PostgreSQL: Уровень 1. Основы SQL

Преподаватель:

Валентин Степанов

Модуль 7. Перекрестные запросы (выборка данных из нескольких таблиц)

- Структура перекрестного запроса
- Опция INNER JOIN
- Опция OUTER JOIN
- Соединение таблицы с самой собой
- Подзапросы
- Объединение результирующих множеств

Соединение таблиц

- T1 тип_соединения T2 [условие_соединения]
- Перекрёстное соединение
  - T1 CROSS JOIN T2
- Соединения с сопоставлениями строк
  - T1 { [INNER] | { LEFT | RIGHT | FULL } [OUTER] } JOIN T2 ON логическое_выражение
  - T1 { [INNER] | { LEFT | RIGHT | FULL } [OUTER] } JOIN T2 USING (список столбцов соединения)
  - T1 NATURAL { [INNER] | { LEFT | RIGHT | FULL } [OUTER] } JOIN T2

Соединение таблицы с самой собой

- Декартово произведение
- Соединение таблиц на основе неравенства значений атрибутов
- Явное использование декартова произведения таблиц

Внешние соединения

- Левое внешнее соединение (LEFT JOIN)
- Правое внешнее соединение (RIGHT JOIN)
- Полное внешнее соединение (FULL JOIN)

Сочетание запросов

- запрос1 UNION [ALL] запрос2
- запрос1 INTERSECT [ALL] запрос2
- запрос1 EXCEPT [ALL] запрос2

Подзапросы

- Скалярный подзапрос
- Подзапрос в предикате IN
- Некоррелированный
- Предикат EXISTS
- Коррелированный (связанный) подзапрос
- Подзапрос в предложении SELECT
- Подзапрос в предложении FROM
- Подзапрос в предложении HAVING
- Вложенные подзапросы
- Общее табличное выражение

Модуль 8. Модификация данных в СУБД

- Добавление строк в таблицу при помощи команды INSERT
- Использование команды UPDATE для изменения строк таблицы
- Удаление данных из таблицы при помощи команды DELETE
- Понятие транзакции

Вставка строк в таблицы

- INSERT
- ON CONFLICT
- RETURNING
- COPY

Обновление строк в таблицах

- UPDATE
- RETURNING

Удаление строк из таблиц

- DELETE
- USING
- RETURNING
- TRUNCATE

Транзакции и их свойства

- **Транзакция** – множество операций, которые переводят базу данных из одного корректного состояния в другое корректное при условии, что транзакция выполнена полностью и без помех со стороны других транзакций
- **Согласованность** определяется ограничениями целостности и семантикой приложения
- **Атомарность** – транзакция либо выполняется полностью, либо не оставляет никаких следов
- **Изоляция** – защита от нарушения корректности (аномалий) при конкурентном выполнении транзакций

Блокировки

- Задача: упорядочение конкурентного доступа к разделяемым ресурсам
- Механизм
  - перед обращением к данным процесс захватывает блокировку
  - после обращения — освобождает (обычно в конце транзакции)
  - несовместимые блокировки приводят к очередям

Модуль 9. Подключение к СУБД из прикладной программы

- Понятие клиентской библиотеки
- Основные виды клиентских библиотек
- Понятие объектно-реляционного соответствия

Расширение возможностей PostgreSQL

- PL/pgSQL
- Расширяемость типов
- Языки программирования
- Фоновые процессы
- Пул соединений
- Внешние данные
- Клиентские приложения

Процедурные языки

- Назначение
  - дополнить SQL процедурными возможностями
- Создание
  - CREATE [TRUSTED] LANGUAGE имя
    HANDLER обработчик_вызова
    [INLINE обработчик_внедренного_кода]
    [VALIDATOR функция_проверки];
  - установка из расширения
- Доступные языки
  - в дистрибутиве: PL/pgSQL, PL/Tcl, PL/Perl, PL/Python
  - дополнительно: PL/Java, PL/V8, PL/R, PL/PHP, ...

Доверенные языки

- Доверенные
  - ограниченная функциональность в части взаимодействия с ОС
  - по умолчанию доступ для всех пользователей
- Недоверенные
  - полная функциональность языка
  - доступ только у суперпользователей
  - обычно к названию языка добавляют «u»: plperl → plperlu

Преимущества

- Производительность
  - меньше команд
  - меньше пересылки данных
  - меньше разборов команд
- API к объектам БД
  - гибче управлять согласованностью данных
  - гибче управлять доступом к данным
- Работа с внешними данными

PL/pgSQL

- Появился в версии 6.4 в 1998 году
  - устанавливается по умолчанию с версии 9.0
- Цели создания языка
  - используется для написания функций и триггеров
  - добавляет управляющие структуры к языку SQL
  - позволяет использовать любые пользовательские типы, функции и операторы
  - доверенный язык
  - простота использования
- На основе Oracle PL/SQL

Структура блока

- Блочно-структурированный язык

  [<<метка>>]
  [DECLARE
      объявления]
  BEGIN
      операторы
  [EXCEPTION
      обработка_ошибок]
  END [метка];

- Важно:
  - BEGIN — начало исполняемой секции в PL/pgSQL-блоке
  - BEGIN; — команда SQL, открывающая транзакцию

Расширяемость типов

- Богатый набор существующих типов
  - числа, строки, даты, слабоструктурированные, геометрия, ...
  - массивы
- Средства для создания новых типов
  - простые: на основе уже имеющихся типов
  - сложные: новый тип с нуля

Составные типы

- Набор именованных атрибутов (табличная строка)
- Создание нового типа
  - вручную: CREATE TYPE AS
  - автоматически при создании отношения

Типы перечислений

- Упорядоченный набор значений
  - значения выглядят, как текстовые строки
  - хранятся как 4-байтовые числа (oid)
- Создание нового типа
  - только вручную: CREATE TYPE AS ENUM

Диапазонные типы

- Интервал значений от начальной до конечной точки
  - открытый (>, <), закрытый (≤, ≥), неограниченный
- Создание нового типа
  - CREATE TYPE AS RANGE
  - уже есть для integer, bigint, numeric, timestamp, date
- Операции
  - сравнение
  - проверка на включение диапазона или элемента, на пересечение
  - объединение, пересечение, вычитание
  - и др.

Интервалы дат и времени

- Интервал значений определенной длины
- Операции
  - арифметика дат и времени

Домены

- Ограничение допустимых значений существующего типа
  - ограничение NOT NULL
  - значение по умолчанию DEFAULT
  - проверка CHECK
- Создание нового типа
  - только вручную: CREATE DOMAIN

Базовые типы

- Тип, не сводящийся к комбинации существующих
- Создание нового типа
  - CREATE TYPE, требует программирования на языке Си
  - множество типов предоставляются расширениями

Действия над типами

- Функции, принимающие или возвращающие значения типа
- Приведение типов
  - преобразование значения одного типа к другому типу
- Операторы
  - унарные (префиксные и постфиксные)
  - бинарные (инфиксные)

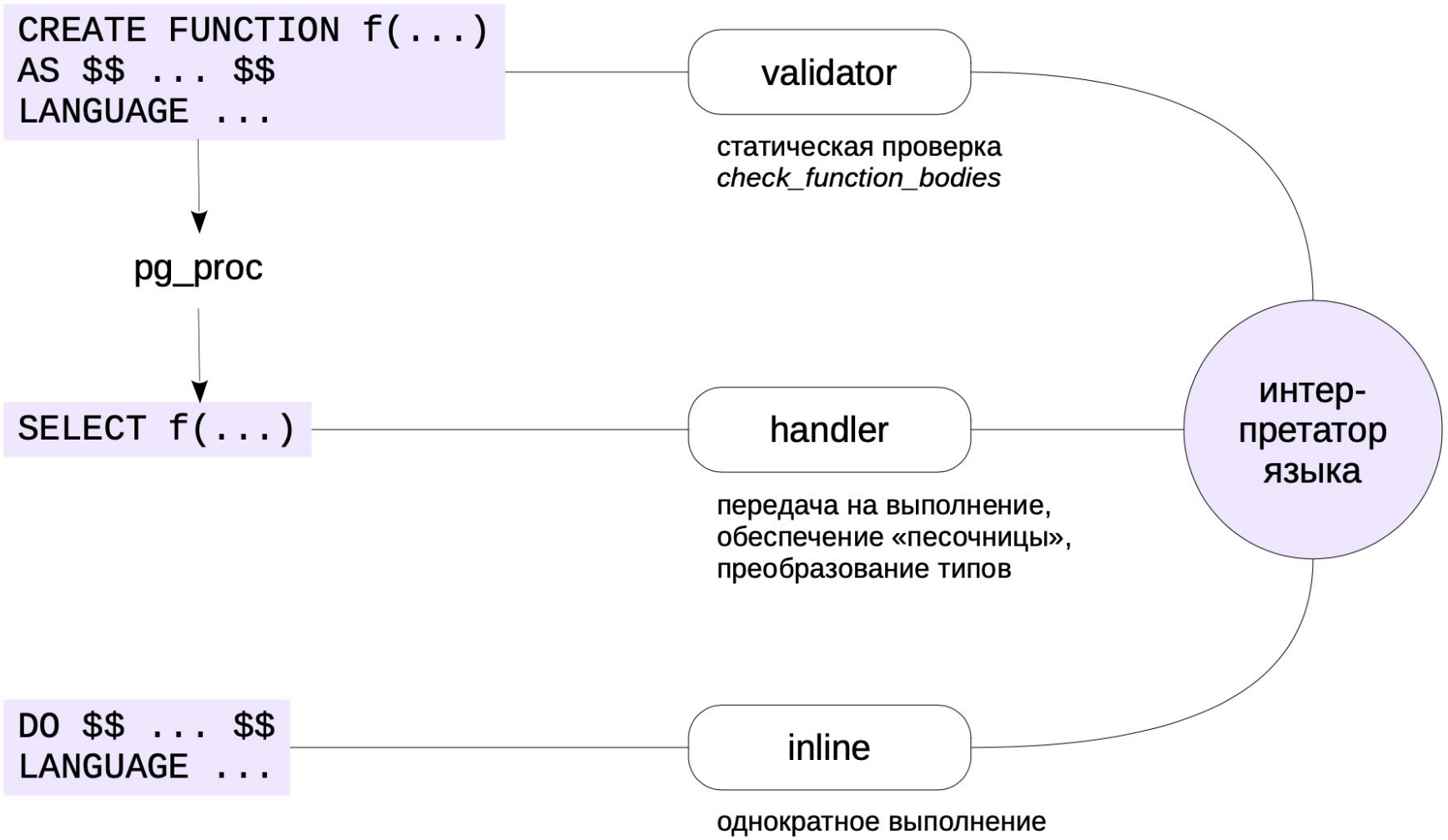
Языки программирования

- «Встроенные»
  - C
  - SQL
- Стандартные (поддержка сообщества)
  - PL/pgSQL
  - PL/Perl, PL/Python, PL/Tcl
- Сторонние
  - PL/Java, PL/V8, PL/R, PL/Lua, ...

Доверенные языки

- Доверенные
  - гарантируют работу пользователя в рамках выданного доступа
  - ограничено взаимодействие с окружением и внутренностями СУБД, язык должен позволять работать в «песочнице»
  - по умолчанию доступ для всех пользователей
- Недоверенные
  - доступна полная функциональность языка
  - доступ только у суперпользователей
  - обычно к названию языка добавляют «u»: plperl → plperlu

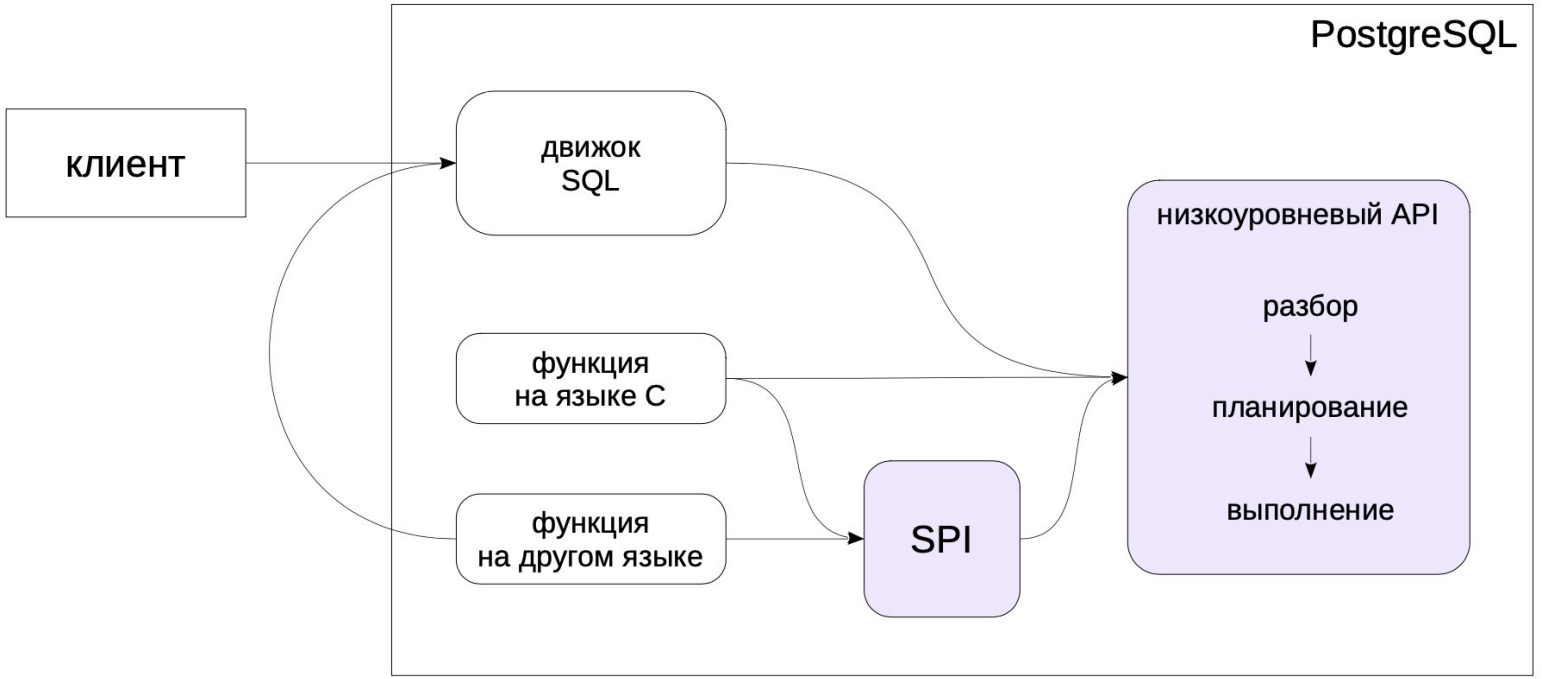
Подключение нового языка



Задачи

- Обработка информации, хранимой в базе данных
  - подпрограмма всегда выполняется в контексте подключения к БД
  - PL/pgSQL интегрирован с SQL
  - для остальных — интерфейс серверного программирования (SPI)
- Вычисления, не связанные с базой данных
  - возможности PL/pgSQL сильно ограничены
  - эффективность
  - удобство использования
  - наличие готовых библиотек
  - специализированные задачи

Интерфейс SPI



Из чего выбирать

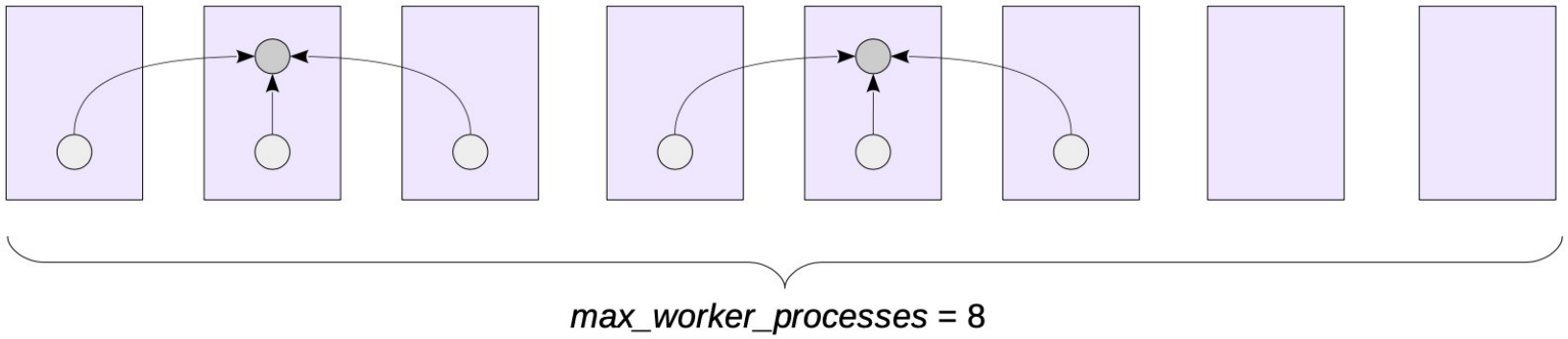
- Интенсивная работа с данными
  - SQL, PL/pgSQL
- Проверенные, часто используемые
  - PL/Perl, PL/Python, PL/V8, PL/Java
- Другие языки общего назначения
  - PL/Go, PL/Lua, ...
- Специальные задачи
  - PL/R (статистика), PL/Proxy (удаленные вызовы), ...
- Максимальная эффективность
  - C

Фоновые процессы

- Фиксированный набор служебных процессов
  - postmaster
  - walwriter
  - checkpointer
  - autovacuum
  - и другие
- Динамически порождаемые фоновые процессы
  - контролируются процессом postmaster
  - имеют доступ к разделяемой памяти
  - могут устанавливать внутреннее соединение с базами данных
  - реализуются на языке C

Использование

- Параллельное выполнение запросов
  - max_parallel_workers = 8
- Параллельное выполнение служебных команд
  - max_parallel_maintenance_workers = 2
- Логическая репликация
  - max_logical_replication_workers = 4



Прикладные задачи

- Некоторые идеи
  - планировщик заданий внутри СУБД
  - асинхронная обработка событий
  - распараллеливание сложной обработки данных
  - автономные транзакции
  - и другое
- Но пользовательские задачи неудобно писать на языке C

Расширение dblink

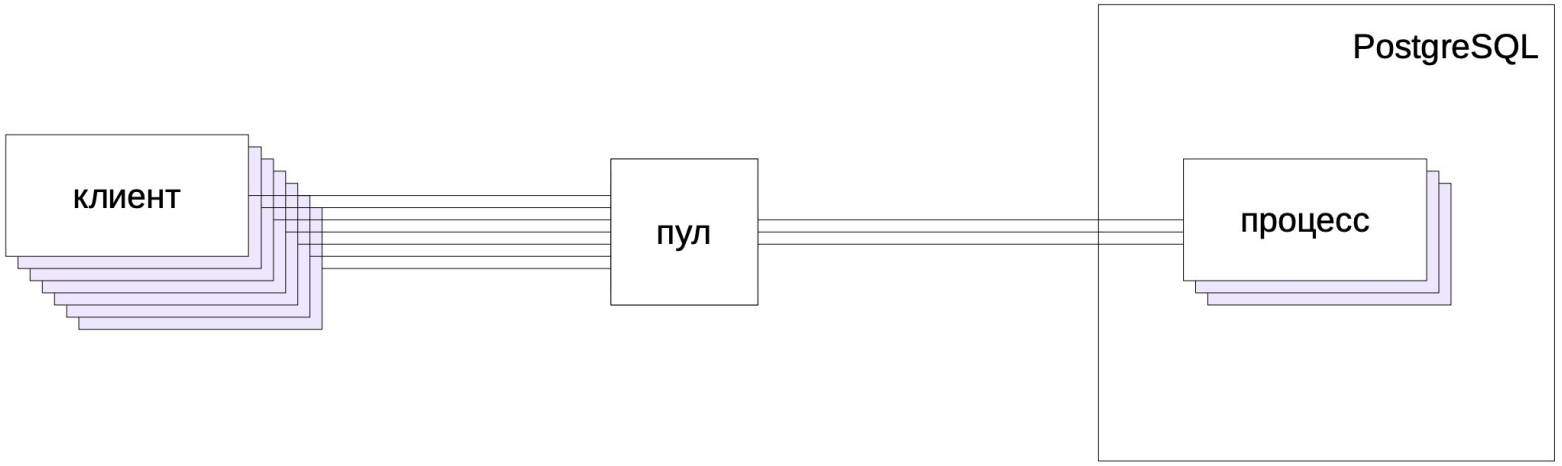
- Назначение
  - выполнение произвольных SQL-команд на удаленном сервере
  - в качестве удаленного сервера может выступать и локальный
- Плюсы
  - стандартное расширение
- Минусы
  - фоновые процессы не используются
  - устанавливается новое соединение

Расширение pg_background

- Назначение
  - возможность выполнить команду SQL в фоновом процессе
  - команда может вызвать подпрограмму на любом серверном языке
- Плюсы
  - используются фоновые процессы
- Минусы
  - стороннее расширение
- https://github.com/vibhorkum/pg_background

Пул соединений

- Обычно большую часть времени соединение простаивает
- Менеджер пула соединений
  - открывает и удерживает несколько соединений с сервером
  - для клиентов выглядит как сервер PostgreSQL,
    для сервера PostgreSQL выглядит как клиент

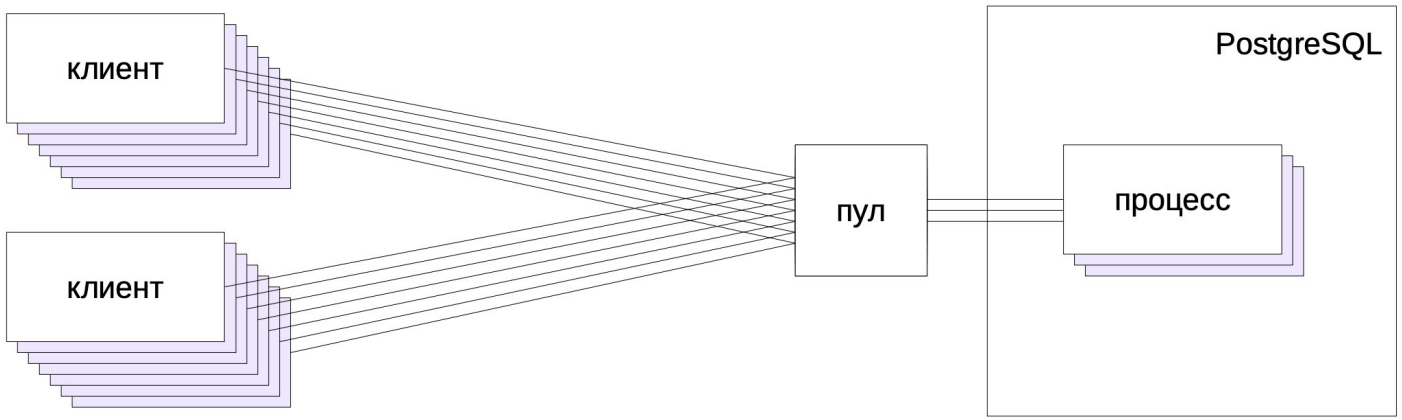


Место в архитектуре

- На сервере приложения
  - клиент использует быстрое локальное подключение
  - подключение к серверу устанавливается один раз и переиспользуется

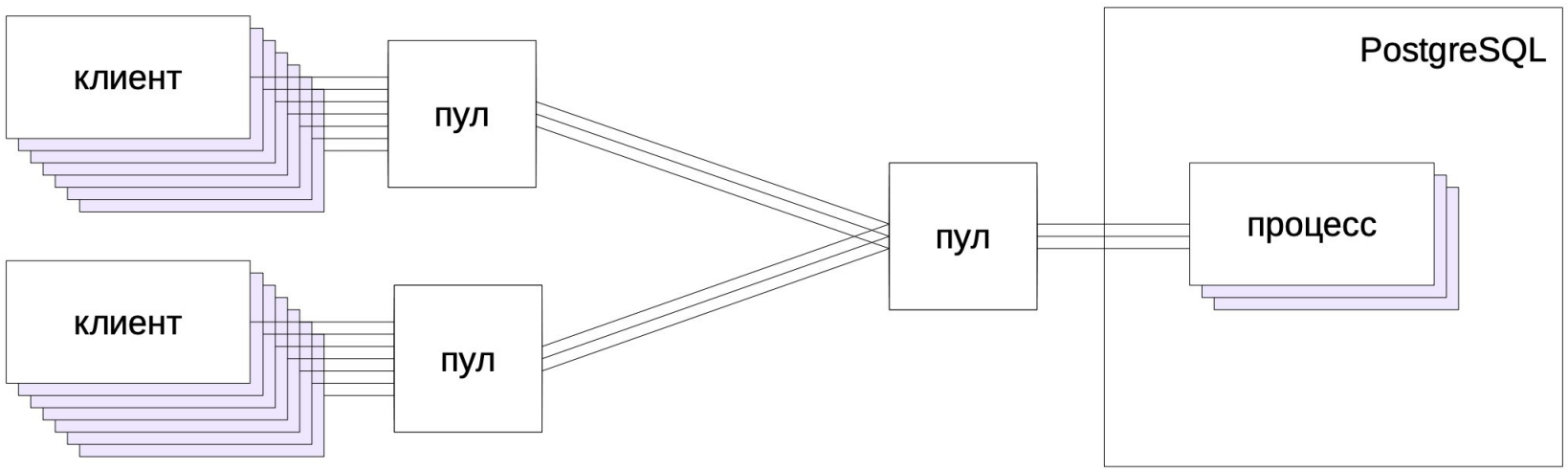
Место в архитектуре

- На сервере баз данных
  - если используется несколько серверов приложений
- Встроенного пула соединений нет



Место в архитектуре

- И на серверах приложений, и на сервере баз данных
  - полезные свойства комбинируются
  - накладные расходы растут, но обычно невелики



Доступные продукты

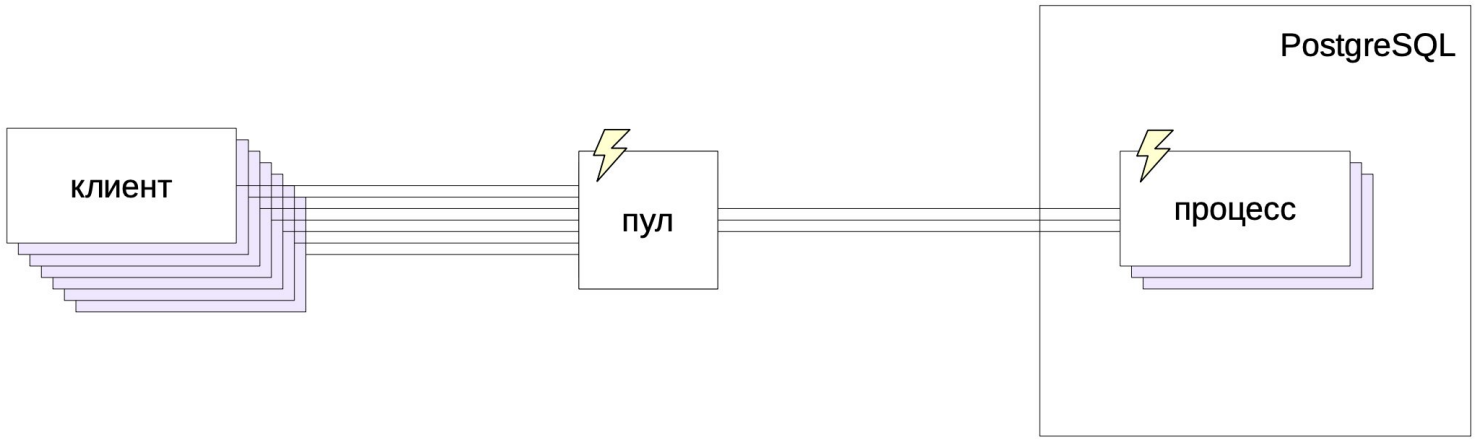
- PgBouncer
  - стандарт де-факто
- Pgpool-II
  - не только пул, но и балансировка нагрузки
- Odyssey
  - многопоточный, высокопроизводительный
- и др.

Режимы работы

- «Пул сеансов»
  - может иметь смысл только для коротких сеансов
- «Пул транзакций»
  - разные транзакции одного клиентского сеанса могут выполняться в разных соединениях с сервером
  - основной режим
  - есть особенности, которые надо учитывать при разработке
- «Пул операторов»
  - запрещает транзакции, состоящие более чем из одного оператора
- Новый пул соединений создается для каждой пары (база данных, роль)

Аутентификация

- Менеджер пула аутентифицирует соединения клиентов, сервер баз данных аутентифицирует соединения пула
  - требуется специальная настройка
  - можно использовать файл формата pg_hba.conf



Управление

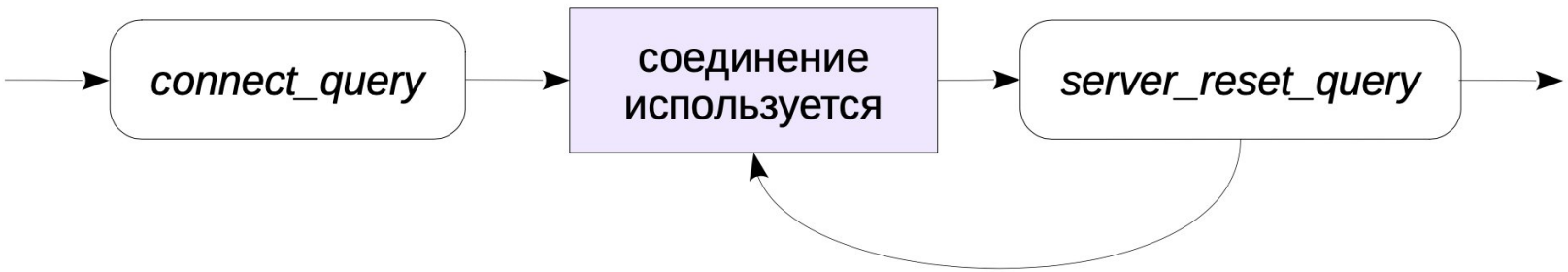
- Консоль управления PgBouncer
  - подключение psql к псевдобазе pgbouncer
  - SHOW — ряд команд для вывода состояния и другой информации
  - PAUSE — приостановка клиентов (для перезагрузки базы данных)
  - RESUME — возобновление работы
  - RELOAD — перечитывание файла конфигурации
  - и другие возможности

Особенности

- Действия, локализованные в сеансе, а не в транзакции
  - подготовленные операторы (PREPARE/EXECUTE)
  - временные таблицы (кроме ON COMMIT DROP)
  - функция currval
  - установка конфигурационных параметров (SET/RESET)
  - рекомендательные блокировки на уровне сеанса межпроцессные уведомления (LISTEN/NOTIFY)
  - курсоры с фразой WITH HOLD
  - загрузка разделяемых библиотек (LOAD)
  - расширения, сохраняющие состояние на уровне сеанса

Подготовка соединений

- Начальные действия и изоляция



Внешние данные

- Представление внешних данных как обычных таблиц
  - стандарт ISO/IEC 9075-9 (SQL/MED)
  - поддерживаются SELECT, INSERT, UPDATE, DELETE
  - триггеры
  - обычное разграничение доступа (GRANT, REVOKE)
- Способ миграции данных из других СУБД
- В будущем — механизм для шардинга
  - вместе с секционированием

Компоненты

1. Обертка сторонних данных
   - каков тип внешнего источника данных?
   - postgres_fdw: внешний сервер PostgreSQL

Компоненты

1. Обертка сторонних данных
   - каков тип внешнего источника данных?
2. Внешний сервер
   - как подключиться к внешнему источнику?
   - postgres_fdw: узел, порт, база данных

Компоненты

1. Обертка сторонних данных
   - каков тип внешнего источника данных?
2. Внешний сервер
   - как подключиться к внешнему источнику?
3. Сопоставление ролей
   - как локальные роли связаны с разграничением доступа, используемым внешним источником?
   - postgres_fdw: и там, и там — обычные роли PostgreSQL

Компоненты

1. Обертка сторонних данных
   - каков тип внешнего источника данных?
2. Внешний сервер
   - как подключиться к внешнему источнику?
3. Сопоставление ролей
   - как локальные роли связаны с ролями внешнего источника?
4. Внешние таблицы
   - какова структура внешних данных?
   - postgres_fdw: данные в обычных таблицах

Соединения и транзакции

- Управление соединениями
  - открывается при первом обращении к внешним данным
  - остается открытым до конца сеанса
- Управление удаленными транзакциями
  - фиксируется или прерывается автоматически, когда фиксируется или прерывается локальная транзакция
  - Repeatable Read для локальных транзакций уровней Read Committed и Repeatable Read
  - Serializable для локальных транзакций Serializable
  - согласованность не гарантируется (нет двухфазной фиксации)

Сравнение с dblink

	postgres_fdw	dblink
доступные команды SQL	SELECT, INSERT, UPDATE, DELETE	любые
совмещение в запросе локальных и внешних данных	да	сложно
управление соединением	автоматическое	ручное
управление транзакциями	автоматическое	ручное
глобальные транзакции	нет	нет
стандартизованный способ	да	нет

Расширение file_fdw

1. Обертка сторонних данных
   - тип внешнего источника — текстовый файл с разделителями
2. Внешний сервер
   - как подключиться — не требуется дополнительной информации
3. Сопоставление ролей
   - не имеет смысла, т. к. во внешнем источнике нет ролей
4. Внешние таблицы
   - структура данных — описание полей файла в виде столбцов таблицы

Другие доступные обертки

- Базы данных
  - ODBC, JDBC
  - различные реляционные и NoSQL СУБД
- Файлы различных форматов
- Геоданные и научные данные
- Веб-ресурсы
  - стандартные протоколы (RSS, IMAP, WWW...)
  - для различных сайтов
- Можно написать собственную обертку
  - на C или Python (с помощью расширения Multicorn)

PostgreSQL для приложения

- Отдельный пользователь
  - CREATE USER
- Удаленное подключение
  - listen_addresses = '*' в postgresql.conf
- Библиотека для подключения
  - libpq – штатная библиотека
  - php-pgsql – PHP
  - libdbd-pg-perl – Perl
  - python3-psycopg2 – Python
  - libpostgresql-jdbc-java – Java
  - pg – Node.js

Объектно-реляционное отображение (ORM)

- Изоляция БД
- Изоляция объектной модели данных приложения
- Абстрагирование от физической структуры БД