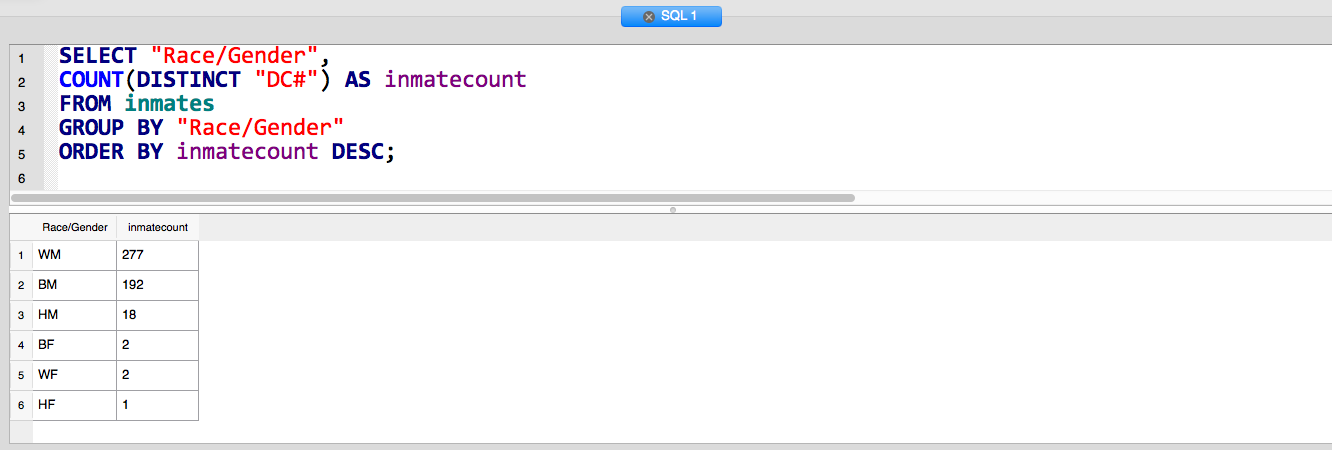
sql SELECT "DC#", "Inmate Name", COUNT(1) AS crimecount FROM inmates GROUP BY "DC#" ORDER BY crimecount DESC LIMIT 10;



**Получение счета Race/Gender**

Хотите демографический подсчет отдельных заключенных? Нет проблем:

sql SELECT "Race/Gender", COUNT(DISTINCT "DC#") AS inmatecount FROM inmates GROUP BY "Race/Gender" ORDER BY inmatecount DESC;

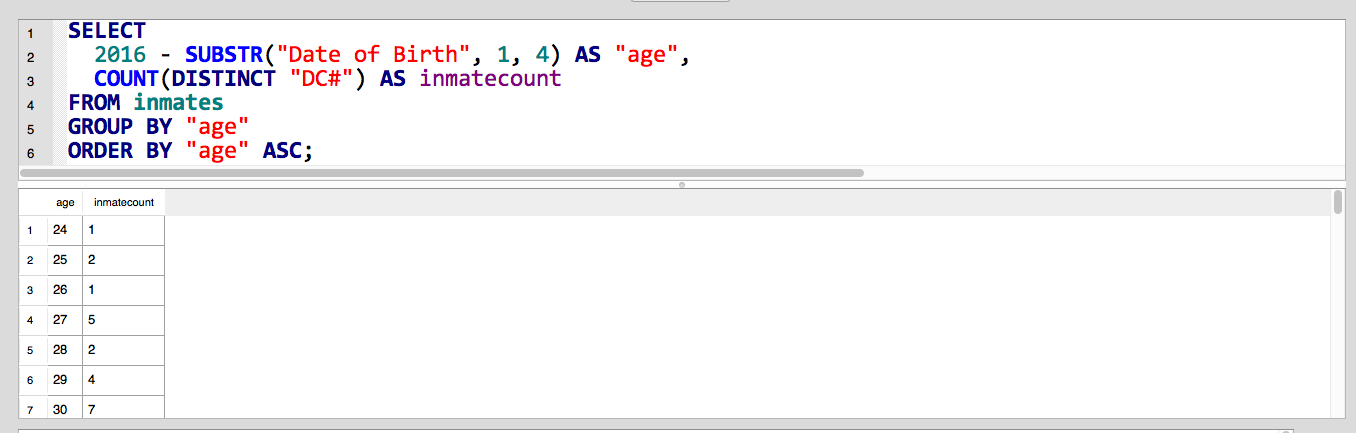


**Группировка и совокупность с любыми значениями, которые вы хотите**

(примечание: существует гораздо более строгий и разумный способ расчета дат и промежутков времени. Я игнорирую его для краткости)

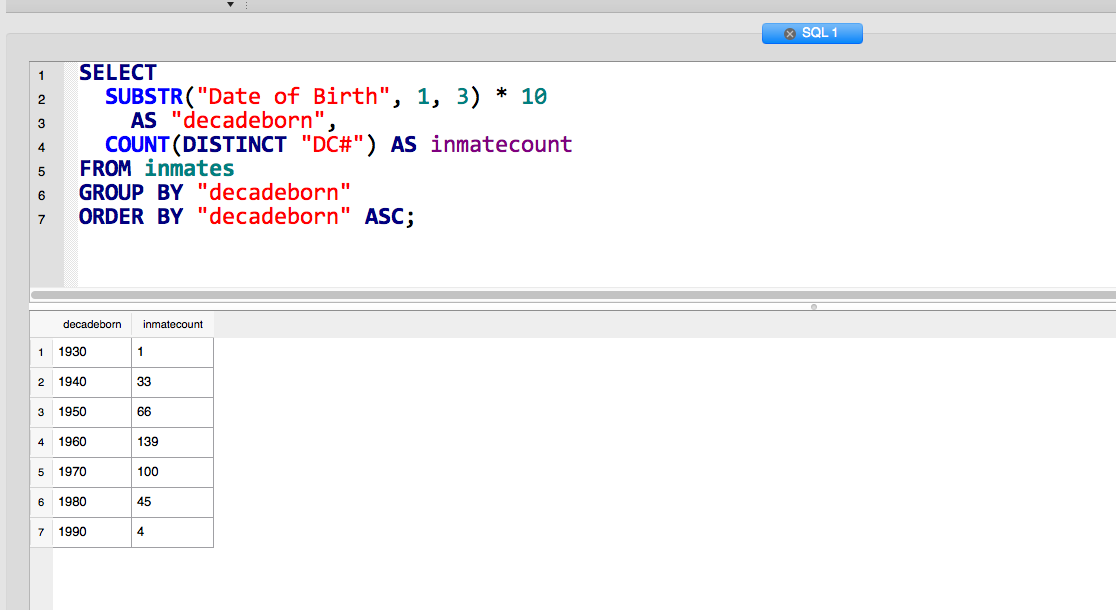
Что, если я захочу получить количество заключенных по *возрасту* ? Это может быть грубо и небрежно рассчитано с 2016 - SUBSTR("Date of Birth", 1, 4)::

sql SELECT 2016 - SUBSTR("Date of Birth", 1, 4) AS "age", COUNT(DISTINCT "DC#") AS inmatecount FROM inmates GROUP BY "age" ORDER BY "age" ASC;



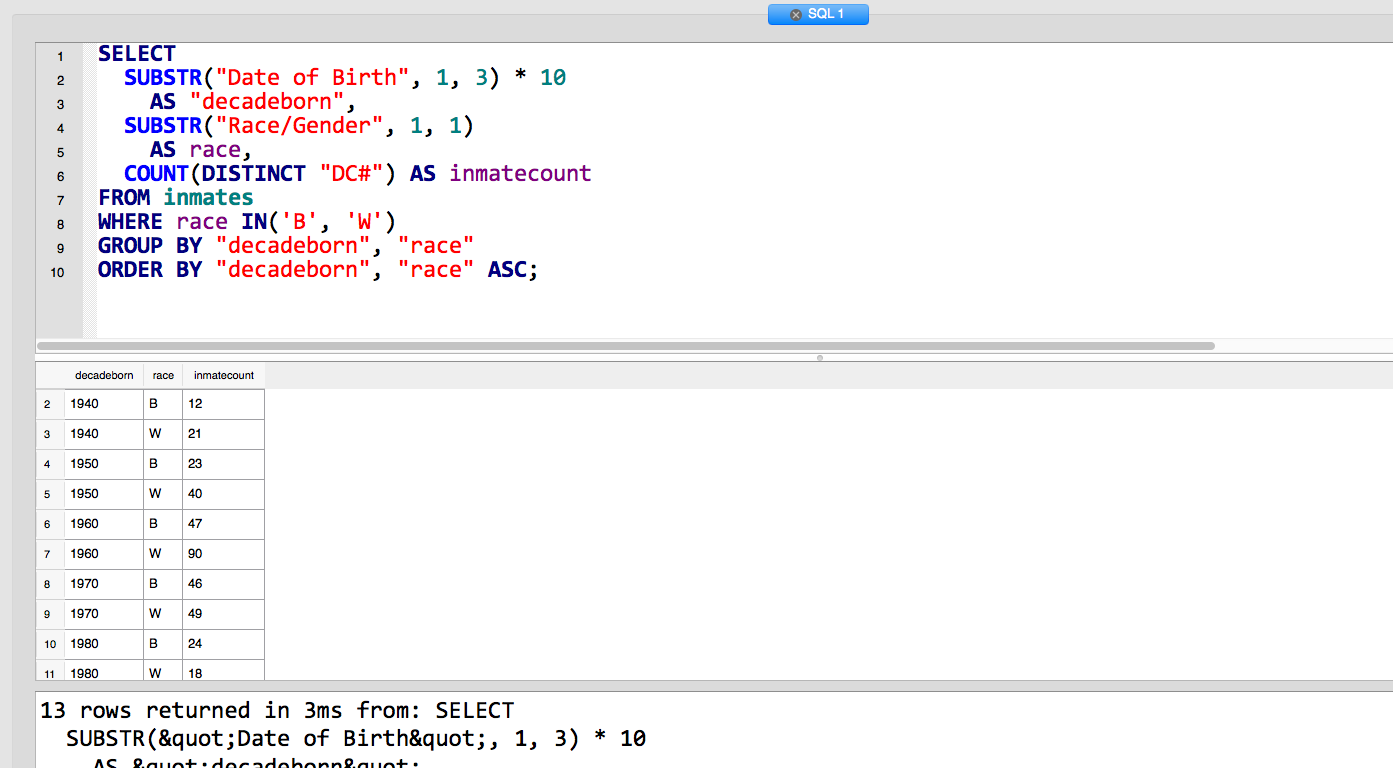
Эти возрастные группы (по каждому году) являются слишком сложными. Давайте сделаем что-то более простое: сгруппируйте заключенных к десятилетию, в котором они родились:

sql SELECT SUBSTR("Date of Birth", 1, 3) \* 10 AS "decadeborn", COUNT(DISTINCT "DC#") AS inmatecount FROM inmates GROUP BY "decadeborn" ORDER BY "decadeborn" ASC;



Еще одна совокупность: по расе и десятилетия, и давайте упростим ее на «черный» и «белый»:

sql SELECT SUBSTR("Date of Birth", 1, 3) \* 10 AS "decadeborn", SUBSTR("Race/Gender", 1, 1) AS race, COUNT(DISTINCT "DC#") AS inmatecount FROM inmates WHERE race IN('B', 'W') GROUP BY "decadeborn", "race" ORDER BY "decadeborn", "race" ASC;



**Портирование данных**

Я думаю, что для этого достаточно случайных SQL-запросов. Давайте вернемся к другим частям работы с данными, особенно *визуализации* .

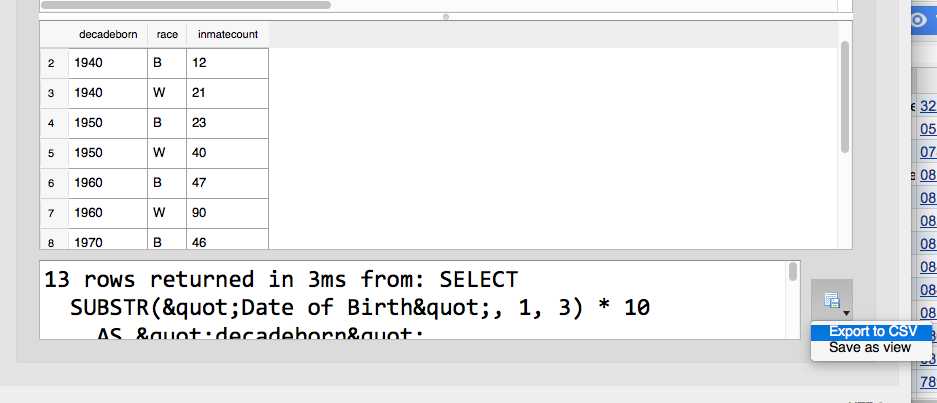
Базы данных обычно сосредоточены на хранении и сортировке данных. Именно поэтому они могут открывать и собирать наборы данных в тысячи раз больше и быстрее, чем могут делать электронные таблицы. Компромисс в том, что они не хороши для других. В отличие от таблиц, типичная база данных не поставляется со встроенным производителем диаграмм.

Нет проблем. Таблицы - отличный инструмент визуализации. Все, что нам нужно сделать, это экспортировать наши данные из базы данных и импортировать их в электронную таблицу.

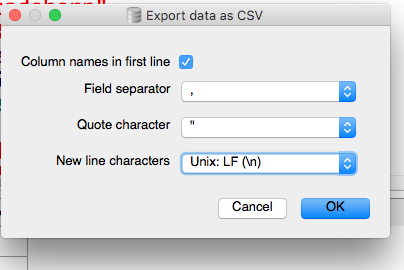
Мне понравилось самое последнее скопление, которое мы сделали из заключенных из Флориды. Результат был похож на таблицу, т.е. таблицу? Это потому, что для всех целей и целей это *еще* одна таблица. В отличие от электронной таблицы, нам не нужно создавать новую вкладку / таблицу / лист для хранения наших результатов.

**Экспорт данных**

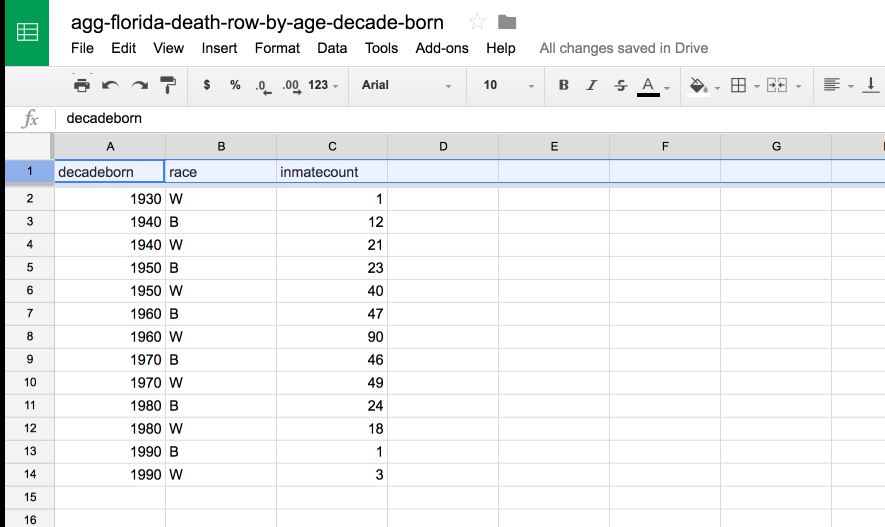
ОК, еще один конкретный советник для БД-браузера. В представлении **Execute SQL** с показанными результатами есть маленькая кнопка в правом нижнем углу, которая позволяет нам **экспортировать в CSV** :



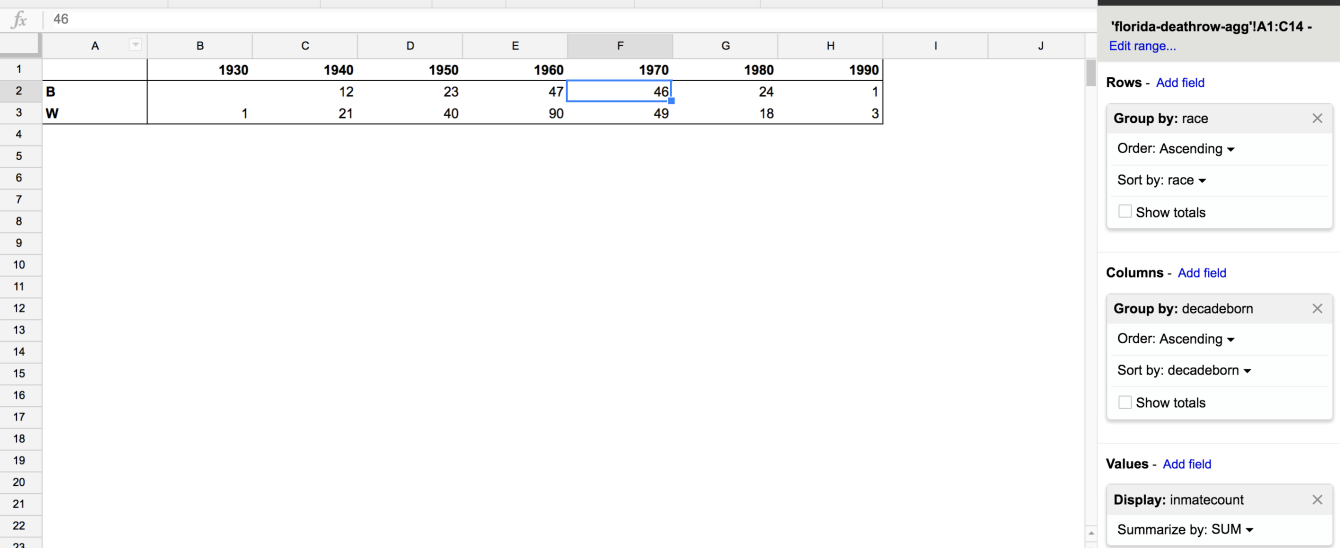
Мы будем перенаправлены во всплывающее меню: параметры по умолчанию прекрасны, хотя для «Unix» следует установить «Новые строчные символы», чтобы быть в безопасности:



Как выглядит наш новый CSV-файл, когда мы импортируем его в новую таблицу? [Точно так же, как вы ожидали](https://docs.google.com/spreadsheets/d/17ST4BCtriPPnGZHzTR6s4F-LpC6NYIAUSZbl6iQjVgQ/edit#gid=391272853) :

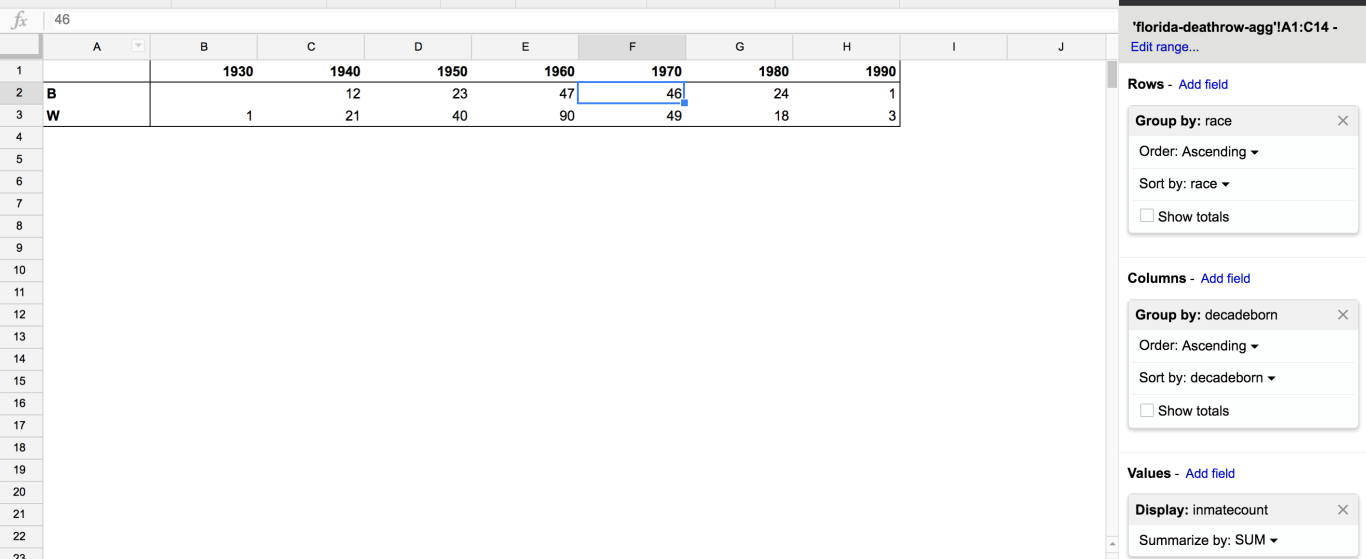


Теперь мы делаем [сводную таблицу сводных данных, чтобы переупорядочить данные специально для того, как это хотят инструменты для работы с таблицами](https://docs.google.com/spreadsheets/d/17ST4BCtriPPnGZHzTR6s4F-LpC6NYIAUSZbl6iQjVgQ/edit#gid=1907468360) :

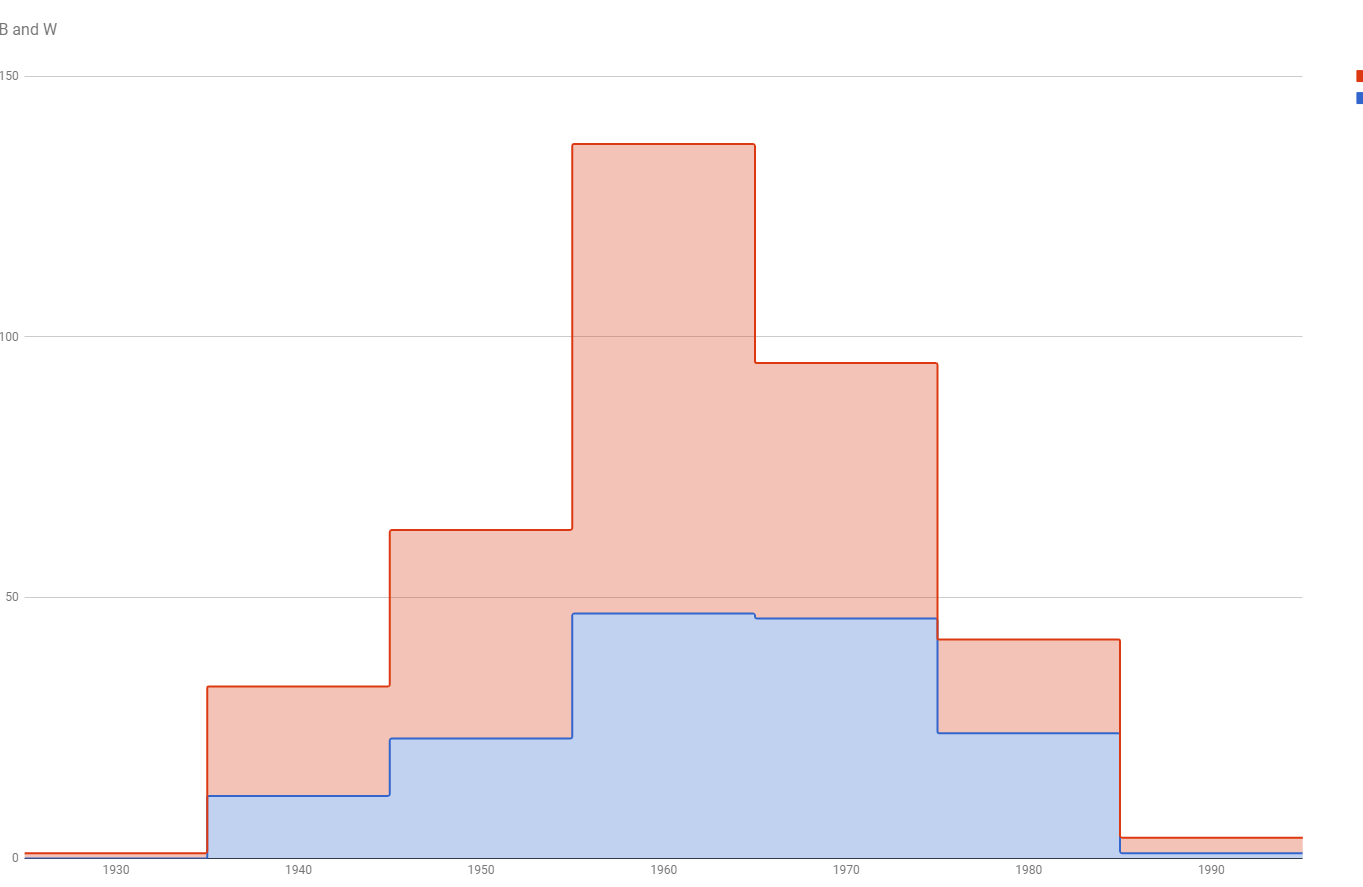


Обратите внимание, что эта визуальная перестановка данных - это одна вещь, которая является почти непреодолимой сложностью в SQL, поэтому многое, что делает анализ в SQL и переносит его в электронную таблицу, - это то, к чему вы привыкнете.

Наконец, мы визуализируем, используя мастер диаграммы Google Spreadsheet:



Вот диаграмма в ее [интерактивной славе](https://docs.google.com/spreadsheets/d/17ST4BCtriPPnGZHzTR6s4F-LpC6NYIAUSZbl6iQjVgQ/edit#gid=1721577448) :



Ничего впечатляющего, но это всего лишь быстрая демонстрация. Все, что мы ведем здесь, применимо к каждому набору данных, который мы помещаем в базу данных, и мы даже не попали в интересные и мощные части SQL.