СУБД — Lurkmore

Народ требует ~~хлеба и~~ зрелищ!

Народ требует иллюстраций к статье!
В конце концов, если бы мы хотели почитать, мы бы пошли в библиотеку.

СУБД — система управления базами данных. СУБД позволяет сосредоточиться на работе с данными, абстрагировавшись от их физического размещения, а также берет на себя заботу эффективного их сохранения и выборки. Хлеб насущный админов баз данных, предмет бешеного обожания аналитегов, от которых начальство требует нового отчёта вотпрямщаз, и бешеной ненависти быдлокодеров, которым эта НЁХ ломает такую стройненькую и уютненькую процедурку.

С появлением пхп, джабб и перлов СУБД анально оккупировали эти ваши энторнэты, так что современный говнодизайнер при лепке сайта пользуется ими, не приходя в сознание.

Суть

Весь смысл использования базы данных в том, что когда данных становится больше, чем несколько книг в этом вашем экселе и эти данные внезапно становится невероятно сложно структурировать, а попытки получить полный список приводят среднестатистический компьютер в состояние синего экрана от нагрузки, то в дело вступают СУБД, которые берут на себя это нелегкое и весьма оплачиваемое со стороны крупных предприятий бремя.

Короче, СУБД освобождает разработчика от задач хранения, модификации и поиска данных. Его дело — указать, какие данные взять, и что с ними сделать. Все остальное сделает сама СУБД.

SQL

Устав от написания бесконечных процедур поиска, вставки, удаления, замены, программисты голубого гиганта сообразили переложить эту обязанность на саму СУБД. Для этого они запилили язык структурированных запросов (SQL). Итог очевиден — вместо тысячи строк кода пишется одна строка, которая сделает ровно то, о чем ее просили, а не то, что породил больной разум говнокодера. Плюсом — все изменения в базе данных выполняются одинаково, что позволяет при правильном администрировании легко и быстро восстановить исходное состояние при сбоях и ошибках.

Теоретически, SQL должен обеспечивать и независимость от платформы, возвращая один и тот же результат для любой СУБД, реализующей её. На практике каждый производитель СУБД добавляет в SQL свои, присущие только этой СУБД фишечки, чем усложняет работу программиста и разработчика СУБД, но это, конечно же, не мешает пэхапэ-быдлокодерам использовать ORM'ы, которые представляют из себя некую прослойку между грязным SQL и абстракциями высокого уровня, которая генерирует SQL вместо программиста и позволяет забить на особенности СУБД. Но пэхапэшники используют её, потому что не знают SQL, да и короче написать

```
$some_data = SomeData::find(20);
```

чем

```
$sql = "SELECT id
FROM Some_db.Some_data
WHERE id=20";
$result = $db->query($sql);
```

При этом с возрастанием сложности запроса количество SQL кода доходит до нескольких сотен строк, в то время как в ORM максимум 2-3. Но, конечно, вполне нормальное явление, что на чистом SQL запрос будет выполняться 15 секунд, а на ORM 30 минут. При этом сам по себе SQL имеет громоздкий синтаксис, но писать простые запросы можно, просто немного зная английский язык.

Разновидности

Существует туева хуча разных СУБД, которые могут отличаться настолько, что не будут понимать SQL друг друга. Или вообще, не являться реляционными и, соответственно, не поддерживать SQL вовсе. Ниже приведены наиболее известные реляционные СУБД.

- **Oracle**

Удостоился ажно отдельной статьи здесь. Огромная сложная махина, для работы с которой требуются админы 80 уровня с сертификатами, подготовка коих стоит с десяток килобаксов. Умеет делать все,

вплоть до того, что может выступать движком для веб приложений (APEX называется). Работает на Unix, винде и еще черт знает чем. Стоит ебических деньжищ, так что, ее могут себе позволить только большие конторы, что у нас, что на Западе. ИЧСХ, если ты не собираешься на ней крутить производственную базу, а просто разрабатываешь приложения, изучаешь или играешься, то скачать и пользоваться ей можно абсолютно бесплатно, даже самой ентерпрайзной версией, что есть таки вин.

- **MS SQL Server**

Microsoft в свое время, в лучших своих традициях, подсуетилась и скопипиздила СУБД у Sybase. С тех пор, как они заявляют, ее полностью переписали. Как и всегда, MS клала и кладет прибор на стандарты и идет своим уникальным путем. Хотя, многим нравится. СУБД получилась уровнем пониже Oracle, хотя, по этому поводу идут свои холивары. Для управления БД используется почти исключительно UI, что несколько роднит эту систему с Delphi: вместо знания DDL тут нужно знать куда тыкнуть мышкой. Ну, есть еще PowerShell, но он никому не всрался. Раньше работал исключительно под Виндой, но начиная с 2016-й версии подружился с Linux. ИЧСХ, по простоте установки Linux версия на десять порядков проще и быстрее виндовой. Могут, блджад, когда захотят, сукины дети.

- **PostgreSQL**

Этот ваш open source во всей своей силе. Была разработана в университете Беркли, откуда еще много чего повыходило. Древняя, как говно мамонта — Ingres, от которого произошел PostgreSQL (собственно, **Post** In**gres**) начинал разрабатываться еще в начале 70-ых, когда этого вашего Oracle в проекте не было. Справедливости ради, следует заметить, что с изначальным Ingres'ом современный Postgres общего имеет почти что ни хуя, т.к. в конце 80-ых был практически полностью переделан. Доволько таки винрарна, как для продукта столь древнего, достаточно быстро развивается, имеется большое количество сторонних приблуд. Ее уважают админы Oracle за то, что авторы не выебывались, а нормально следовали стандартам. Как и в случае многих других open source программ, очень хреново уживается с виндой. Логотип — голова слона, что символизирует.

- **MySQL**

Тоже open source, но более дикий и лохматый. Самая популярная база для создания веб-сайтов, входит в состав так называемого LAMP (Linux, Apache, MySQL, PHP). Поддерживает несколько движков СУБД, из которых не все поддерживают транзакции ACID (движок MyISAM как раз такой). Его когда-то прикупила себе Sun Microsystems, а теперь, когда сама Sun принадлежит Oracle, MySQL тоже оказался в его загребущих лапах. Сие перепугало сообщество разработчиков, и они быстро форкнули MySQL, запилив MariaDB, Percona и еще парочку менее значимых ответвлений. До 5-ой версии был объектом всеобщих насмешек из-за слабого функционала, неполноценной поддержки даже древних стандартов SQL и обилия крайне рагульных и костыльных решений, во многом спровоцированных как раз таки возможностью использования нескольких движков. Нынче все стало сильно получше, но, как следствие, изрядно усложнилась настройка — по сложности тюнинга современные версии приближаются к PostgreSQL. Впрочем, ситуаций, когда такой тюнинг реально нужен — одна на миллион. Однако, по закону вселенской несправедливости, такая ситуация обязательно возникнет именно у тебя.

- **IBM DB2**

В свое время была весьма популярной и даже использовалась в качестве штатной СУБД в Oracle, однако в какой-то момент IBM откровенно забила на свое детище, в результате чего многие новомодные фишечки в нее попали сильно позже, чем в оппонентов. До сих пор где-то кем-то используется, но в сравнении с любой другой СУБД из данной статьи ее процент на рынке очень невелик.

- **SQLite**

Это не столько СУБД, сколько библиотека, которая позволяет хранить данные в файле, обращаясь в нему как к базе данных с помощью SQL. Там тоже можно создавать таблицы и индексы, выполнять DML и запросы. Удобно в случае, если не хочется возиться с XML или создавать свой формат файла. Ныне имеется практически в каждом телефоне, хотя далеко не всякий разработчик знает, что это именно SQLite прячется за привычными вызовами Android-API.

Админ СУБД (DBA)

Администратор СУБД — это такая темная личность, которая для разработчиков отрабатывает ту же роль, что и админ локальной сети для обычных юзеров. А именно — постоянно вставляет палки в колеса (по их мнению).

- **DBA vs разработчики.**

Эти две стороны чуть ли не постоянно испытывают ненависть друг к другу и всячески пытаются избежать взаимодействия. Претензии разрабов в том, что DBA не дает им делать то, что они хотят, что, как правило, сводится к сваливанию данных в несвязанные таблицы и написанию кривых запросов. Также, админ может не дать пользователям какие-либо права в базе данных и требовать обоснования для их раздачи. В итоге, разрабы, вместо того, чтобы лишний раз подумать или спросить у знающих людей,

начинают городить костыли, чтобы обойти запреты. Отсюда растут ноги у печально знаменитой схемы из 4 таблиц, в которой можно хранить любые данные любой структуры.

Со стороны админа же, пользователи выглядят как варвары, которые забивают гвозди микроскопом. SQL они принципиально знать не хотят, обосновывая это тем, что знать его — это задача админа. Если в команде есть архитектор баз данных или отдельная группа людей, отвечающая за взаимодействие с БД, то это сглаживает проблему, но в реалиях этой страны (да и не только этой) проще нанять лишних 2-3 быдлокодера, чем одного архитектора с сертификатами и опытом. Повальное непонимание работы базы данных, игнорирование переменных привязки, неадекватное использование транзакций, попытки отгородиться от базы с помощью ORM, вызывают хронический facepalm у DBA.

Стоит отметить, что при нынешнем уровне развития железа, большинство компаний вполне могут за вменяемые деньги прикупить тачки, на которых откровенно чрезжопно спроэктированная база будет чувствовать себя относительно нормально. Многие проблемы в таких ситуациях решаются поверхностным тюннингом и частичным переписыванием приложения, направленным на оптимизацию отправляемых в базу запросов. DBA выходят на сцену тогда, когда все вышеперечисленное уже не помогает. Точнее, должны выходить. На практике же их регулярно нанимают "чтоб были" и заставляют разгребать говно, которым по-хорошему должны заниматься наложившие его говнокодеры. Отсюда тонны холиваров, срачей и обидок. Также не стоит забывать, что многие DBA ну очень уж трясутся из-за неполной утилизации ресурсов тачки, из-за чего могут принимать воистину пизданутые решения ради весьма условной выгоды. По мере накопления таковые решения могут изрядно усложнить процесс разработки и поддержку, что вызывает багор уже у программистов и админов.

- **DBA vs руководство.**

Основная задача руководства, как известно, заработать деньги, и чем быстрее тем лучше. И это входит в прямой конфликт с задачей админа наладить стабильную быструю работу базы данных. Из-за этого админ почти никогда не привлекается к процессу разработки базы и его советов никто не спрашивает. Эта задача сбрасывается на команду разработчиков, которая см. выше. То, что они разрабатывают, естественно, глючит и тормозит, но, ЧСХ, претензии по этому поводу валятся на DBA, вызывая у него приступы лютой ненависти.

Если система разрабатывается с нуля, то на ее развитие еще как-то можно повлиять. Однако, чаще всего она либо уже была разработана кем-то до пришествия админа на предприятие, либо была портирована с более простой СУБД, которая не справлялась с нагрузкой, в надежде, что на более мощной системе она заработает быстрее сама по себе. Про масштабируемость руководство слышать не хочет. В случае каких-либо проблем претензии предъявляются угадайте кому.

В интернетах

В интернетах СУБД являются темой срача и фаллометрии, как, впрочем, и все остальное в этой вселенной. Уровень ФГМ на тематических форумах зашкаливает при упоминании вражьей базы данных. Конечно, разные СУБД используются для разных целей, но это не мешает выяснять на никому ненужных синтетических тестах, что Oracle аж на 0.00000001 сек быстрее SQL Server, но стоит туеву хучу денег, что выливается в суммы, сопоставимые с годовым бюджетом небольших государств. Но справедливости ради стоит отметить, что при этом сама СУБД кроссплатформенна в отличие майкрософтовского поделия.

В вебе стандартом де-факто является MySQL, ибо бесплатный и легко установить, но при возрастании нагрузок жутко тормозит. Мускулистов легко затроллить, предложив им нормально расшардить свою БД (справедливости ради, это в той или иной степени проблема всех РСУБД). Также среди мускулистов популярен срач Myisam vs InnoDB, которые по сути низкоуровневые движки, которые непосредственно отвечают за доступ к данным. В последнее время данный срач постепенно сходит на нет, т.к. разрабы на MyISAM забили хуй, а вот InnoDB, напротив, активно развивается, в той же MariaDB вообще планируют со временем выпилить все движки, кроме него. Когда мускула не хватает по производительности, а кредит на Oracle никто не даст, то в дело вступает PostgreSQL, которая также открытая и не тормозит.

Еще один достаточно новый срач — это реляционные СУБД vs Документо-ориентированные. Суть срача в том, что реляционки больше известны, под них больше разработчиков и для них уже все есть, но они жутко тормозят при хранении большого количества однородных данных — тех же хэш-таблиц или деревьев с большим уровнем вложенности и слабой поддержкой индексов, в добавок к этому, их достаточно сложно горизонтально масштабировать (т.е. тупо размазать на н-цать слабых дешевых серваков, вместо одного мощного и дорогого). В то же время как документо-ориентированные обещают изменить мир, но все никак не могут из-за банального отсутствия транзакционности и нормальных проверок корректности схем. И еще тонны плюшечек, которые есть в реляционках. Истина весьма проста: каждой задаче свой инструмент, так что в тех случаях, когда нужно хранить дофига слабо структурированных и не очень важных данных, эта ваша MongoDB будет вполне себе нормальным выбором, а вот если за проеб данных руководство готово яйца отрезать, то без реляционных СУБД никак. Разумеется, быдло этого не понимает и с пеной у рта доказывает превосходство именно своего решения.

Новые разработки

Многие современные СУБД произошли (прямо или косвенно) от Ingres, написанной в Университете Беркли в 70-х (ОМГ! BSD оттуда же!). После того, как основные алгоритмы поиска и хранения данных вместе с языком SQL стабилизировались, разработчики современных систем заняты в основном:

- Добавлением поддержки малопопулярных возможностей SQL;
- Добавлением тучи малопопулярных функций и типов;
- Написанием новой СУБД с нуля (теперь и на PHP);
- Написанием трансляторов из базы данных в объекты (облегчая тем самым работу быдлокодерам);
- Перепиливанием системы частично или полностью для поддержки XML.

Последний пункт является причиной отдельной темы срача, где участники делятся на лагерь, поклоняющийся Oracle, MS SQL Server и DB2 за умение работать с XML без бубна, и лагерь, старающийся убедить противника в том, что XML = УГ.

С легкой руки одной корпорации, которая применяет данный принцип в своей системе, в последнее время часто встречается рекомендация при программировании всех операций по сохранению данных в файлах заменять их записью в «легкие» базы данных типа SQLite. Это позволяет абстрагироваться от конкретной реализации операций чтения/записи файлов в разных системах и переложить всю черновую работу на движок базы данных, который все-таки пилят не самые последние быдлокодеры. Внезапно, ничего нового в этом нет. w:PalmOS юзала этот принцип за год до основания Google.

Object-Relational Mapping (ORM)

Крайне противоречивая НЁХ, люто-бешено обсираемая быдлокодерами, которые надеялись, что теперь…

- Не надо больше изучать SQL
- Не надо писать руками DML.
- Не надо заморачиваться с курсорами.
- Не надо разбираться в планах выполнения запросов.
- Блджад, вообще ничего не надо знать про СУБД!

а вышло:

- говнокод, который, через ORM, адски насилует базу данных во все порты. Так как ORM транслирует его в сотни, тысячи неоптимизированных запросов, причем, бывает, что и откровенно левых.

После чего они получив животворящих пиздюлей от DBA сдриснули на уютненький писать статьи про ужасы ORMa.

Если рассматривать зрелую платформу (например Java SE/EE), реализацию (например JPA/Hibernate актуальной версии) и прогера выше junior, то ПРОФИТ от ORM следующий:

- Скорость разработки и рефакторинга зашкаливающе выше, чем при написании JDBC-запросов врукопашную
- Можно наконец-то писать Persistence Layer по канонам ООП с использованием шаблонов проектирования

Так же с ORM никто (кроме собственного скудоумия) не мешает:

- дергать хранимочки. Для полноценного их использования, правда, ORM нужен как собаке пятая нога. Поля вручную мапить надо будет и там и там.
- писать запросы и выполнять их, лаконичность всё равно выше, чем у JDBC. Однако, ничего сложнее базового SQL тут не напишешь. Аналитические функции, рекурсивные запросы, транспонирование и т. д. идут лесом. Вы купили Oracle? Поздравляю, вы зря потратили деньги.
- бонусом генерировать запросы программно (criteria-api), что позволяет как и в Mongo API избежать целого пласта граблей, связанных с парсингом SQL-команд. Недостаток тот же, плюс жесткий хардкод на выходе, который годится для тривиальных случаев. Если запрос более-менее сложный и длинный — этот код становится наглухо нечитаемым.

Любители ORM в упор отказываются признавать тот факт, что их любимый ООП тупо плохо совместим с реляционной моделью и те компромиссы, на которые приходится идти, порой (да что уж там, скажем прямо — почти всегда) негативно сказываются на производительности. Также, разработчики часто не заморачиваются написанием отдельных запросов для каждой задачи, вместо этого выгребая из базы объекты целиком только ради того, чтобы узнать значение одного поля. Добро пожаловать в энтерпрайз, бро. Это не столько проблема ORM, сколько прогеров и современных методов разработки, но дело усугубляет то чувство вседозволенности, которое дает ORM.

Итого: Для полноценной работы с SQL надо прочитать и понять обычно одну-две книги по SQL. Потом выяснить, что SQL у Оракла и SQL Mysql, сцуко, совсем разный. Прочитать ещё одну-две книги про Oracle. Потом про PostgreSQL или еще что-то. Однако, обычно все останавливаются на 1-2 СУБД, не пытаясь объять необъятное. Для полноценной же работы с ORM надо прочитать и понять одну книгу. Выучить 10-20 аннотаций. Поэкспериментировать несколько дней. И вот, новый быдлокодер готов к промышленной эксплуатации. Отсюда вывод: если вы думаете, что ORM всесилен и самодостаточен, значит вы просто

плохо знаете SQL и работу баз данных. Их более глубокое изучение вас может сильно огорчить.

А также

Субдуральная гематома — понятие медицинское, выглядит как обширное кровоизлияние в мозг. С сабжем связано чуть менее, чем никак, зато по производимому эффекту сходно с процессом изучения структуры и работы с этими вашими СУБД.

См. также

- Быдлокодер

Языки программирования

++i + ++i 1C AJAX BrainFuck C Sharp C++ Dummy mode Erlang Forth FUBAR God is real, unless explicitly declared as integer GOTO Haskell Ifconfig Java JavaScript LISP My other car Oracle Pascal Perl PHP Prolog Pure C Python RegExp Reverse Engineering Ruby SAP SICP Tcl TeX Xyzzy Анти-паттерн Ассемблер Быдлокодер Выстрелить себе в ногу Грязный хак Дискета ЕГГОГ Индусский код Инжалид дежице Капча КОИ-8 Костыль Лог Метод научного тыка Очередь Помолясь Проблема 2000 Программист Процент эс Рекурсия Свистелки и перделки Спортивное программирование СУБД Тестировщик Умение разбираться в чужом коде Фаза Луны Фортран Хакер

Языки программирования

Just Another Fucking Acronym

14/88 1C 265 A.C.A.B. ADSL AFAIK AFK AISB AJAX Aka All your base are belong to us AMV ASAP ASL ASMR ASUS EEE BAT BBS BDSM BOFH BRB BSOD BTW CMS Command & Conquer Copyright Counter-Strike CYA DC DDoS Delicious flat chest Direct Connect DIY DJ Doki Doki Literature Club! DOS DRM EFG Etc Five Nights at Freddy's Frequently asked questions FTL FTN FTW FUBAR GIF GIMP GNAA GPON Grammar nazi Grand Theft Auto GTFO Happy Tree Friends HBO How It Should Have Ended I see what you did there I2P IANAL IDDQD IIRC IMHO In before Internet Explorer IRC IRL ITT JB (ЛОР) JFGI Kerbal Space Program KFC KISS Let's get ready to rumble! LFS Livejournal.com LMAO LMD LOL Low Orbit Ion Cannon M4 MacOS Microsoft MILF MMORPG MSX MTV N.B. NASCAR NEDM NES NoNaMe Not Your Personal Army NRB NSFW O RLY? OK OMG OS/2 P. S. P2P Panty and Stocking with Garterbelt