



PGBootCamp.ru

Минск, 16 апреля 2024 г.



Пошаговая отладка исходного кода PostgreSQL на примере pg_store_plans

Лев Николаев, «Тантор Лабс»

* Обо мне

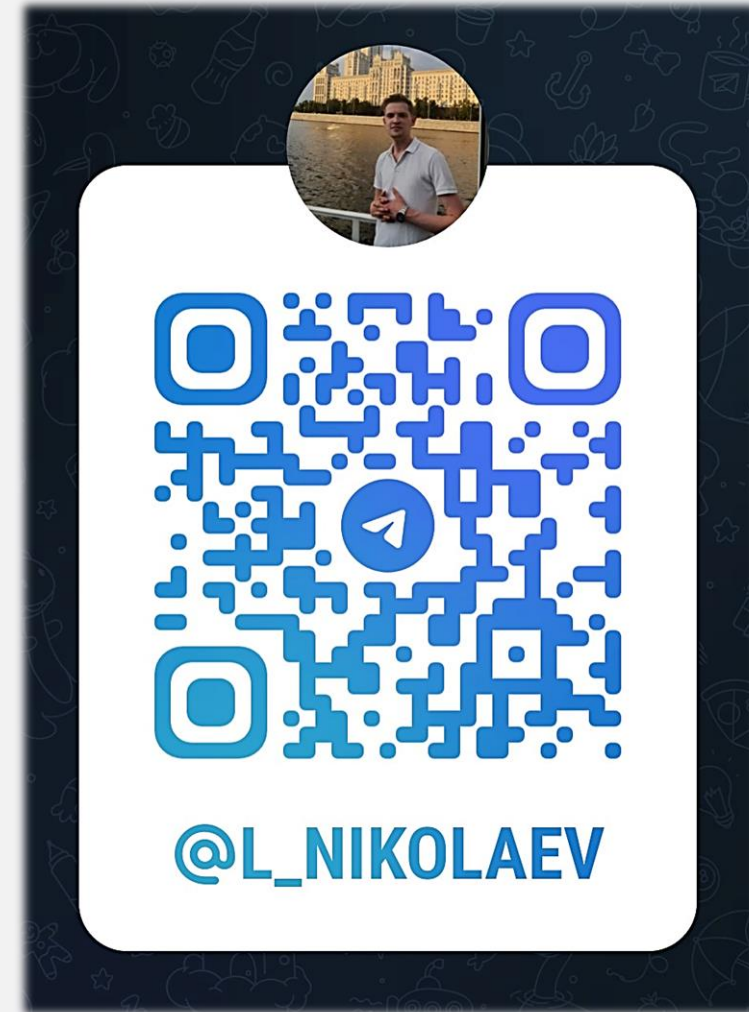


Образование:

- Магистратура «МАДИ»

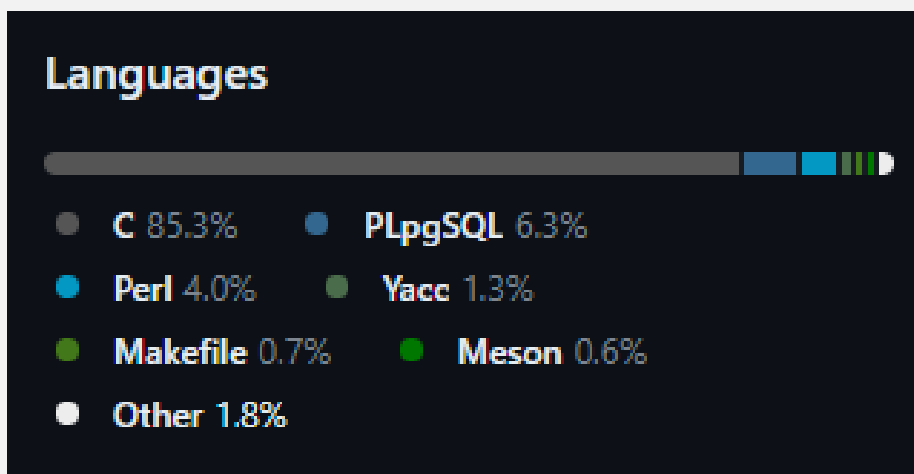
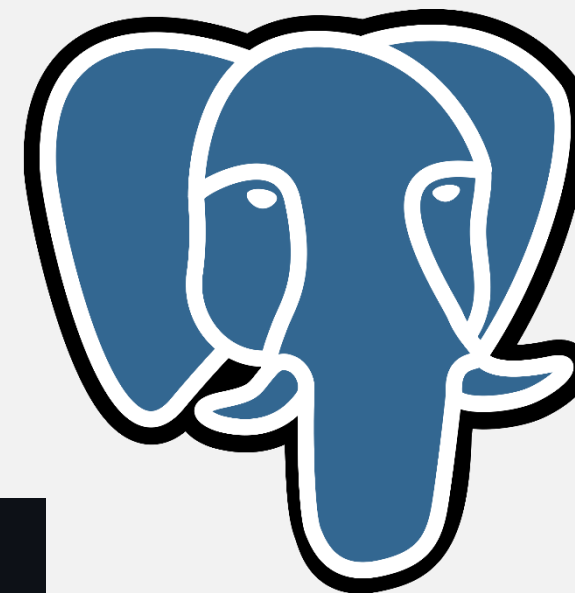
Опыт работы в проектах:

- «МАДИ» TKG, 3D-Modelling
- Tantor(PostgreSQL)



* PostgreSQL

- Открытый исходный код
- Регулярные обновления
- Активное сообщество и CommitFest
- Документация на уровне исходного кода
- Язык C
- 30+ лет ведется разработка



* pg_store_plans: pull request



Added support for PG(v.16.1) #31

Open LevNikolaev wants to merge 2 commits into `ossc-db:master` from `LevNikolaev:pg16_1.7`

Conversation 1

Commits 2

Checks 0

Files changed 4



LevNikolaev commented on Dec 6, 2023

...

Dear pg_store_plans Development Team,

I am pleased to submit this pull request, which brings forward compatibility updates and enhancements for PostgreSQL version 16.1, along with modifications for better support when building with `cassert`. Below, I have detailed the key changes made:

1. Conditional Compilation Adaptation for PostgreSQL 16.1:

- Updated conditional compilation checks in `pg_store_plans.c`. These modifications extend support to PostgreSQL 16.1, ensuring the codebase remains compatible with the latest PostgreSQL version.

2. Header File Path Adjustments for PostgreSQL 16.1:

- Incorporated new include paths (e.g., `#include "nodes/queryjumble.h"`), aligning with the changes in PostgreSQL 16.1's directory structure and APIs.

3. Variable Initializations for `cassert` Compatibility:

- Adjusted the initialization of several static integer variables (`track_level`, `plan_format`, and `plan_storage`). These were previously uninitialized, potentially causing issues when building with `cassert`, as it necessitates all variables to be initialized to ensure robust error checking. I have set their default values to `1`, aligning with the expected default behavior of the system.
- This change enhances the stability of the module, especially in debug builds where `cassert` is commonly enabled, and ensures predictable behavior across different builds and environments.

4. JSON and YAML Processing Function Modifications:

- Altered JSON and YAML processing functions to be compatible with PostgreSQL 16.1's updated behavior and API requirements. This ensures that the module's functionality concerning JSON and YAML processing remains stable and consistent with the latest PostgreSQL standards.

* pg_store_plans: Что это?

- Собирает планы выполнения запросов
- Упрощает и нормализует планы
- Анализ производительности
- Различные форматы хранения:
 - Text
 - Json
 - Yaml

```
### text-chopped ##### Plan 27: Join Filter
Limit (cost=0.00..21.52 rows=100 width=12) (actual time=6.954..7.113 rows=100 loops=1)
  Output: tt2.a, tt2.b, tt2.c
  Buffers: shared hit=40, temp written=12
  -> Nested Loop (cost=0.00..43048282.11 rows=200023334 width=12) (actual time=6.953..7.096 rows=100 loops=1)
    Output: tt2.a, tt2.b, tt2.c
    Join Filter: ((tt2.a < tt3.a) AND ((tt3.a + tt2.a) < 100000))
    Rows Removed by Join Filter: 700
```

```
### normalize ##### Plan 27: Join Filter
{"p":{"t":"5","`":false,"ac":false,"o":["tt2.a","tt2.b","...]}
```

* QR github notes



* Установка PostgreSQL 16



```
git clone --branch REL_16_1 --single-branch
https://github.com/postgres/postgres.git
cd postgres/

./configure --prefix=/usr/local/pgsql \
  --enable-tap-tests \
  --with-python \
  --with-icu \
  --with-lz4 \
  --with-zstd \
  --enable-debug \
  --enable-cassert

make -j4 && sudo make install
```

* Новая группа и пользователь postgres



```
groupadd -r postgres
useradd -r -g postgres -d /var/lib/postgresql \
-s /bin/bash postgres

mkdir -p /var/lib/postgresql
chown postgres:postgres /var/lib/postgresql
chmod 700 /var/lib/postgresql
```


* Установка pg_store_plans 1.8



```
git clone https://github.com/LevNikolaev/bootcamp.git contrib/  
vim contrib/Makefile  
# Добавьте название расширения в "Makefile" для его сборки  
вместе с PostgreSQL.  
>>  
        vacuumlo          \  
        pg_store_plans  
# Соберите и установите расширения postgres ('world-bin').  
make -j4 world-bin && sudo make install-world-bin
```

* Инициализация кластера



```
mkdir /var/log/postgresql
chown postgres:postgres /var/log/postgresql
mkdir /var/lib/postgresql/data
chown postgres:postgres
/var/lib/postgresql/my_data
chown postgres:postgres /usr/local/pgsql
su - postgres
/usr/local/pgsql/bin/initdb \
-D /var/lib/postgresql/my_data
```

* Настройка postgresql.conf



```
cat >> /var/lib/postgresql/my_data/postgresql.conf << EOL
shared_preload_libraries =
'pg_stat_statements,pg_store_plans'
logging_collector = on
log_directory = '/var/log/postgresql/'
log_filename = 'postgresql.log'
log_statement = 'all'
log_min_messages = debug1
log_duration = on
log_destination = 'csvlog'
log_error_verbosity = verbose
log_lock_waits = on
EOL
```

* Запуск БД и проверка лога



```
/usr/local/pgsql/bin/pg_ctl \  
-D /var/lib/postgresql/my_data \  
start
```

```
ps f --forest -u postgres | grep -v "f --forest -u postgres"
```

```
>>
```

PID	TTY	STAT	TIME	COMMAND
10015	?	Ss	0:00	/usr/local/pgsql/bin/postgres -D /var/lib/postgresql/my_data
10016	?	Ss	0:00	_ postgres: logger
10017	?	Ss	0:00	_ postgres: checkpointer
10018	?	Ss	0:00	_ postgres: background writer
10020	?	Ss	0:00	_ postgres: walwriter
10021	?	Ss	0:00	_ postgres: autovacuum launcher
10022	?	Ss	0:00	_ postgres: logical replication launcher

```
tail -f /var/log/postgresql/postgresql.csv
```

```
>>
```

```
... "starting background worker process " "logical replication  
launcher" """,,,,,,"do_start_bgworker, postmaster.c:5723", "", "postmaster", ,0
```

* Создание расширений в кластере



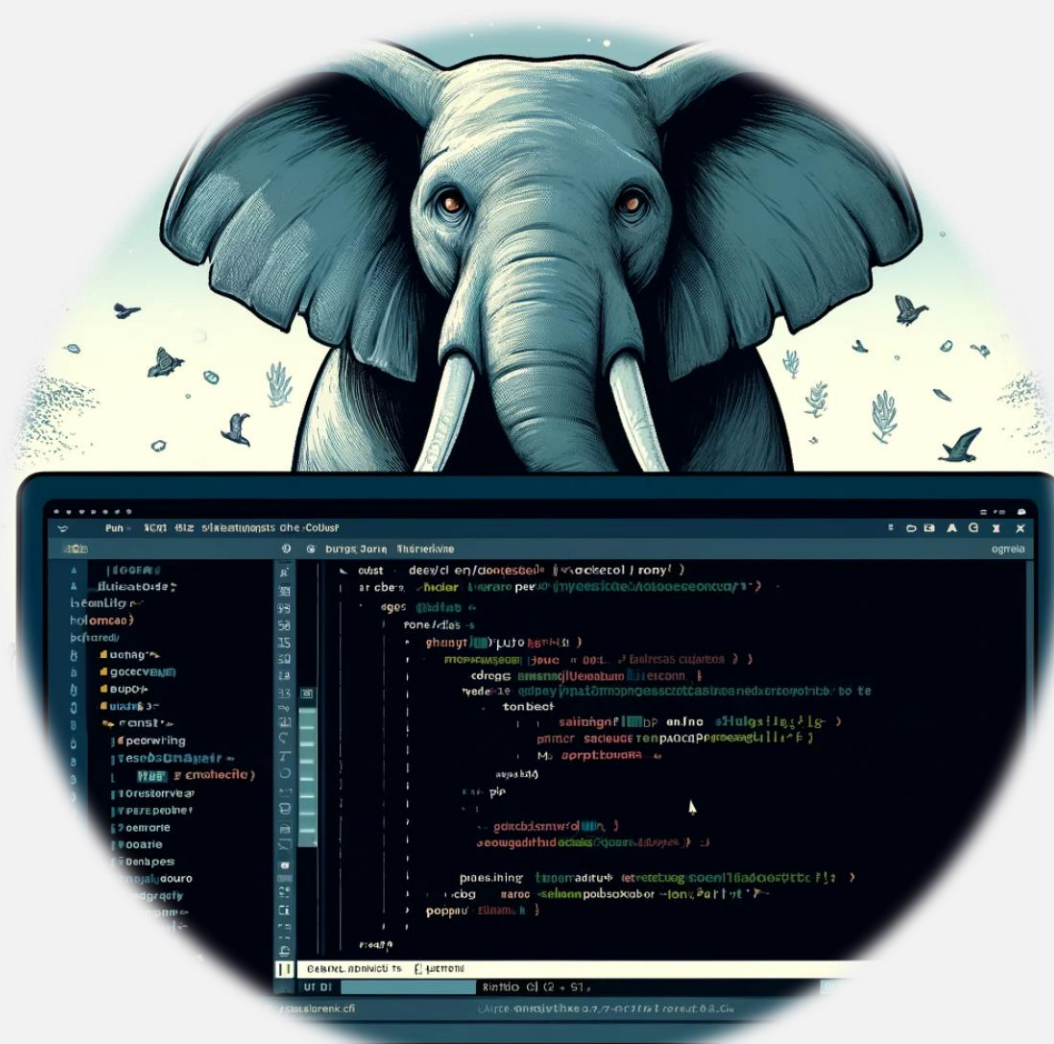
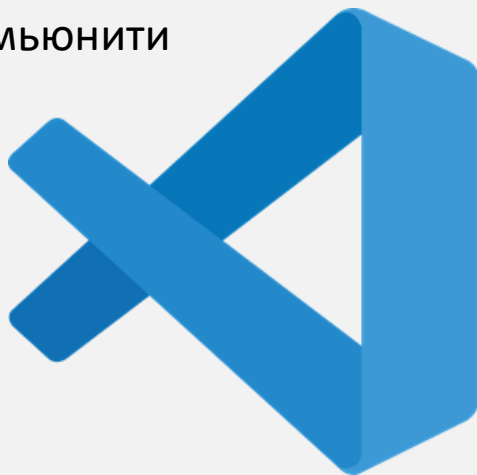
```
sudo -u postgres /usr/local/pgsql/bin/psql -p  
5432 -c "CREATE EXTENSION pg_stat_statements;"
```

```
sudo -u postgres /usr/local/pgsql/bin/psql -p  
5432 -c "CREATE EXTENSION pg_store_plans;"
```

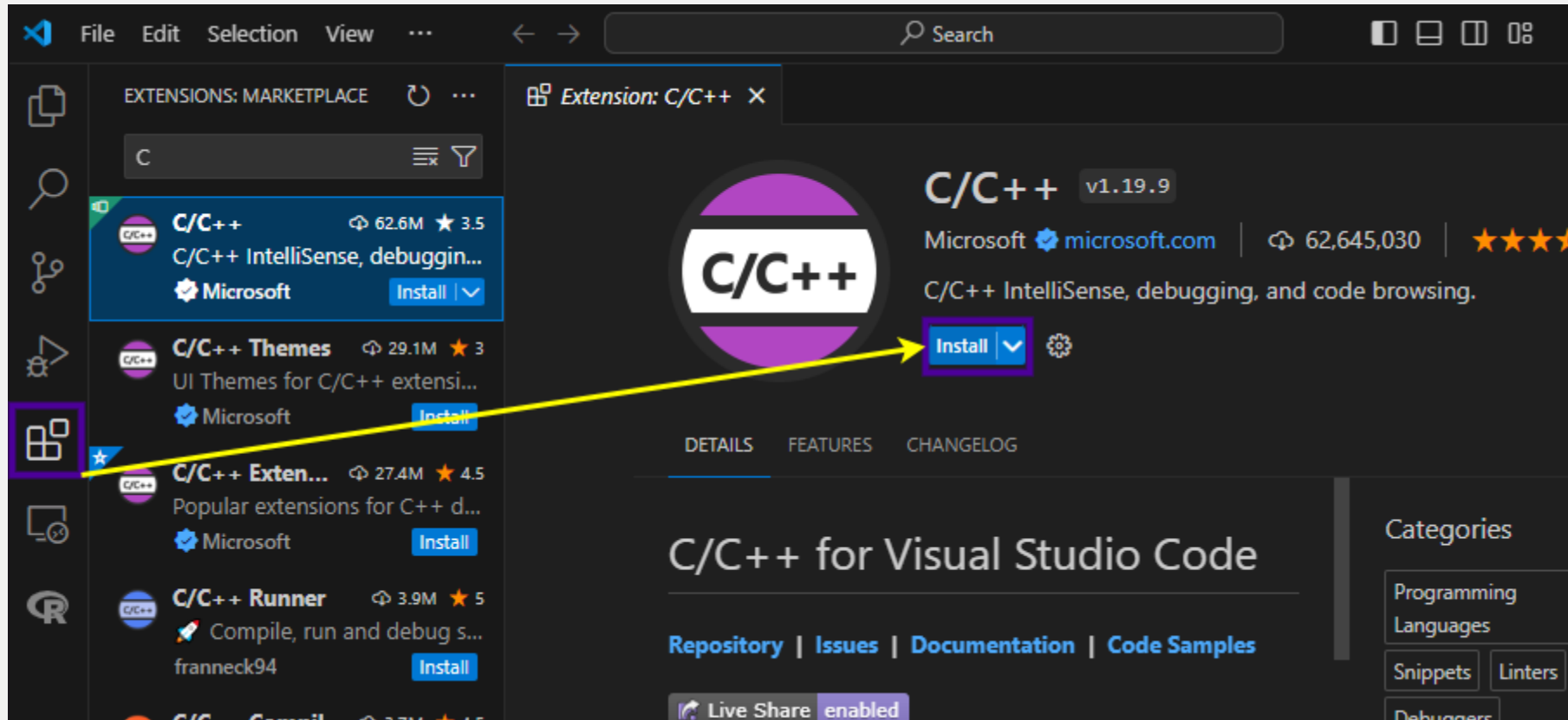
✳ Пошаговая отладка

Преимущества отладки в VS Code

- Интеграция с Git
- Поддержка множества языков
- Настраиваемые расширения для улучшения отладки
- Удобство перехода по исходному коду
- Удобное управление процессом отладки
- Большое комьюнити



* Установка расширения в VS Code

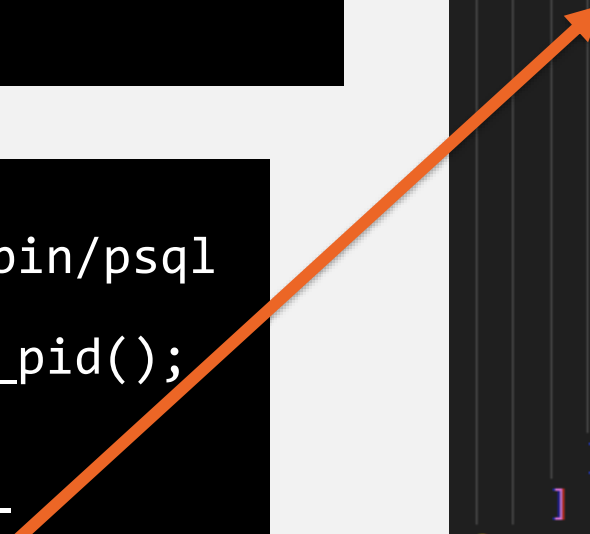


* pg_backend_pid и launch.json

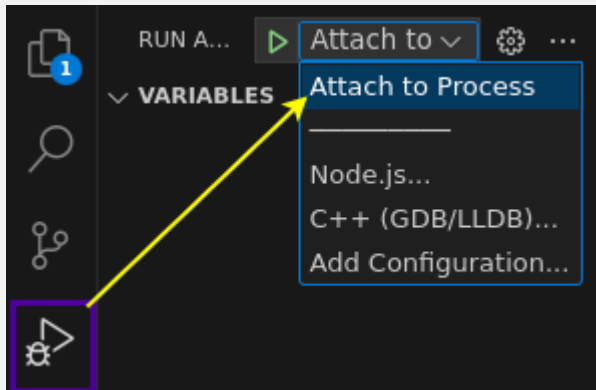
```
mkdir -p postgres/.vscode  
cd postgres/.vscode/  
vim launch.json
```

```
su - postgres  
/usr/local/pgsql/bin/psql  
  
SELECT pg_backend_pid();  
>>  
   pg_backend_pid  
-----  
             655  
  
(1 row)
```

```
{  
  "version": "0.2.0",  
  "configurations": [  
    {  
      "name": "Attach to Process",  
      "type": "cppdbg",  
      "request": "attach",  
      "program": "/usr/local/pgsql/bin/postgres",  
      "processId": "<ID процесса>",  
      "MIMode": "gdb",  
      "miDebuggerPath": "/usr/bin/gdb",  
      "setupCommands": [  
        {  
          "description": "Enable pretty-printing for gdb",  
          "text": "-enable-pretty-printing",  
          "ignoreFailures": true  
        }  
      ]  
    }  
  ]  
}
```



* Интерфейс отладки VS Code





ОБНАРУЖЕНИЕ ПРОБЛЕМЫ РАБОТЫ РАСШИРЕНИЯ



```
Bitmapset *  
bms_add_member(Bitmapset *a, int x)  
{  
    int      wordnum,  
            bitnum;  
  
    if (x < 0)  
        elog(ERROR, "negative bitmapset member not allowed");  
    if (a == NULL)  
        return bms_make_singleton(x);  
    wordnum = WORDNUM(x);  
    bitnum = BITNUM(x);  
  
    /* enlarge the set if necessary */
```

* Демонстрация отладки



Спасибо!



Минск, 16 апреля 2024 г.

