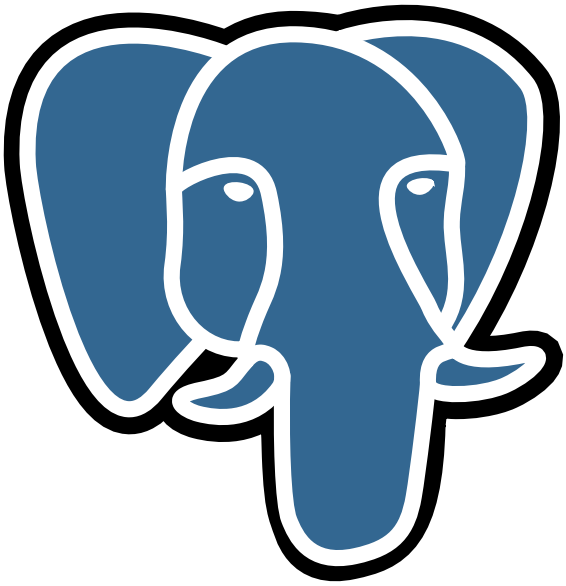

История о PostgreSQL

© Е.М. Балдин*

Эта статья была опубликована в декабрьском 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 мая 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

Это прибор, — С ним работают

Витька Корнеев о
диване-трансляторе.

Глава 2

Работа с базой

В прошлой части было описано как создать базу данных и запустить **postmaster**. Дело за малым: надо научиться сохранять данные и доступаться до них. Для этого следует договориться с **postmaster** — благо его «родной» язык довольно высокоуровневый.

Как и в предыдущей части всё рассматривается с точки зрения дистрибутива Debian (Sarge). При прочтении на это следует делать поправки.

2.1 SQL

В качестве языка общения с реляционными базами данных в подавляющем большинстве случаев используется язык SQL. Изначально эти три буквы были сокращением фразы Structured Query Language (язык структурированных запросов). Сейчас, когда язык стал стандартом, SQL уже не является аббревиатурой — это обычное название, которое произносится как «эс-кью-эл». Несмотря на это, даже англоязычные специалисты по прежнему часто называют SQL «сиквел». По-русски также часто говорят «эс-ку-эль».

У этого языка есть недостатки, что приводит к тому что в реальности он дополняется различными расширениями. Кстати, сам Кодд — «отец» реляционных баз данных считал SQL неудачной реализацией его теории. Но на сегодня это мощный открытый промышленный стандарт, который позволяет решать множество типовых задач по созданию, модификации и управления данными — он есть здесь и сейчас.

За время своего существования SQL претерпел несколько ревизий. В таблице 2.1 перечислены основные ревизии стандарта.

Степень соответствие PostgreSQL стандарту SQL:2003 подробно рассмотрена в Приложении D (Appendix D. SQL Conformance) стандартной документации.

В стандартной же документации есть и простейший учебник, и исчерпывающий справочник по SQL. Существует море литературы в которой подробно и не очень рассказывается что же это за «зверь такой» SQL. Необходимый для «вхождения в

Год	Ревизия	Нововведения
1986	SQL-86, SQL-87	Первая версия стандарта ANSI. Принят ISO в 1987 году. Стандартизация синтаксиса.
1989	SQL-89	Стандартизован механизм ссылочной целостности.
1992	SQL-92 (SQL-2)	Множество нововведений. В отличии от предыдущих версий, где стандарт просто сертифицировал уже имеющиеся на рынке реляционных БД возможности были заложены основы для развития языка. Введены три уровня соответствия стандарту: Entry (начальный), Intermediate (промежуточный), Full (полный). Мало какая из баз данных поддерживает SQL-92 больше чем Entry.
1999	SQL:1999 (SQL-3)	Добавлены регулярные выражения, рекурсивные запросы, триггеры. Определена интеграция с объектно-ориентированным подходом. Вместо трёх уровней соответствия введён набор свойств (features).
2003	SQL:2003	Стандартизованы XML-зависимые нововведения, интервальные функции (window functions), стандартные последовательности и столбцы с автоматически генерируемыми значениями.

Таблица 2.1. Ревизии SQL

технологию» минимум настолько прост, что основы изучаются в пределах одного дня вдумчивого чтения учебника.

Для того чтобы данные куда-то сохранить, необходимо создать «хранилище» для них — таблицу/таблицы:

```
CREATE TABLE fiodata (id int, fio text)
CREATE TABLE phonedata (id int, number text)
```

Теперь можно добавлять данные:

```
INSERT INTO fiodata VALUES (1, 'Иванов И.П.')
INSERT INTO phonedata VALUES (1, '555-32-23')
```

и так далее. Созданы две обычные таблицы «без наворотов» в одной хранятся имена, а в другой телефоны. Сопоставление телефонов именам идёт через поля `id`. Зачем так? На одно имя может быть заведено несколько телефонов, а на одном телефоне может «сидеть» несколько человек.

Теперь надо данные извлечь и в этом нам поможет оператор SELECT. Собственно говоря, пользователю кроме этого оператора больше ничего знать и не надо — все выборки делаются с помощью него. Выведем все имена и соответствующие им телефоны:

```
SELECT fio, number
        FROM fiodata, phonedata WHERE fiodata.id=phonedata.id
```

SQL заслуживает большего чем это «микровведение», но его и так и сяк придётся изучить тем, кто реально хочет связаться с базами данных. То есть надо читать книжки. А если подходить к делу серьёзно, то кроме описания SQL следует изучить и основы реляционных баз данных, того же К. Дж. Дейта (C. J. Date) — но это уже совсем другая история.

2.2 Командная строка

Когда набирается текст, а SQL это именно текст, то лучше чтобы ничего вокруг не отвлекало. Надёжная, «толстая» и дешевая связь вещь хорошая, только вот не всегда она случается. Часто командная строка вне конкуренции.

2.2.1 psql

Вместе с пакетом **postgresql-client** поставляется утилита psql — интерактивная оболочка для «разговоров» с PostgreSQL. Она же — лучший инструмент для администрирования.

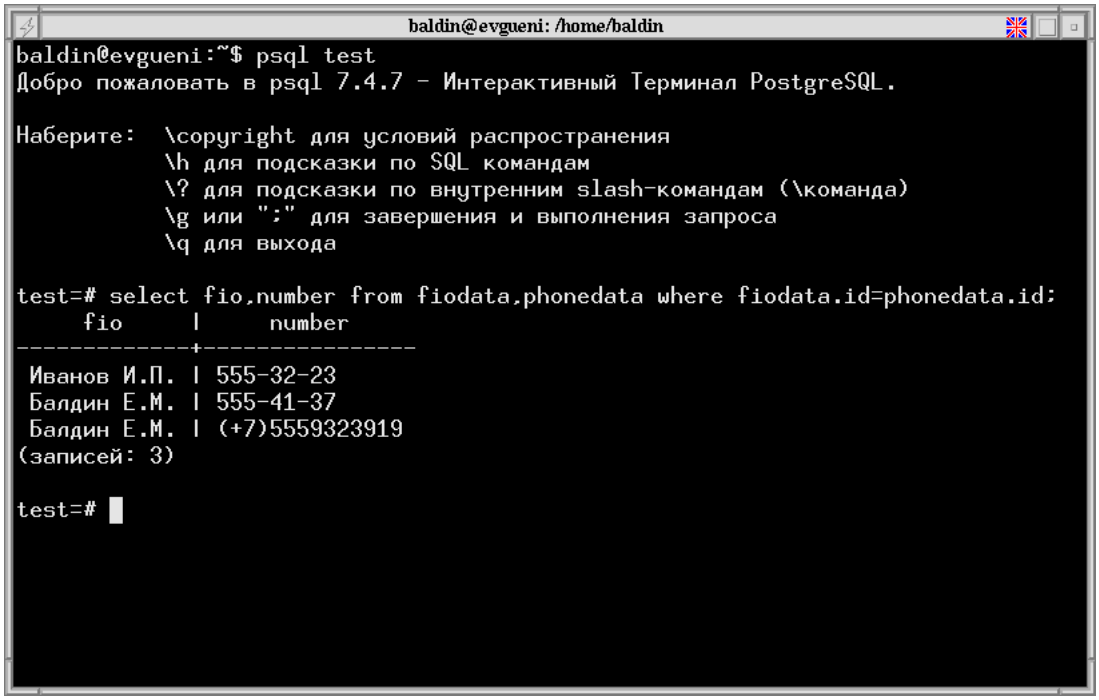

Рис. 2.1. Окно psql

Пусть существует база данных test, в которой заведены таблицы fiodata и phonedata, описанные в предыдущем разделе. Подсоединимся к базе и что-нибудь «спросим» у неё:

```
> psql test
```

```
Добро пожаловать в psql 7.4.7 – Интерактивный Терминал PostgreSQL.

Наберите:  \copyright для условий распространения
           \h для подсказки по SQL командам
           \? для подсказки по внутренним slash–командам (\команда)
           \g или ";" для завершения и выполнения запроса
           \q для выхода

test=> SELECT fio, number
test->       FROM fiodata,phonedata 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)
```

Если хочется подсоединиться к серверу на другой машине, то нужно указать имя машины после ключика `-h`. Ключик `-U` позволяет сменить имя пользователя по умолчанию.

psql позволяет передавать SQL-команды серверу. Обратите внимание, что для завершения SQL-команды используется точка с запятой «;».

Как и всякая человеко-ориентированная оболочка **psql** использует библиотеку Readline. Это означает наличие стандартных горячих emacs-подобных комбинаций символов для общепринятого редактирования ввода командной строки, в том числе и завершение SQL-команд по «TAB». По клавише «TAB» завершаются не только SQL-команды, но и названия таблиц и имена колонок, если это возможно.

psql поддерживает историю команд, которая сохраняется в `.psql_history`. Это так же особенность библиотеки Readline. Полезным является интерактивный поиск по истории команд, который вызывается с помощью комбинации `C^r`.

Кроме команд SQL **psql** имеет набор собственных специальных команд. Все такие команды начинаются с обратной косой черты «\». Число спец-команд довольно обширно и полное их описание можно найти выполнив команду man psql. Далее будет перечислены наиболее интересные из них:

\q Закончить работу с **psql**. Выйти из оболочки.

\? Вывести справку по имеющимся спец-командам.

\h [SQL-команда] Вывести помощь по запрашиваемой SQL-команде в форме Бэкуса-Наура (Backus Naur Form). SQL-команда может состоять из нескольких слов. При исполнении `\h` без аргумента выводится список доступных SQL команд.

\! [shell-команда] Запустить командный интерпретатор и выполнить shell-команду.

\i «файл» Прочить текстовый «файл» и выполнить имеющиеся в нём команды. Удобно для нетривиальных операций.

Имя файла с командами можно передать при запуске **psql** через ключик `-f`. В этом случае после чтения и исполнения всех команд **psql** автоматически прекращает работу.

\o [«файл»] Сохранить результаты выполнения будущих команд в «файл». Если аргумент отсутствует, то вывод переключается на терминал. **psql** имеет набор команд, которые позволяют сформировать вывод как угодно пользователю.

Указать имя файла, в котором следует сохранить результаты, также можно передать при запуске **psql** с помощью ключика `-o`. Этот ключ удобно применять совместно с ключом `-f`.

\d [«регулярное выражение»] Вывести структуру объекта. Годится для таблицы (table), представления (view), индекса (indexes) или последовательности (sequences). Список объектов можно получить добавив первую букву названия объекта `t`, `v`, `i`, `s` к команде `\d`.

В дополнение к вышесказанному, **psql** поддерживает простейший механизм присваивания значений собственным переменным и их интерполяции в SQL-запросах:

```
test=> \set proba 'phonedata'
test=> select * from :proba;
 id |      number
----+------------------
  2 | 555-41-37
  2 | (+7)5559323919
  1 | 555-32-23
(записей: 3)
```

Следует учитывать, что интерполяция переменных не работает, если переменная используется внутри SQL-строки. В любом случае это хоть что-то.

2.2.2 gql-shell

Небольшая **psql**-like оболочка созданная одним человеком. Разработка заморожена. Естественно, **gql-shell** не владеет всеми возможностями **psql**, зато может подсоединяться и «разговаривать» не только с PostgreSQL. Для подсоединения к базе данных используется библиотека GQL (Generic C++ SQL Library). Для работы с PostgreSQL необходимо установить драйвер:

```
> sudo apt-get install gql-shell
> sudo apt-get install libgql-driver-0.5-pg
> gql-shell pg:test
Welcome to gql-shell, the interactive SQL 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=>
```

2.2.3 dbishell

dbishell — интерактивная оболочка на основе Perl::DBI. Как и в случае с **gqlshell**, поддерживает не только PostgreSQL. **dbishell** представляет из себя скрипт на perl и занимает при установке чуть больше 150 кб.

```
> sudo apt-get install dbishell
> dbishell --driver Pg /
           --dsn host=localhost\;database=test /
           --user baldin
Password:
Using DBIShell::dr::Pg engine

dbi:Pg:host=localhost;database=test:baldin>quit/
```

Для завершения любой команды используется косая черта «/».

2.3 GUI в помощь

Следует признать, что программа с графическим пользовательским интерфейсом выглядит гораздо солиднее какой-то там командной строки. Об эффективности работы в случае необходимости показать, что занят важным делом речь, естественно, не идёт. Зачем один терминал, когда можно открыть кучу красивеньких окошечек с иконками? ☺

2.3.1 PgAccess

Когда обсуждается графический пользовательский интерфейс к PostgreSQL, тут же всплывает слово PgAccess (`http://www.pgaccess.org/`). PgAccess был создан Константином Теодореску (Constantin Teodorescu) и имеет довольно длительную историю развития. На текущий момент разработка, похоже, заморожена. С другой стороны «нет худа без добра»: новых версий тащить не надо — достаточно поставить то, что идёт стандартно с Вашим дистрибутивом:

```
> sudo apt-get install pgaccess
> pgaccess
```

Для того чтобы подсоединиться к базе данных, необходимо воспользоваться диалогом открытия соединения: **Database** → **Open**. По умолчанию, предполагается что **postmaster** запущен на этом же компьютере (Host: localhost) и «слушает» порт номер *5432*. Если при установке PostgreSQL ничего специально не делалось, то так оно и есть. Далее требуется указать базу данных к которой надо подсоединиться, пользователя и, если необходимо, пароль.

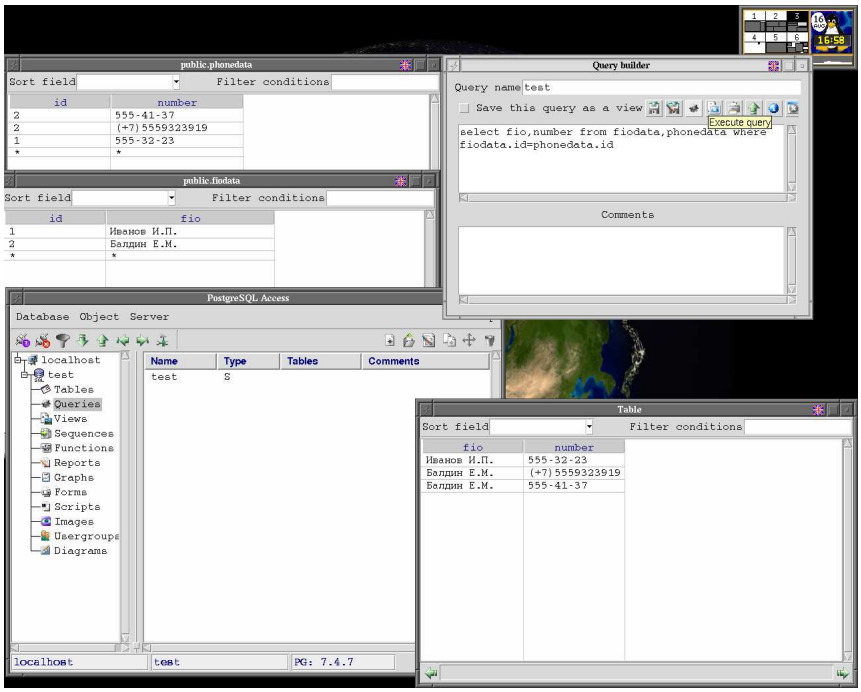

Рис. 2.2. pgaccess явно что-то умеет

PgAccess — это кросс-платформенный графический интерфейс к PostgreSQL, написанный на чистом Tcl/Tk, и как следствие этого, работает везде, где этот инструментарий имеется, даже на «альтернативной» платформе. Размер дистрибутива по современным меркам крошечный: при установке всё укладывается в 4 Мб.

В программе есть возможность создавать, редактировать и просматривать таблицы, запросы, представление, функции, пользователей, то есть довольно многое из того, что можно делать с помощью SQL. Плюс к этому можно создавать графические формы для ввода/просмотра данных, рисовать простые диаграммы и графики, просматривать картинки, сохранённые в базе данных. Так как программа написана на Tcl/Tk, то есть возможность писать свои скрипты используя объекты уже определённые в PgAccess.

Если хочется «сляпать» на скорую руку формочку, которую можно запустить фактически где угодно после минимального использования напильника, то PgAccess вполне может подойти для этого дела. В самом PgAccess масса ограничений и недостатков, но так как программа относительно небольшая, то её можно доделать «по месту».

Информацию о созданных формах, запросах и тому подобных объектах PgAccess сохраняет непосредственно в базе данных в таблицах, начинающихся с префикса `pga_`. Так что то, что сделано кем-то одним, будет доступно и *всем* пользователям базы.

P.S. После первого запуска необходимо настроить шрифты: **Database** → **Preferences** → **Look & Feel**. Выбор шрифтов по умолчанию не совсем адекватен. С другой стороны при желании это настраивается.

P.P.S. Наличие PgAccess на машине, с моей точки, зрение поощряет нездоровое желание у пользователей что-то «сляпать», а не сделать по человечески. Так что работать с этим предметом надо осторожно, и, если нет необходимости, то лучше убрать его от греха подальше. По мне, так psql *гораздо* удобнее и эффективнее, а самое главное пользователи в БД гораздо реже наведываются ☺.

2.3.2 pgAdmin III

Программа порадовала заявлением, что она наиболее популярная и «фичастая» платформа администрирования и разработки для PostgreSQL, а также своим отсутствием в дистрибутиве Debian (Sarge), посему установка начинается с выкачивания программы на свой диск. Благо на сайте проекта `http://www.pgadmin.org/` можно найти сборки под множество дистрибутивов. Есть и специальный репозитарий для Debian — в `/etc/apt/source.list` добавляется строка:

```
deb [MIRROR URL]/pgadmin3/release/debian sarge pgadmin
```

Где вместо [MIRROR URL] подставляется одно из официальных зеркал PostgreSQL, например: `ftp://ftp.ru.postgresql.org/pub/mirrors/pgsql`, и производится установка программы:

```
> sudo apt-get update
> sudo apt-get install pgadmin3
> pgadmin3
```

При этом скачивается около 7.5 Мб. После запуска можно убедиться, что программа выглядит вполне солидно.

Новое подсоединение подключается через меню **File** → **Add server**. Требуется указать **Address** (localhost для локальной машины), сделать краткое описание соединения (**Description**), выбрать базу данных (**Maintenance DB**) и пользователя. После подсоединения доступны все объекты, которыми может управлять пользователь под которым произведено соединение.

PgAdmin III это продукт для администрирования и управления базами данных под управлением PostgreSQL и его потомков[1]. PgAdmin III содержит в себе графический интерфейс для управления данными, SQL-редактор с графическим представлением EXPLAIN, имеет инструменты для создания и редактирования таблиц,

[1] В следствии того, что PostgreSQL распространяется под BSD лицензией, имеется несколько коммерческих продуктов, основанных на его коде, например, EnterpriseDB, Pervasive Postgres и SRA PowerGres

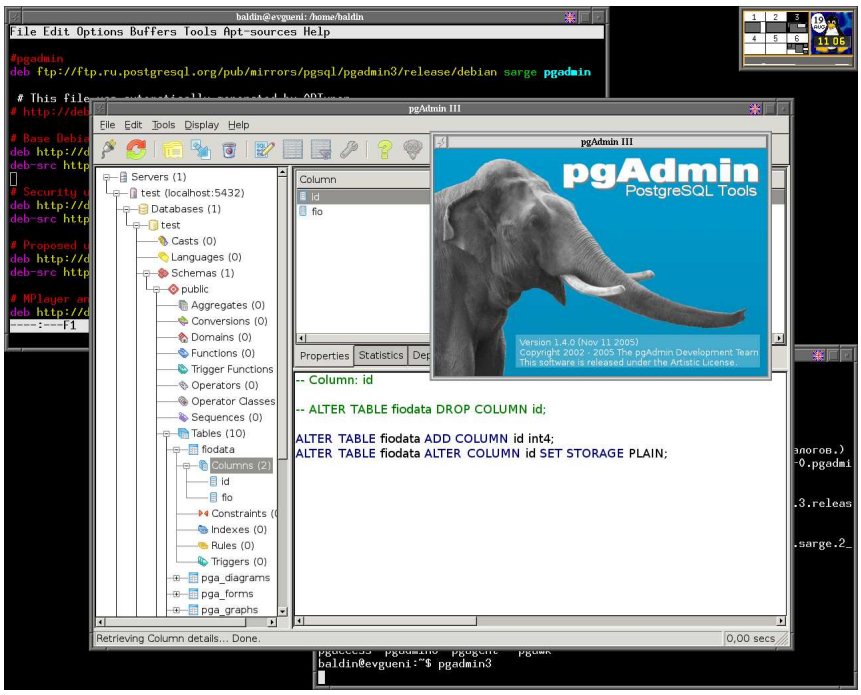

Рис. 2.3. pgAdmin III подробно объясняет что надо «сказать» чтобы создать выбранный объект.

умеет управляться с системой репликации Slony-I и многое другое, что действительно упрощает администрирование. И всё-таки PgAdmin III не для пользователя. Пока нет понимая в том, что происходит, не следует уповать на картинки.

Изначально, pgAdmin разрабатывался под «альтернативную» операционную систему, но на сегодня этот продукт является многоплатформенным решением, и работает не только под Linux, но и под Mac OSX, FreeBSD и даже Solaris. В качестве графической библиотеки при разработки pgAdmin III была выбрана wxWidgets (`http://www.wxwidgets.org`).

Русский перевод интерфейса, в принципе, существует, но на текущий момент не поддерживается. С другой стороны, это прежде всего инструмент администрирования и управления данными. Но в любом случае разработчики приветствуют новые и обновлённые переводы.

2.3.3 TOra

TOra возникла благодаря тому, что Генри Джонсон (Henrik Johnson) не смог запустить VMWare с альтернативной системой в 2000 году. В то же время ему хотелось иметь графическую утилиту для администрирования Oracle, подобную той, которой пользовались его друзья, так и не отошедшие от альтернативной операционной системы. TOra — это toolkit for Oracle. Так было, но на сегодня, в том числе и вследствие того, что TOra написана на библиотеки qt3, так же можно работать

и с PostgreSQL. Кроме PostgreSQL дополнительно поддерживается MySQL, и всё, что работает через ODBC.

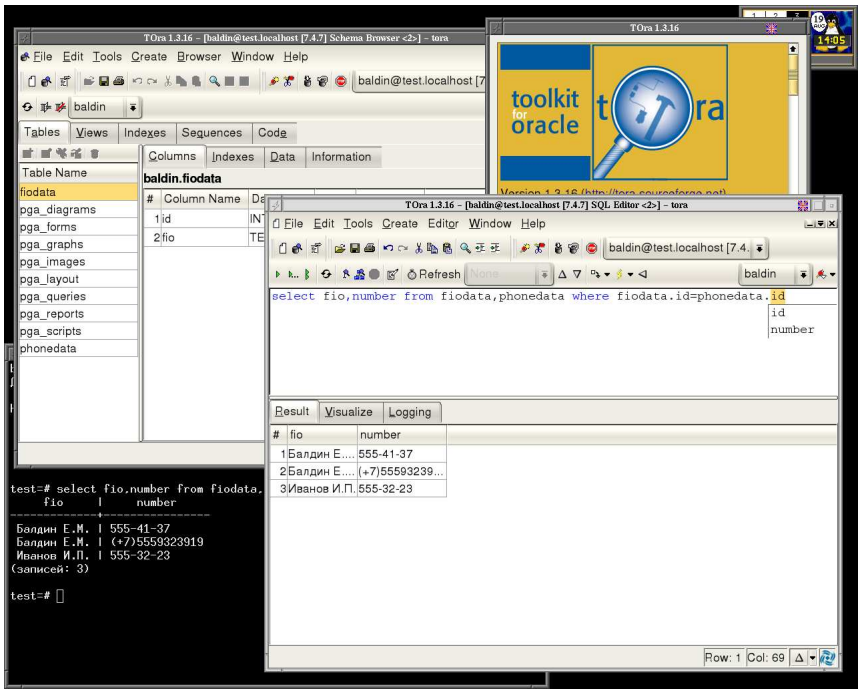

Рис. 2.4. tora знает всё об SQL и кое-что сверх того.

Установка и запуск TOra просты:

```
> sudo apt-get install tora
> tora
```

tora предлагает диалог создания нового соединения сразу при старте. Требуется указать **Connection provider** (PostgreSQL), Username, Host (localhost), Port (5432) и DataBase.

Вследствие своего прошлого многие возможности TOra привязаны к особенностям Orcale. В случае работы с PostgreSQL, TOra полезна прежде всего как браузер по SQL-объектам, SQL-терминал и изощрённый SQL-редактор. Как и в случае pgAdmin III TOra позволяет создавать и редактировать таблицы с помощью графических диалогов, но не владеет специфичными для PostgreSQL настройками.

TOra — это крепко «сбитый» программный продукт, который позволяет работать с разными реляционными СУБД в пределах одной программы.

2.3.4 OpenOffice и SDBC

Open Office — монстр, но ситуация на сегодня такая, что люди любят монстров, и ничего в обозримом будущем с этим не поделаешь. ☺

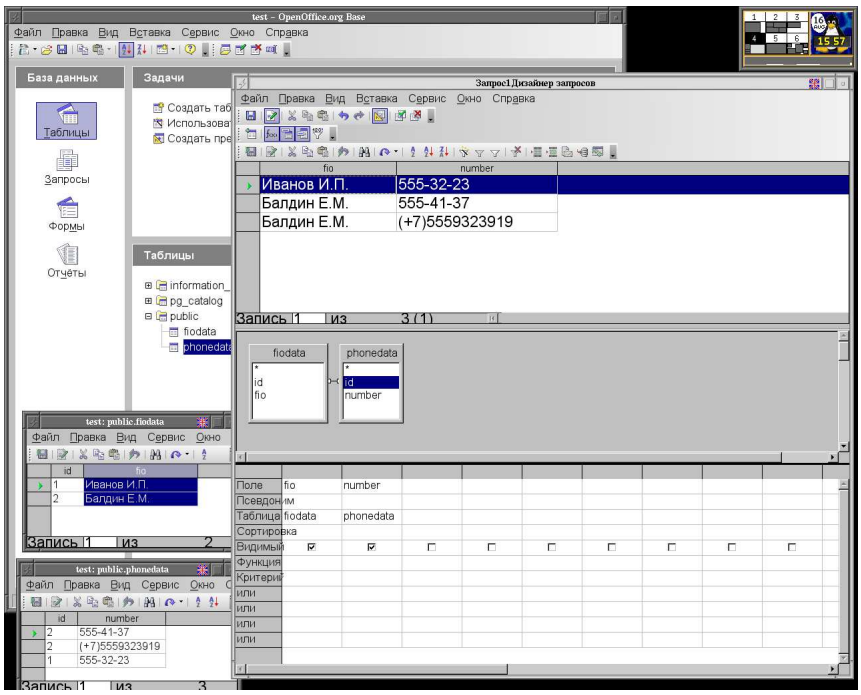

Рис. 2.5. OpenOffice + PostgreSQL

Для прямого доступа из OpenOffice к PostgreSQL без промежуточного уровня в виде ODBC/JDBC драйверов разрабатывается postgresql-sdbc драйвер. На сегодня в стандартной поставке OpenOffice этот пакет отсутствует.

Для установки необходимо скачать zip-архив этого драйвера с его домашней странички `http://dba.openoffice.org/drivers/postgresql/index.html` и положить куда-нибудь у себя на диске *не распаковывая (!)*. Далее, запустив OpenOffice, следует открыть диалог управления пакетами: **Сервис →Управление пакетами...** и с помощью кнопки «Добавить» установить этот пакет. В моём случае после установки пришлось перезапустить OpenOffice.

Для подсоединения к уже существующей базе данных PostgreSQL следует открыть диалог «Мастера базы данных»: **Создать → Базу данных → Выбор базы данных** → поставить галочку **Подключиться к существующей базе данных** → выбрать **postgresql**. Далее, при настройке соединения в следует ввести строчку вида:

---

dbname=**"имя␣БД"** host=**"адрес␣сервера"**

подставив вместо **"имя␣БД"** и **"адрес␣сервера"** базу данных, которая предварительно уже была создана и адрес сервера на котором «крутится» **postmaster**, например, **dbname=test host=localhost**. Далее, во вкладке «Аутентификация пользователя» необходимо ввести имя пользователя и можно протестировать соединение. Если тест прошёл нормально, то можно продолжить и выполнить соединение.

Во время окончания действия мастера предлагается сохранить всё что проделано в odb-файле (формат «База данных OpenDocument»). Затем это соединение можно будет выполнить простым открытием файла. Туда же сохраняется информация обо всех созданных формах, запросах и отчётах. Как конкретно создаются формы и отчёты — это совсем другая история и относится она не PostgreSQL, а к OpenOffice.

P.S. При выборе таблиц видно, что они в PostgreSQL разбиты на группы. Пользовательские таблицы по умолчанию находятся в группе **public**. В группах **pg_catalog** и **informaion_schema** представлена системная информация и статистика.

2.4 Что выбрать?

Естественно, рассмотрены далеко не все возможные программы общего назначения для работы с PostgreSQL, но даже из того что рассмотрено нельзя выбрать что-то одно. Каждая программа имеет свои особенности и преимущества. **psql** позволяет легко работать удалённо, OpenOffice удовлетворяет нашу любовь к монстрам, PgAdmin III содержит множество подсказок по делу, PgAccess удивляет своей интеграцией с TCL/Tk, а TOra — «красивая» ☺.

Я всегда выбирал **psql**, но это скорее всего связано со специфичностью решаемых мной задач. Я вполне могу представить себе ситуацию, когда наличие, например, TOra значительно облегчит жизнь. Ну и не следует забывать про данные, которые кто-то должен вводить. Если лень писать специальную программку, которую по хорошему лучше таки написать, то OpenOffice поможет, особенно если вводить не Вам. ☺