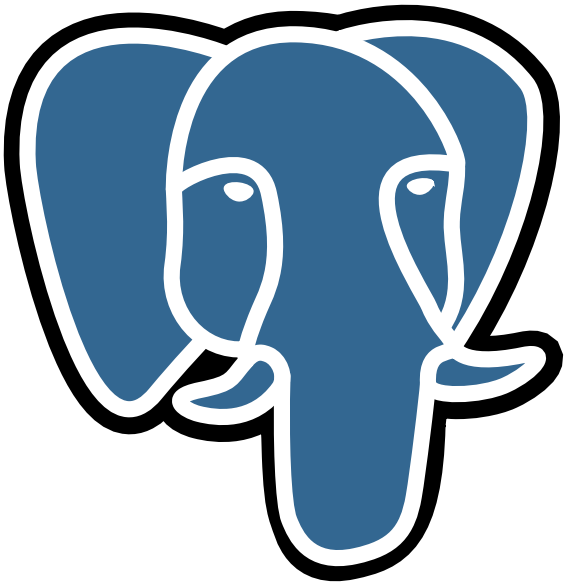

История о PostgreSQL

© Е.М. Балдин*

Эта статья была опубликована в ноябрьском номере русскоязычного журнала Linux Format (`http://www.linuxformat.ru`) за 2006 год. Статья размещена с разрешения редакции журнала на сайте `http://www.inp.nsk.su/~baldin/` и до апреля месяца все вопросы с размещением статьи в других местах следует решать с редакцией Linux Format. Затем все права на текст возвращаются ко мне.

Текст, представленный здесь, не является точной копией статьи в журнале. Текущий текст в отличии от журнального варианта корректор не просматривал. Все вопросы по содержанию, а так же замечания и предложения следует задавать мне по электронной почте `mailto:E.M.Baldin@inp.nsk.su`.

Текст на текущий момент является просто *текстом*, а не книгой. Поэтому результирующая доводка в целях улучшения восприятия текста не проводилась.

*e-mail: E.M.Baldin@inp.nsk.su

Слон взят с сайта `http://pgfoundry.org/projects/graphics/`. Изображение предоставляется под лицензией BSD.

Оглавление

Разве же так можно? Разве же такие вещи алгоритмизируешь?

Магнус Ф. Редькин об определениях счастья.

Глава 1

Введение

Новая информация добывается потом и кровью. Чтобы не утерять найденное — её надо сохранить. А чтобы потом суметь найти необходимое — её следует структурировать. PostgreSQL — предназначен для постоянного[1] хранения структурированных данных[2].

1.1 Это кто такой?

PostgreSQL — это реляционная база данных. PostgreSQL — это программный продукт с открытым исходным кодом и свободной (в прямом смысле этого слова) лицензией. Собственно говоря, этим всё сказано.

1.1.1 Реляционная база данных

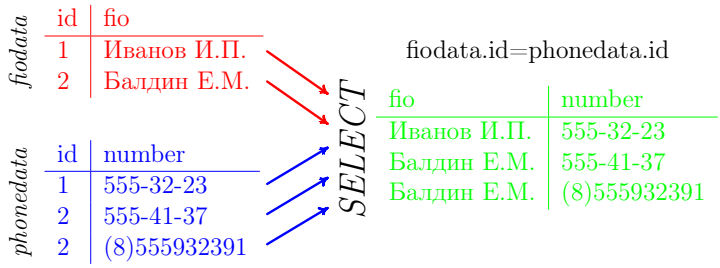

Рис. 1.1. Получение новой таблицы из уже имеющихся

Информация в реляционных базах данных хранится в виде обычных плоских двумерных таблиц. Доступ к данным в таблице можно получить по её имени. В таблице есть именованные столбцы (column) и строки (row) — очень простая и понятная концепция. Пользователю предоставляется набор операторов, результатом

[1] Постоянность означает сохранность данных, даже если программа перестала работать.

[2] Хранить можно и *не* структурированные данные, но это уже моветон.

действий которых так же являются таблицы. Это особенность реляционной базы данных называется *замкнутость*. Это очень важное свойство, так в результате любых действий порождаются объекты того же типа, что и объект над которым совершались эти самый действия. Следствием замкнутости является возможность применять к результату все имеющиеся в наличии операторы. Иными словами можно пользоваться *вложенными выражениями*[3].

1.1.2 Открытый исходный код

PostgreSQL распространяется под BSD лицензией. Почему не GPL? Ответ разработчиков можно перевести[4] примерно так: «PostgreSQL создавался в Беркли (Berkeley), как, собственно говоря, и лицензия BSD. Эта лицензия служила нам верой и правдой много лет. От добра — добра не ищут. Просьба не начинать опять „флеймить" по этому поводу.»

1.2 Генеалогия

Понятие реляционных баз данных было предложено в 70-ых годах прошлого века сотрудником фирмы IBM Эдгаром Ф. Коддом (Edgar F. Codd). В то время это была революция в сфере хранения данных. Головокружительный успех идей Кодда связан ещё и с тем, что он сумел воплотить математическую абстракцию под названием *реляционная алгебра* в жизнь. Многие ответы на практически вопросы были найдены теоретически с использованием математики.

С тех пор прошло более тридцати лет и новой революции пока не предвидится. Двумерные таблицы ещё долго будут основным методом структурирования информации в силу исключительной простоты решения.

Как и в случае TCP/IP практическое воплощение теории в жизнь началось с того, что DARPA (Defense Advanced Research Projects Agency) дало денег профессору. Профессор Михаил Стоунбрэйкер (Michael Stonebraker) написал реляционную базу данных POSTGRES, первый релиз которой был сделан в 1987 году. Профессор Стоунбрэйкер писал базу не с нуля. Его проект основывался на одной из самых первых реляционных баз данных Ingres к созданию которой приложил руку сам Кодд — её имя частично присутствует в названии проекта (POST-GRES — после Ingres).

POSTGRES использовался как для реальных дел в качестве СУБД, так и для исследования теории реляционных баз данных в стенах университетов. В 1994 году два студента Андрэ Ю (Andrew Yu) и Джолли Чен (Jolly Chen) добавили движок SQL, который уже к этому моменту стал бесспорным промышленным стандартом для реляционных СУБД. Так появился Postgres95 который в 1996 году сменил имя на PostgreSQL. Имя больше не менялось, но активная разработка не прекращается

[3] Вложенные выражения, это многоуровневые выражения, причём, использование имён реальных таблиц обязательно только на самом низком уровне. В остальных случаях в качестве объектов действия могут быть вычисляемые выражения.

[4] Очень вольный перевод.

не на миг. Последняя версия сервера баз данных на февраль 2007 года 8.2.3. Подробнее об истории можно узнать в стандартной документации, идущей с программой или на сайте `http://www.postgresql.org`.

Семейство Ingres/PostgreSQL породило множество коммерческих реализаций[5] систем управления баз данных, благо лицензия позволяет.

1.3 А как оно работает?

На рис. 1.2 показана схема работы типичного приложения. Процессу POSTMASTER, который существует всегда[6] на серверную машину посылается запрос на подключение. Если запрос на подключение проходит проверку, то POSTMASTER создаёт свою копию. Все дальнейшие операции между базой данных и клиентом проводятся через эту копию POSTMASTER. На каждое соединение создаётся своя копия — это позволяет производить все действия с данными *непосредственно* на сервере.

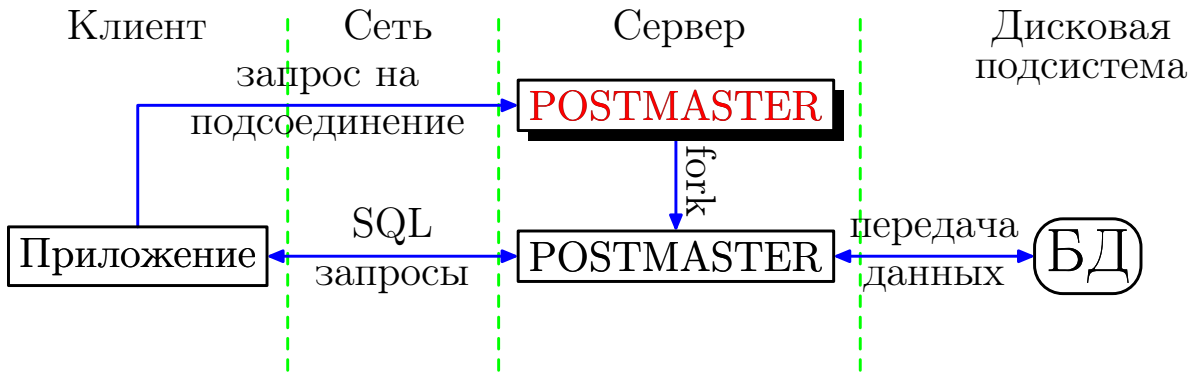

Рис. 1.2. Схема работы приложения с PostgreSQL

1.4 Установка и запуск

Установка базы данных это не совсем тривиальная процедура. Лучше довериться стандартной сборке из вашего базового дистрибутива, даже если версия по умолчанию кажется устаревшей[7]. Самостоятельно собирать пакет из исходников лучше только в том случае, если точно известно, что в базовой версии нет необходимой функциональности.

[5] Особенно много потомков у Ingres — та же Sybase. Код Sybase в свою очередь в 1988 году был продан одной известной фирме, которая в 1992 году выпустила продукт в названии которого есть имя этой фирмы и слова «SQL Server». У POSTGRES в прямых потомках ходит Informix.

[6] За исключением тех случаев когда компьютер выключен или сервис был остановлен администратором и *очень* редко по причине какой-либо ошибки. Я не знаю какая статистика у других, но из моего опыта все такого рода ошибки связаны с человеческим фактором.

[7] Базовая версия PostgreSQL в Debian stable (Sarge) на момент написания статьи 7.4.7, в то время как последняя версия базы данных 8.2.3.

Если в будущем необходимо будет сменить версию PostgreSQL, то следует учитывать, что в случае крупных изменений (major releases) могут изменять внутренние форматы системных таблиц и файлов данных. В этих случаях необходимо выполнить процедуру «dump/restore», которая гарантировано сохранит данные при «переезде». В отличии от крупных изменений, небольшие правки (minor releases[8]) как правило не требуют никаких действий со стороны администратора БД.

Число пакетов в дистрибутиве в описании которых упоминается PostgeSQL довольно велико. Например, в Debian (Sarge) таких пакетов 182, что несколько меньше чем число пакетов связанных с именем mysql (212), но превышает число упоминаний InterBase/Firebird (22), sqlite (50) и, естественно, Oracle (19). Это не о чём не говорит, но корреляция, скорее всего, какая-то есть. К счастью все 182 пакета ставить не обязательно — для Debian (Sarge) достаточно двух/трёх пакетов:

```
# устанавливаются исполняемые файлы и файлы настроек
# необходимые для функционирования Базы Данных
> apt-get install postgresql
```

В случае подобной установки в обязательном порядке доставляется базовый набор программ, которые можно ставить на клиентских машинах для удалённой связи с БД — пакет **postgresql-client**.

Если же несмотря ни на что хочется установить всё из исходников самостоятельно, то следует выполнить примерно следующую последовательность действий:

```
> wget ftp://ftp.postgresql.org/pub/source/v8.2.3/postgresql-8.2.3.tar.bz2
> tar xvfj postgresql-8.2.3.tar.bz2
> cd postgresql-8.2.3
> ./configure
> make
> su
> make install
> adduser postgres
> mkdir /usr/local/pgsql/data
> chown postgres /usr/local/pgsql/data
> su - postgres
> /usr/local/pgsql/bin/initdb -D /usr/local/pgsql/data
> /usr/local/pgsql/bin/postmaster -D /usr/local/pgsql/data >logfile 2>&1 &
> /usr/local/pgsql/bin/createdb test
> /usr/local/pgsql/bin/psql test
```

Разберём то что происходит поподробнее. После того как с помощью wget получен и распакован архив исходников, привычные команды ./configure и make позволяют осуществить сборку PostgreSQL. Установку (make install) следует производить под суперпользователем (su). После установки необходимо добавить пользователя postgres от имени которого и будет запущен сервер **postmaster**. По умолча-

[8]Меняется только последнее число в версии, то есть переход от версии 7.4.0 к версии 7.4.1.

нию установка программы производится в директорию /usr/local/pgsql/. Для хранения файлов базы предлагается создать директорию /usr/local/pgsql/data. Данные должны принадлежать пользователю postgres (команда chown). В этой же директории хранятся и файлы настройки.

Дальнейшая настройка производится под пользователем postgres (su − postgres). С помощью команды initdb производится инициализация хранилища данных, а вслед за этим производится запуск сервера postmaster. Последние две строчки создают тестовую базу данных test (createdb) и проверяют что к ней можно подсоединиться (psql). Если всё прошло нормально, то должно появиться приглашение вида:

```
Welcome to psql 8.2.3, the PostgreSQL interactive terminal.

Type:  \copyright for distribution terms
       \h for help with SQL commands
       \? for help on internal slash commands
       \g or terminate with semicolon to execute query
       \q to quit

test=#
```

При установке стандартными средствами дистрибутива описанные выше действия выполняются автоматически кроме последних двух строчек. В случае Debian (Sarge) при установке PostgreSQL можно указать где именно расположить директорию с данными. По умолчанию всё помещается в /var/lib/postgres/data. Для других дистрибутивов возможны вариации. Для выяснения подробностей следует изучить README. Например, в случае Debian особенности пакета связанные с дистрибутивом описаны в /usr/share/doc/postgresql/README.Debian.gz. Ниже, если не указано специально, все действия выполняются для дистрибутива Debian (Sarge).

Для администрирования базы данных нет необходимости в суперпользовательских привилегиях. Для этого можно настроить sudo (man sudo), то есть в файл /etc/sudoers (man sudoers) следует добавить примерно следующие строки:

```
# /etc/sudoers
Host_Alias HOME = localhost
User_Alias DBADM = "ваше имя, если Вы администратор базы данных"
Cmnd_Alias DB  = /etc/init.d/postgresql
DBADM HOME =  NOPASSWD: DB
DBADM HOME = (postgres) NOPASSWD: ALL
```

Файлы настройки принадлежат пользователю postgres, поэтому для их изменения необходимо иметь возможность заходить под этим пользователем:

```
> sudo -u postgres bash
> whoami
```

```
postgres
```

либо добавить себя в группу postgres и разрешить этой группе редактировать конфигурационные файлы в директории `/etc/postgres` (chgrp плюс chmod g+w).

Скрипт `/etc/init.d/postgresql` позволяет управлять процессом postmaster

```
> sudo /etc/init.d/postgresql
Usage: /etc/init.d/postgresql {start|stop|autovac-start|
autovac-stop|restart|autovac-restart|reload|force-reload|status}
> sudo /etc/init.d/postgresql status
pg_ctl: postmaster is running (PID: 10868)
Command line was:
/usr/lib/postgresql/bin/postmaster '-D' '/home/postgres/data'
```

Этот же скрипт используется для автоматического запуска сервера при загрузке компьютера. От дистрибутива к дистрибутиву название инициализирующего скрипта может меняться.

После настройки сервера необходимо создать базу данных:

```
> sudo -u postgres createdb "имя_БД"
CREATE DATABASE
```

и завести пользователя:

```
> sudo -u postgres createuser "имя_пользователя"
Разрешить новому пользователю создавать базы? (y/n) n
Разрешить новому пользователю создавать пользователей? (y/n) n
CREATE USER
```

Для того чтобы пройти проверку при запросе на подключения необходимо, чтобы конфигурационный файл `pg_hba.conf` был соответствующим образом настроен. Например, чтобы можно было подключаться к базе данных под тем же именем под которым Вы работаете, в `pg_hba.conf` должны быть примерно следующие строки:

```
#/etc/postgresql/pg_hba.conf
# TYPE  DATABASE    USER IP-ADDRESS    IP-MASK         METHOD
# connections by UNIX sockets
local   all         all                                ident sameuser
# All IPv4 connections from localhost
host    all         all  127.0.0.1 255.255.255.255 ident sameuser
```

Здесь в качестве метода идентификации используется метод ident sameuser[9]. Создав пользователя в соответствии с текущей учётной записью можно подсоединиться к PostgreSQL и начать общаться с сервером базы данных на его родном языке SQL:

[9]Существует более либеральный метод проверки trust — в этом случае пускается кто угодно и под каким угодно пользователем. То есть метод «двери настежь» — некоторым нравится.

```
> psql "имя_БД"

"имя_БД"=> SELECT fio,number FROM fiodata, phonedata
"имя_БД"=>        WHERE fiodata.id=phonedata.id;
      fio       |      number
----------------+------------------
 Иванов И.П.    | 555-32-23
 Балдин Е.М.    | 555-41-37
 Балдин Е.М.    | (+7)5559323919
(записей: 3)
```

К вопросу о номере порта По умолчанию для создания TCP/IP соединения postmaster использует порт (port) за номером 5432. Если номер порта отличается от установленного по умолчанию, то **postmaster** должен быть запущен с ключом `-p [номер порта]`. Для выяснения наверняка достаточно выполнить:

```
> ps axw  grep postmaster  grep -v grep
 4181 ?        S      0:00 /usr/lib/postgresql/bin/postmaster
-D /home/postgres/data
```

Так же номер порта может храниться в переменной окружения `$PGPORT`.

1.5 Почему?

Люди обычно работают с текстовыми файлами. Подавляющий объём структурированной информации до сих пор доставляется до нашего сознания через текст. Без помощи компьютера, просто набивая текст руками, информации производится не так уж и много. Необходимость базы данных в начале пути накопления личной информации не является очевидной. Всё сделанное в принципе реально окинуть взглядом.

1.5.1 Почему БД?

То, что создал один человек, другой человек с большой долей вероятность освоить в состоянии, но разобраться с наследием двух и более людей становится труднова-то. А если это не наследие, а информация идущая в реальном времени из многих (десятков, сотен, тысяч, миллионов) источников? Для начала всё это надо куда-то сохранить, то есть необходимо надёжное хранилище по возможности ни от чего независимое.

Ну это ещё пол беды: данные надо как-то извлечь, причём извлечь надо не абсолютно все данные, иначе человеческий мозг в них утонет, а только нужные. Ком-

пьютеры человеческие мысли пока ещё надёжно[10] не читают, поэтому для начала необходимо нужные данные как-то пометить и лучше это сделать в момент «укладки» в хранилище. То есть хранилище должно быть структурированным, причём структуру можно задавать извне до появления данных.

Когда данных немного — жить можно и так, оставляя ключевую информацию на обрывке листика, надеясь что он не затеряется. Обрывок листика слабо отличается от записи в каком-то файле. Текстовые утилиты типа **grep** существенно облегчают поиск информации, но всегда в конце концов настаёт момент, когда данных становится либо слишком много, либо они слишком часто изменяются и нужно вводить систему — Систему Управления Базой Данных или СУБД.

1.5.2 Почему PostgreSQL?

Когда я примерно семь лет назад пытался понять какую СУБД следует использовать для обеспечения эксперимента в котором я участвую до сих пор, то выбора просто не существовало. Из свободных СУБД только PostgreSQL на тот момент обладал необходимой функциональностью. На сегодня вопрос выбора немного усложнился: подросла в хорошем смысле этого слова MySQL (в последней 5ой версии, говорят, наконец-то даже триггеры появились), были открыты исходники проекта Firebird, в девичестве Interbase от фирмы Borland, да и «игрушечные» проекты типа SQLite тоже не лишены определённых преимуществ. Ну и, естественно, свет клином на открытых разработках не сошёлся — тот же Oracle предлагает свои СУБД для изучения. И всё-таки я *выбираю* PostgreSQL — решение шестилетней давности меня не разочаровало. На редкость устойчивая к внешним воздействиям программа с абсолютно предсказуемым поведением. Даже те случаи, которые мне по неопытности показались «граблями» оказались «фичами» ☺.

Одной из основных целей, которая была поставлена при разработке PostgreSQL является соответствие стандартам. PostgreSQL очень строго следовал ANSI SQL-92, SQL-99 (SQL-2 и SQL-3, соответственно), а теперь и ANSI SQL:2003. Мало кому[11] удаётся похвастаться подобным соответствием стандартам.

В дополнение к стандартам PostgreSQL поддерживает множество полезных расширений. Примером мелкого, но полезного расширения не входящего в стандарт SQL является дополнения к условию для SELECT вида LIMIT/OFFSET[12], которые позволяют получить только указанные строки из результата запроса. PostgreSQL полностью поддерживает механизм транзакций (transactions), вложенные запросы (subselects), триггеры (triggers), представления (views), функциональные индексы,

[10] Удачные опыты по управлению курсором мыши или манипулятором уже зафиксированы, правда, до сих для этого требуются имплантанты. Без имплантантов операторы могут передавать только самые простейшие команды и вряд ли в ближайшее время ситуация кардинально изменится.

[11] Возможно, что вообще некому. В большинстве случаев следование стандарту заканчиваться на вводном (entry) уровне SQL-92.

[12] Мне эти инструкции в своё время сильно облегчили жизнь, точнее увеличили скорость выполнения нужных мне запросов.

ссылочную целостность по внешнему ключу (foreign key referential integrity), изощрённые типы блокировок (sophisticated locking) и многое другое.

К названию PostgreSQL обычно прибавляется слово объектная, то есть полное название звучит как объектно-реляционная база данных PostgreSQL. Пользователю предоставляются необходимые инструменты для создания новых типов данных, функций, операторов и своих методов индексирования. Подобные возможности позволяют работать с довольно нестандартными данными, например, с картографическими объектами — PostGIS (`http://postgis.refractions.net/`).

Размер базы данных, управляемой PostgreSQL не ограничен, так же нет ограничения и на число строк в таблице. Да вообще есть ли ограничения у этого чуда? Да, есть: ваша таблица не может быть больше чем 32 Тбайта, а число столбцов в таблице не может быть больше 250–1600 в зависимости от типа данных. Много это или мало? Зависит от задачи: я, например, как-то упёрся в ограничение по числу столбцов, но скорее по неопытности нежели по необходимости. Описанное выше верно для версии PostgreSQL 8.2.3. Возможно в будущем будут сняты и эти ограничения.

Существует родные интерфейсы для работы с PostgreSQL из языков Java (JDBC), Perl, Python, Ruby, C, C++, PHP, Lisp, Scheme и всего что может связаться через ODBC. PostgreSQL поддерживает хранимые процедуры которые можно написать на множестве языков программирования, включая Java, Perl, Python, Ruby, Tcl, C/C++ и родном для PostgreSQL PL/pgSQL.

По результатам автоматизированного тестирования, проведённом в 2005 году (`http://www.postgresql.org/about/news.363`) в коде PostgreSQL было обнаружено 20 дефектов, что соответствует 1 ошибке на 39 тысяч строк кода. Для сравнения аналогичное тестирование примерно в то же время выявило в ядре Linux по одному дефекту на 10 тысяч строк кода, а в MySQL одно проблемное место приходится на 4 тысячи строк кода. Это не о чём не говорит, так сказать, мелочь, зато разработчикам и пользователям PostgreSQL приятно.

1.6 Информация о subj

Книг по PostgreSQL, выпущенных на русском языке, относительно[13] не много, но они есть и количество их будет расти. Эта область технических знаний не так популярна, как следовало бы. Очевидно, что в будущем без надёжных хранилищ данных будет непросто управляться со всё возрастающим потоком информации.

С другой стороны наличие отличной документации (в том числе и русскоязычной) позволяет достаточно безболезненно «войти в тему». Вполне можно обойтись и без специфичных для PostgreSQL возможностей, а для изучения основ SQL годится любая нормальная книга, коих довольно много. Для введения вполне сгодится «SQL» от Мартина Грабера. Собственно говоря, хватит и стандартной документации, которая идёт в дистрибутиве.

Основной сайт PostgreSQL `http://www.postgresql.org`. Там расположено в том числе и первичное хранилище обширной документации, в которой есть фактиче-

[13]Например, относительно числа книг по PHP+MySQL.

ски вся «мудрость мира», имеющая хоть какое-то отношение к PostgreSQL — надо только уметь читать.

По адресу `http://www.linuxshare.ru/postgresql/`[14] представлена русскоязычная версия сайта. Там же можно найти информация о русскоязычном тематическом списке рассылки `pgsql-ru-general@postgresql.org`[15]. Список не сильно активный, но если хочется перемолвиться о «subj» по русски вполне сгодится.

[14] Виктору Вислобокову, который поддерживает этот ресурс, очевидно нужна помощь.

[15] Подписаться на список рассылки можно, послав письмо на адрес `majordomo@postgresql.org`. В теле письма должна быть указана строчка: subscribe pgsql-ru-general.