PL/pgSQL
Отладка и профилирование

Павел Лузанов
p.luzanov@postgrespro.ru

Предисловие

Аудитория

- разработчики серверной части приложений, PL/pgSQL

Обзор решений

- встроенные в PostgreSQL
- сторонние
- без философии и фанатизма

~~Поиск серебряной пули~~

Темы

- Отладка хранимых процедур
  - Отладчик
  - Служебные сообщения
- Трассировка и профилирование
  - Журнал сервера
  - Расширение PL Profiler
  - Расширение pg_stat_statement

Проверка кода PL/pgSQL

При создании

`CREATE FUNCTION, DO` – только синтаксис

check_function_bodies – включает/отключает все проверки при создании

Дополнительные проверки

plpgsql.extra_warnings – "shadowed_variables"

plpgsql.extra_errors – пока нет

После создания

Расширение plpgsql_check – наличие объектов в БД

Способы отладки

Использование отладчика

Служебные сообщения в коде

PL Debugger

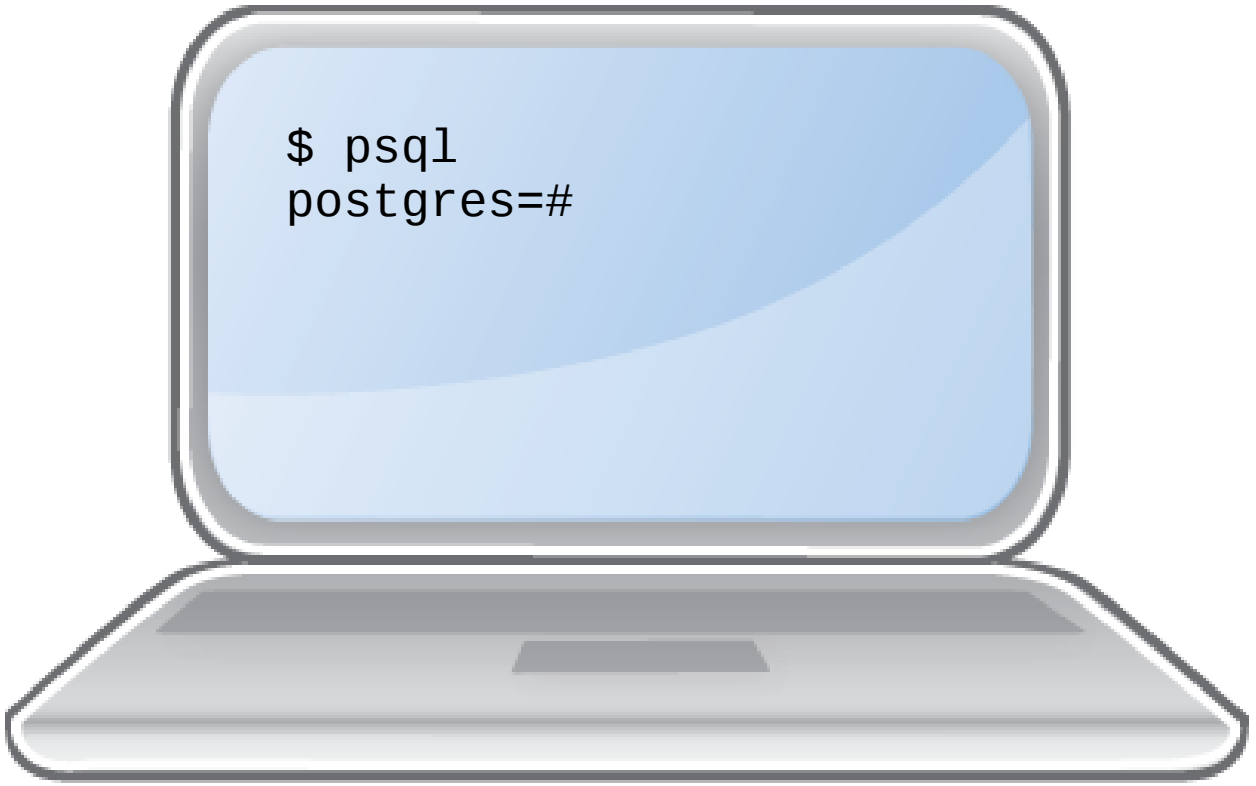
Состав

- Расширение pldbgapi
- Графический интерфейс: PgAdmin, пр.

Возможности

- Не требуется изменение кода
- Отдельная сессия для отладчика
- Установка точек сохранения
- Пошаговое выполнение
- Проверка и установка значений переменных

Демонстрация



Служебные сообщения

Задачи

- Отладка кода
- Мониторинг долгих процессов
- Журнал приложения

Подходы к реализации

- RAISE
- Межпроцессное взаимодействие
- Запись в таблицу
- Запись в файл

RAISE

1. Добавление в код отладочных сообщений

```
CREATE FUNCTION get_count
    (tabname text) RETURNS bigint
AS $$
DECLARE
    cmd text;
    retval bigint;
BEGIN
    cmd := 'SELECT COUNT(*) FROM '
        || quote_ident(tabname);
    RAISE NOTICE 'cmd: %', cmd;
    EXECUTE cmd INTO retval;
    RETURN retval;
END;
$$ LANGUAGE plpgsql;
```

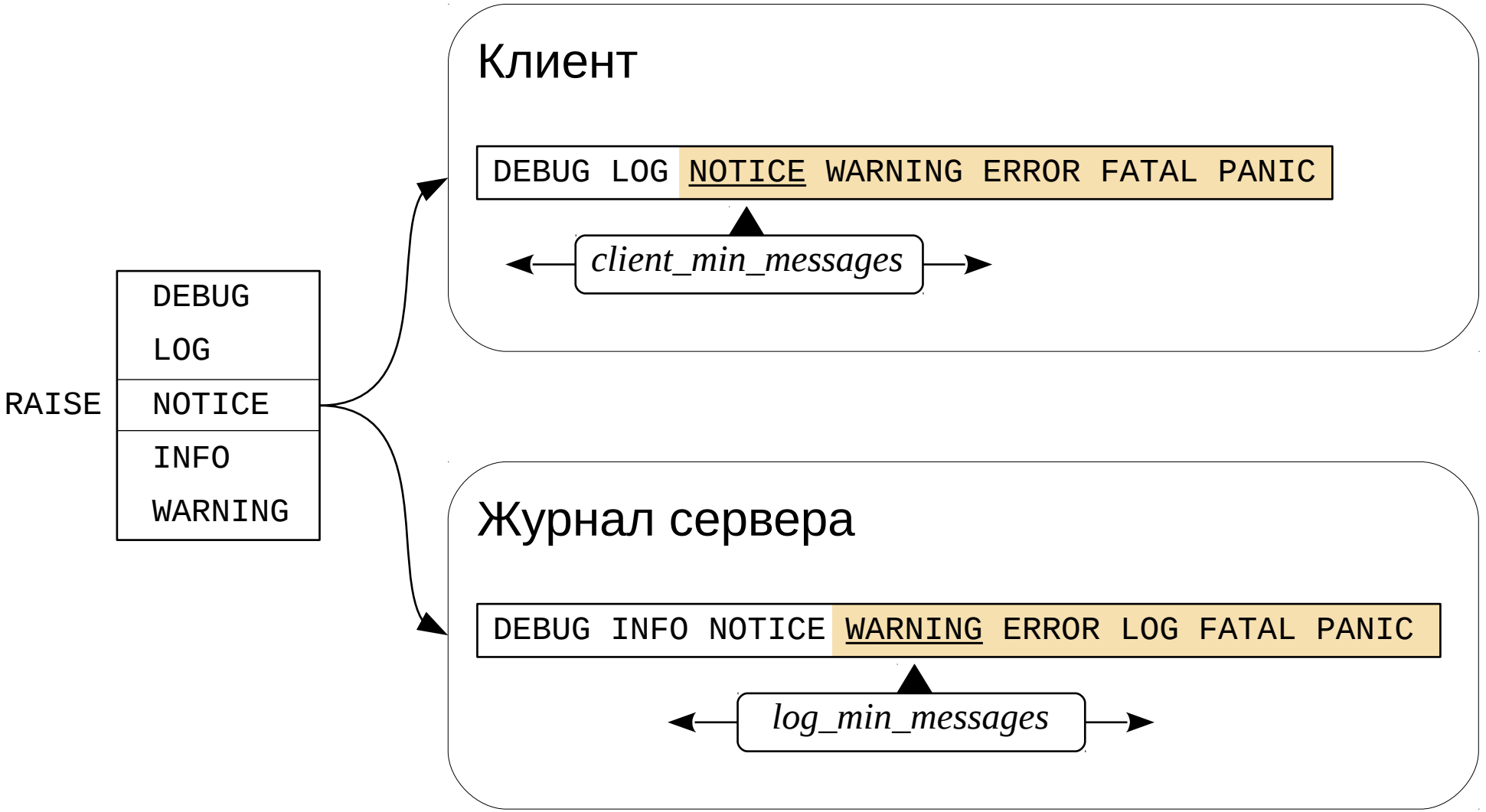
2. Вызов функции

3. Получение сообщений по ходу выполнения

```
psql=> SELECT get_count('pg_class');

NOTICE:  cmd: SELECT COUNT(*) FROM pg_class
 get_count
-----------
       334
(1 row)
```

RAISE


Клиент
DEBUG LOG NOTICE WARNING ERROR FATAL PANIC
client_min_messages
RAISE
DEBUG
LOG
NOTICE
INFO
WARNING
Журнал сервера
DEBUG INFO NOTICE WARNING ERROR LOG FATAL PANIC
log_min_messages

Процесс → Процесс

NOTIFY / LISTEN

- встроенные команды сервера
- транзакционные команды

Реализация dbms_pipe

- стороннее расширение ORAFCE

Статус сеанса (SET application_name)

- встроенное решение: SET application_name / SELECT … pg_stat_activity
- ограниченное применение

Процесс → Таблица

Расширение dblink

- входит в состав сервера
- накладные расходы на создание соединения

Автономные транзакции

- коммерческий дистрибутив (Postgres Pro Enterprise)

Процесс → Файл

Расширение pgadminpack

- входит в состав сервера
- файлы внутри PGDATA

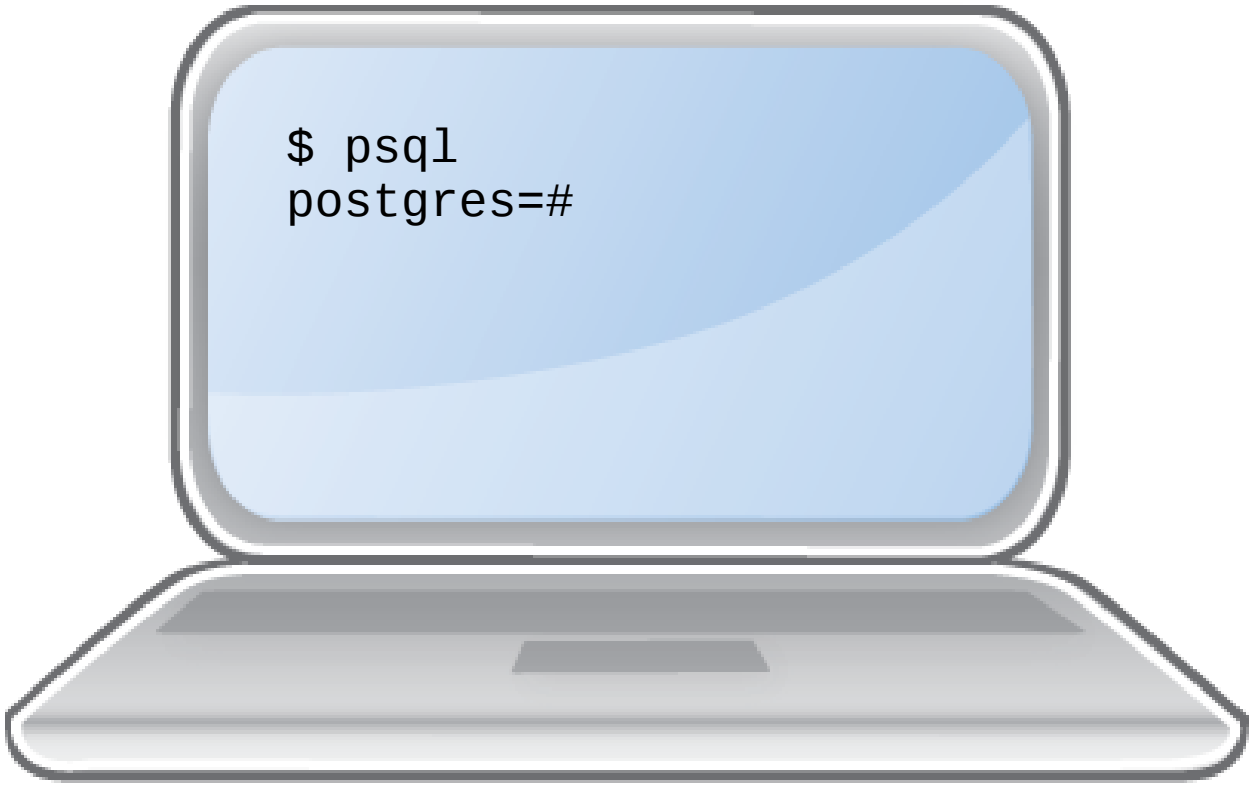
Реализация utl_file

- стороннее расширение ORAFCE

Недоверенные языки

- например, plperlU

Демонстрация



Профилирование

Задачи

- Разбить задачу на части (вплоть до операторов)
- Собрать статистику: продолжительность, количество повторений

Оптимизация

- Отдельные запросы
- Изменение кода, архитектуры решения

Инструменты

- Журнал сервера
- PL Profiler
- pg_stat_statements

Журнал сервера

Параметры конфигурации

```
log_statement                   superuser
log_min_duration_statement      superuser
...
```

Установка

`postgresql.conf`

`ALTER SYSTEM` (`postgresql.auto.conf`)

`ALTER DATABASE … SET` / `ALTER ROLE … SET` — для новых сеансов

`SET` / `set_config()` — во время сеанса

Обновление

`pg_ctl reload`

`pg_reload_conf()`

`kill -HUP` *pid*

Сценарий 1

Задача

Трассировка *всех* процессов включенная *постоянно*

Настройка

(1) postgresql.conf / alter system

(2) обновление конфигурации

Сценарий 2

Задача

Трассировка *всех* процессов
за *интервал* времени

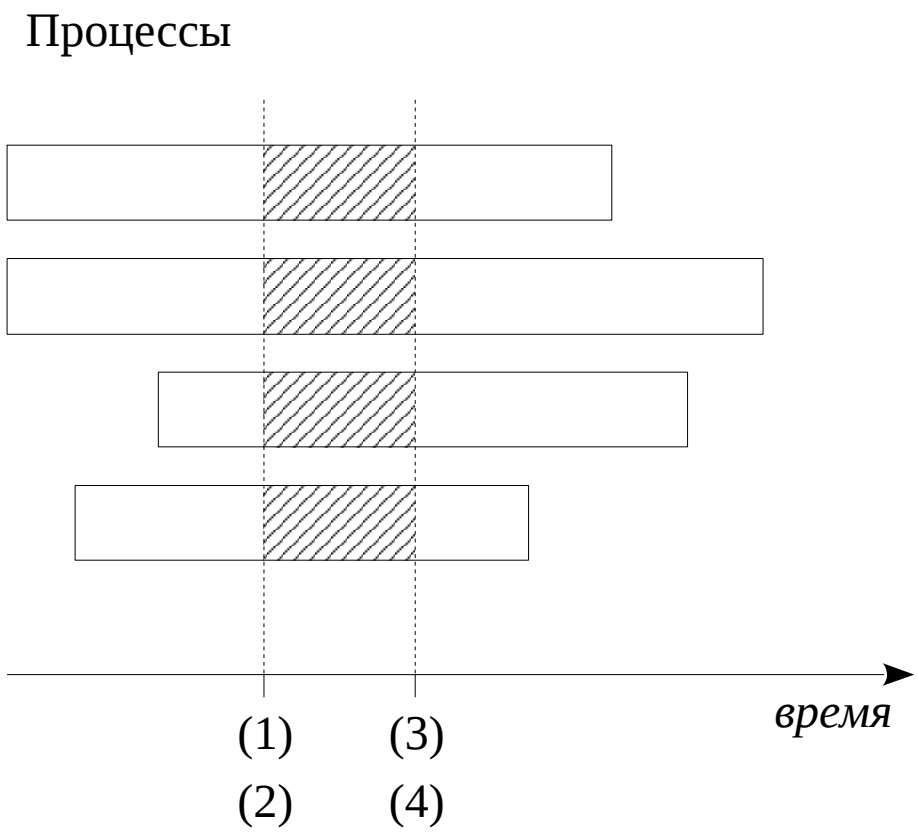
Настройка

(1) вкл: postgresql.conf / alter system

(2) обновление конфигурации

(3) выкл: postgresql.conf / alter system

(4) обновление конфигурации



Сценарий 3

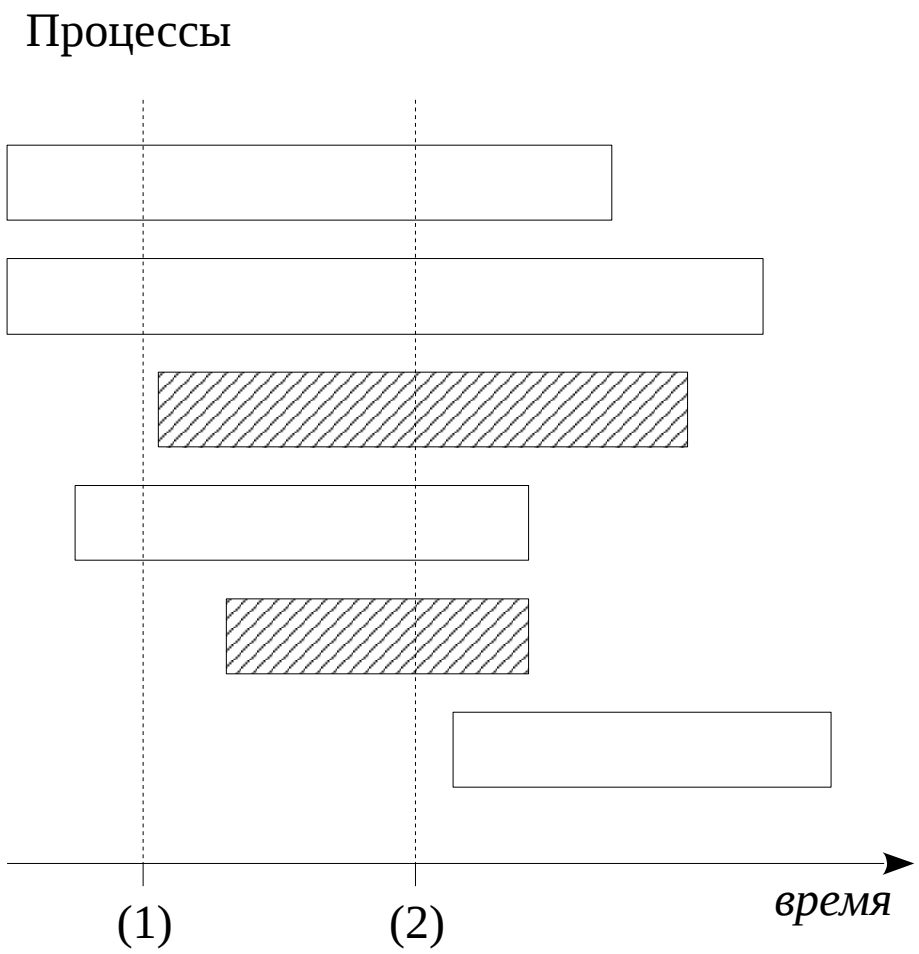
Задача

Трассировка *новых* процессов

Настройка

(1) `ALTER DATABASE … SET`
или `ALTER ROLE … SET`

(2) `ALTER DATABASE … RESET`
или `ALTER ROLE … RESET`



Сценарий 3

Приоритет установки параметров

1. `SET ...`
2. `ALTER DATABASE / ROLE ... SET ...`
3. `ALTER SYSTEM SET …`
4. `postgresql.conf`

Следствия

Трассировка выделенных пользователей:

```
ALTER SYSTEM SET log_statement = 'none';
ALTER ROLE app_admin log_statement = 'all';
```

Трассировка всех, кроме:

```
ALTER SYSTEM SET log_statement = 'all';
ALTER ROLE deamon log_statement = 'none';
```

Сценарий 4

Задача

Трассировка *заданного* процесса
за *интервал* времени
по инициативе *администратора*

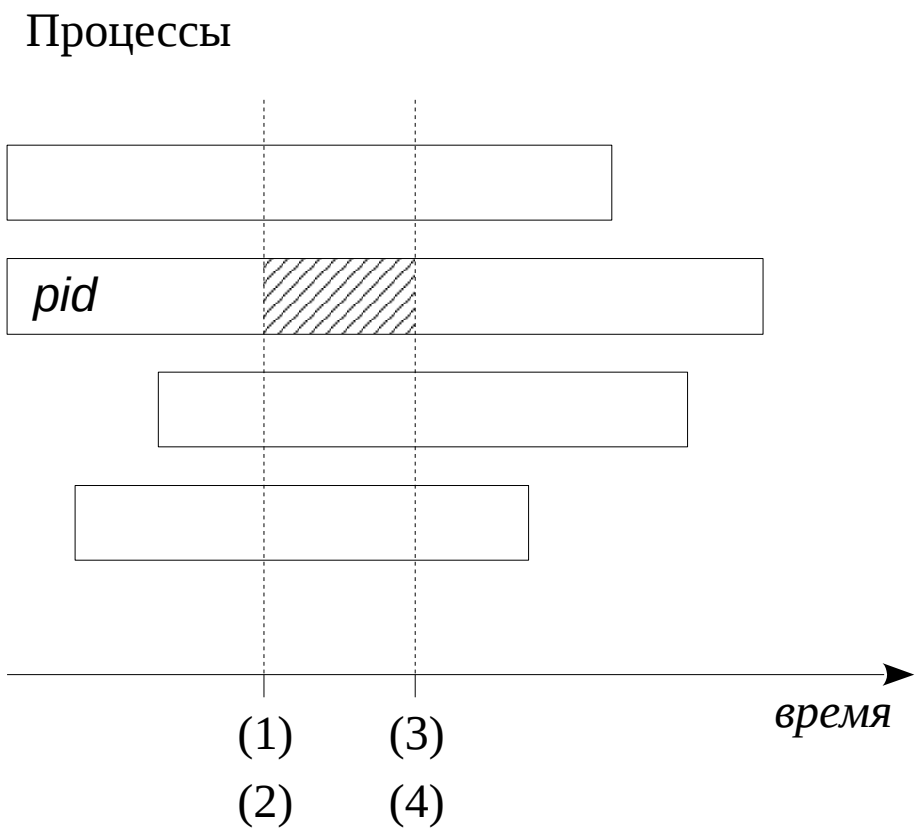
Настройка

(1) вкл: postgresql.conf / alter system

(2) `kill -HUP` *`pid`*

(3) выкл: postgresql.conf / alter system

(4) `kill -HUP` *`pid`*



Сценарий 5

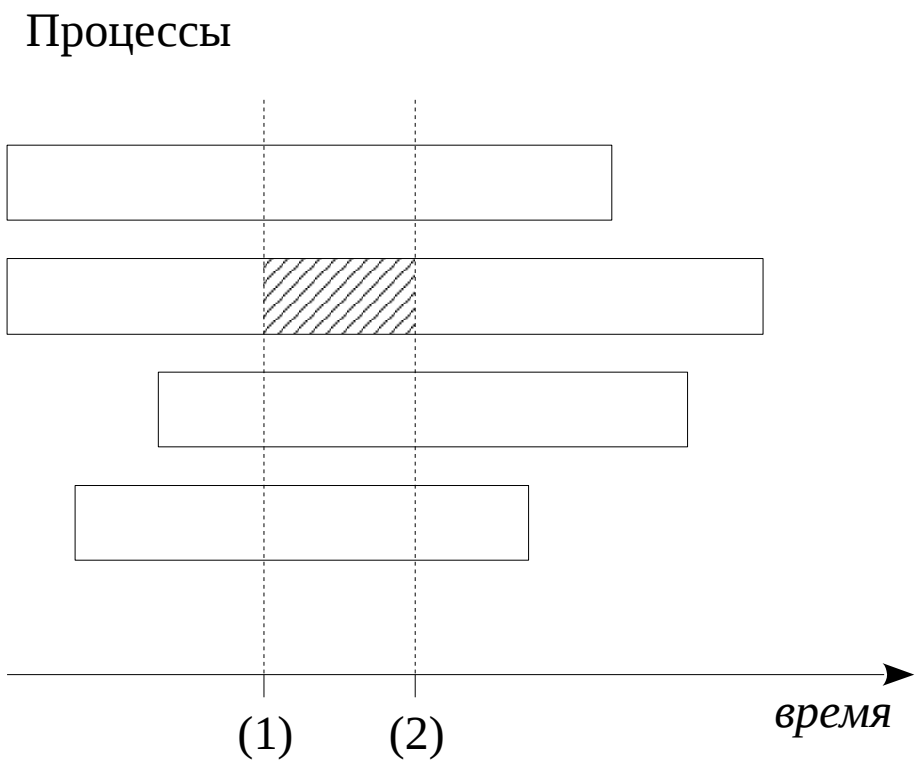
Задача

Трассировка *своего* процесса
за *интервал* времени
по инициативе *пользователя*

Настройка

(1) вкл: SET*

(2) выкл: RESET*



* из функции с правами суперпользователя

AUTO_EXPLAIN

Задачи

- Включение в журнал планов выполнения
- Включение в журнал вложенных запросов

Настройка

- auto_explain.log_min_duration
- auto_explain.log_nested_statements
- ...

PL Profiler

Задачи

- Создание профиля выполнения PL/pgSQL функций

Особенности

- Стороннее расширение (основной автор – Jan Wieck, BigSQL)
- Запись не в журнал сервера
- Гибкие настройки запуска (например, для одного сеанса)
- Отчет о работе в html, включая FlameGraph
- Не работает одновременно с PL Debugger

PG_STAT_STATEMENTS

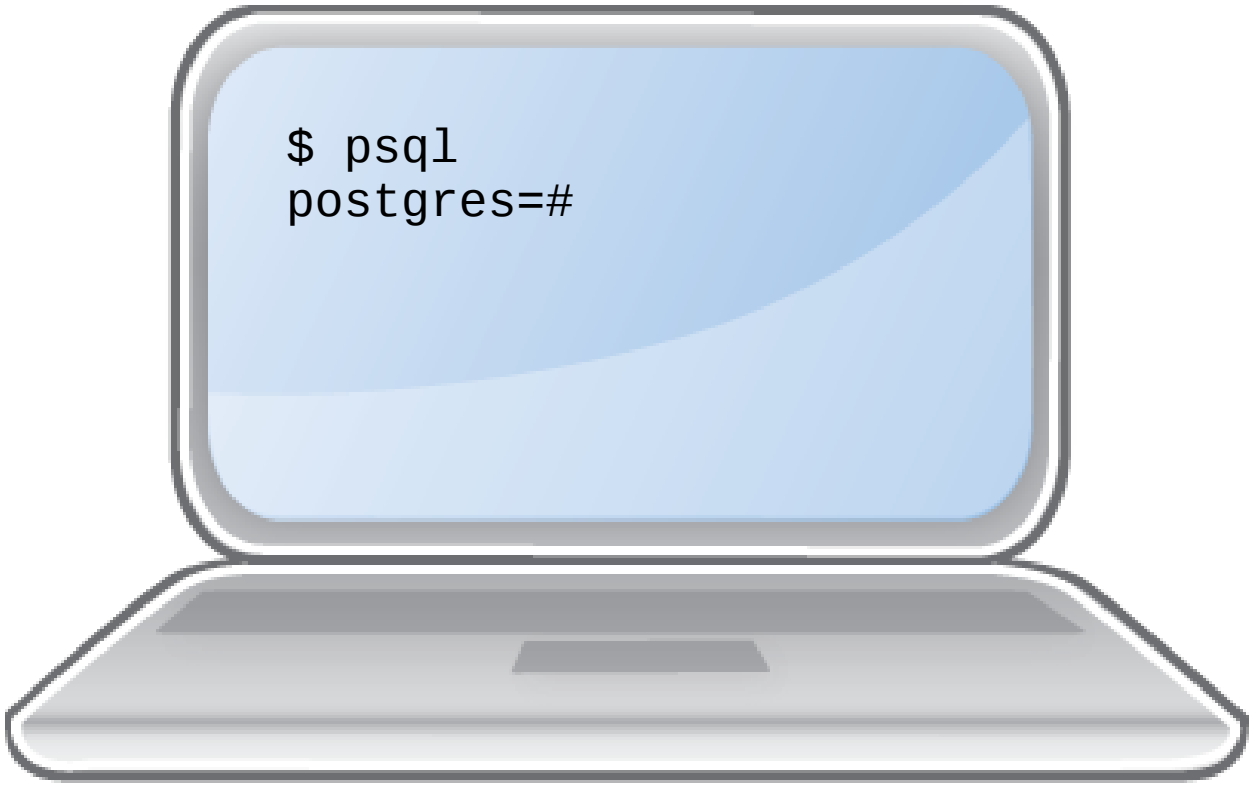
Задачи

- Сбор статистики по исполняемым запросам

Особенности

- Возможность включения вложенных запросов
- Запись не в журнал сервера
- Размер хранилища настраивается
- Статистика в разрезе базы данных, пользователя
- Агрегация похожих запросов (с точностью до констант)

Демонстрация



Итоги

PL Debugger – отладчик PL/pgSQL, API поддерживается сообществом

Служебные сообщения в коде – доступны разные подходы

Гибкие настройки трассировки сеансов

PL Profiler – профилирование PL/pgSQL

pg_stat_statements – статистика по ресурсоемким запросам