

Depurando PL/pgSQL com OmniDB

Sumário

- ✓ Conceitos
- ✓ Requerimentos
- ✓ Alternativas
- ✓ Depurador OmniDB
- ✓ Demonstração
- ✓ Próximos Passos

OmniDB

The screenshot displays the OmniDB web interface. At the top, there's a navigation bar with the OmniDB logo, 'Connections', and user information (admin). Below this, a toolbar shows various connection tabs like '2.7.0', 'pg10 local', and 'mysql local'. The main interface is divided into several sections:

- Left Panel:** A tree view showing the database structure for PostgreSQL 10.1. It includes 'Databases (7)', 'postgres' (with 'Schemas (4)' like public, pg_catalog, information_schema, test), 'Extensions', and 'Logical Replication'. Below this is a 'Properties' tab for the selected table 'sql_features', showing details like Database (postgres), Schema (information_schema), Table (sql_features), OID (12773), Owner (postgres), and Size (56 kB).
- Center Panel:** A query editor with a console window. The query is:

```
1 -- Querying Data
2 select t.*
3 from information_schema.sql_features t
```
- Right Panel:** A results table showing 50 records. The table has columns: feature_id, feature_name, sub_feature_id, sub_feature_name, is_supported, is_verified_by, and comments. The first 13 rows are visible.

	feature_id	feature_name	sub_feature_id	sub_feature_name	is_supported	is_verified_by	comments
1	B011	Embedded Ada			NO		
2	B012	Embedded C			YES		
3	B013	Embedded COBOL			NO		
4	B014	Embedded Fortran			NO		
5	B015	Embedded MUMPS			NO		
6	B016	Embedded Pascal			NO		
7	B017	Embedded PL/I			NO		
8	B021	Direct SQL			YES		
9	B031	Basic dynamic SQL			NO		
10	B032	Extended dynamic SQL			NO		
11	B032	Extended dynamic SQL	01	<describe input statement>	NO		
12	B033	Untyped SQL-invoked functi...			NO		
13	B034	Dynamic specification of cur...			NO		

Evolução

- ✓ 2.3 - 02/11/17
 - ✓ Depurador PL/pgSQL
- ✓ 2.4 - 07/12/17
 - ✓ Monitoramento
- ✓ 2.5 - 12/02/18
 - ✓ Oracle
 - ✓ Painel DDL/Propriedades
- ✓ 2.6 - 15/03/18
 - ✓ Aba console
 - ✓ Exportar queries
- ✓ 2.7 - 12/04/18
 - ✓ MySQL
- ✓ 2.8 - 17/05/18
 - ✓ Túnel SSH
- ✓ 2.9 - 19/06/18
 - ✓ Busca avançada de objetos
 - ✓ Sistema de Plugins
- ✓ 2.10 - 26/07/18
 - ✓ Foreign Data Wrappers

PL/pgSQL

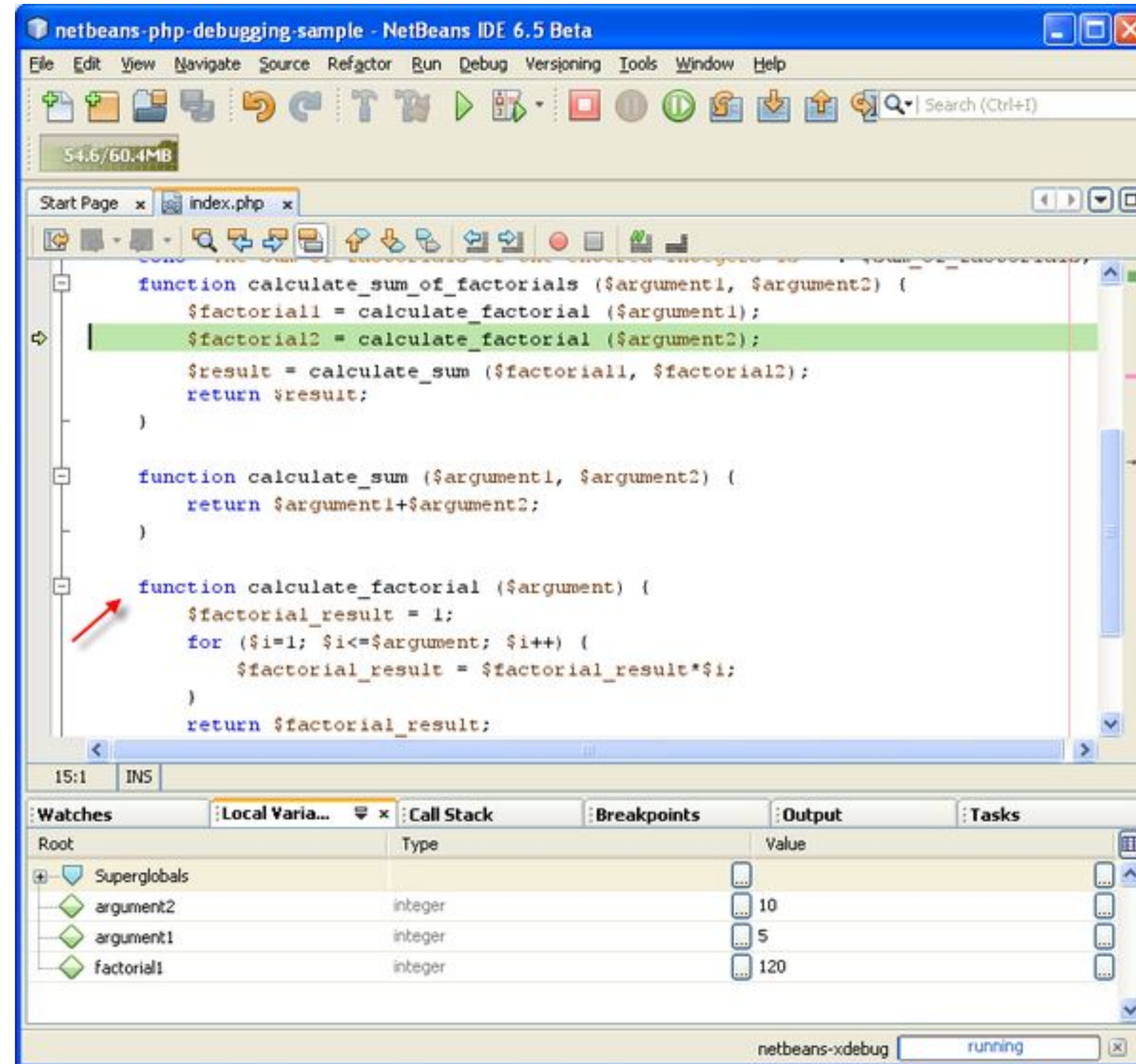
- ✓ Linguagem procedural para PostgreSQL
- ✓ Variáveis
- ✓ Estruturas de controle para criar lógicas complexas
- ✓ Criação de funções e procedimentos (PostgreSQL 11)

```
1 CREATE OR REPLACE FUNCTION omnldb.insert_sort(input_vector float[])
2 RETURNS float[] AS $$
3 DECLARE
4     temp float;
5     output_vector float[];
6     i integer;
7     j integer;
8 BEGIN
9     output_vector := input_vector;
10    i := 2;
11    WHILE i < array_length(output_vector::float[],1) + 1 LOOP
12        temp := output_vector[i];
13        j := i - 1;
14        WHILE j >= 0 AND output_vector[j] > temp LOOP
15            output_vector[j+1] := output_vector[j];
16            j := j - 1;
17        END LOOP;
18        output_vector[j+1] := temp;
19        i := i + 1;
20    END LOOP;
21
22    RETURN output_vector;
23 END;
24 $$ LANGUAGE plpgsql;
```

Depurador

- ✓ O que faz?
 - ✓ Executar trechos de código pausadamente
 - ✓ Monitorar valores de variáveis durante o processo
- ✓ Objetivos
 - ✓ Analisar fluxo de andamento do código
 - ✓ Achar bugs
 - ✓ Encontrar gargalos no corpo da função

Depurador



Depurador PL/pgSQL

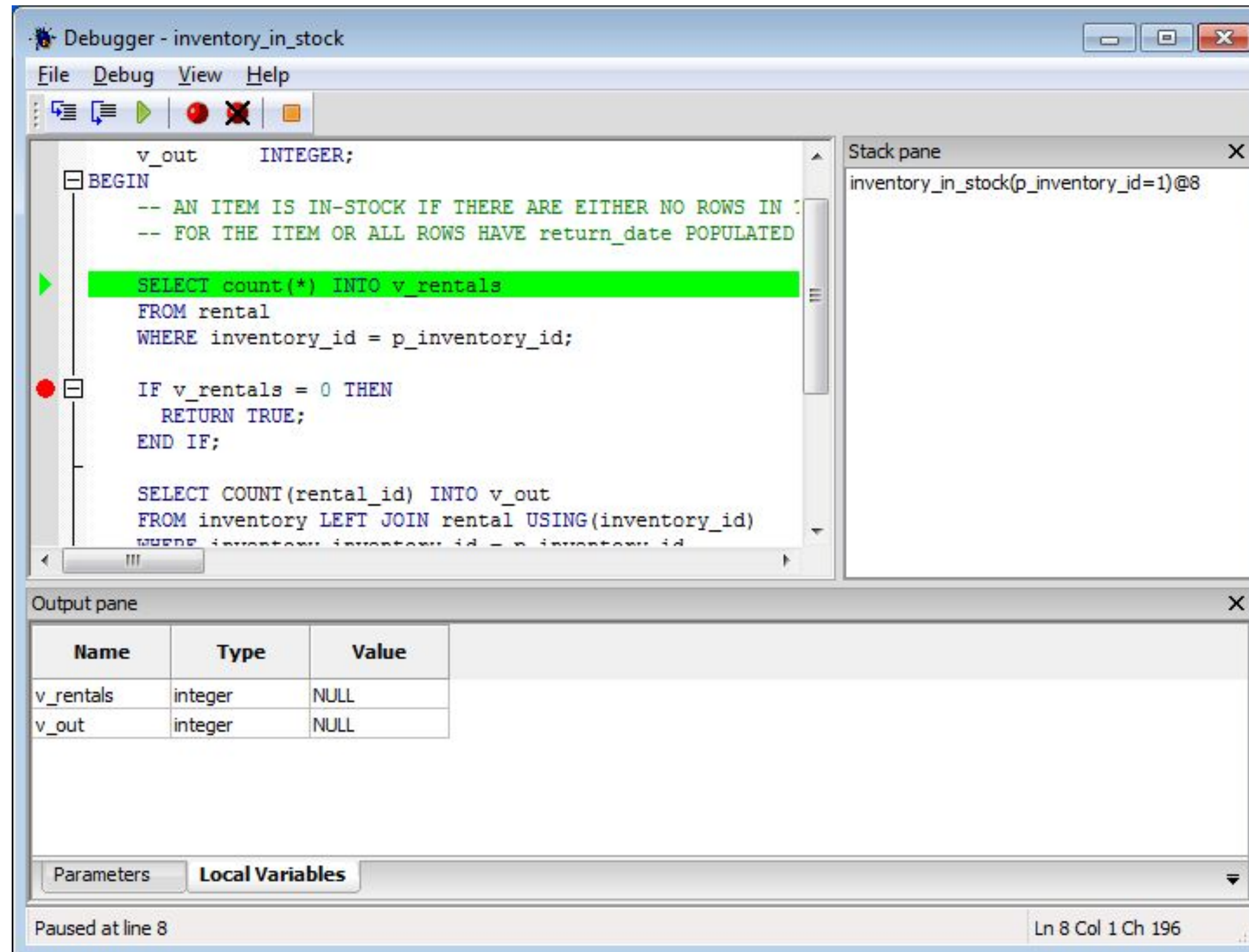
- ✓ Execução de funções PL/pgSQL ocorre dentro do PostgreSQL
- ✓ Perguntas:
 - ✓ Como pausar a execução de uma função PL/pgSQL?
 - ✓ Como enviar breakpoints para uma função PL/pgSQL?
 - ✓ Como recuperar valores de variáveis?

PostgreSQL Hooks

- ✓ Bibliotecas externas carregadas pelo PostgreSQL
- ✓ Interrompem ou modificam rotinas padrões
- ✓ Hooks para diversos fins:
 - ✓ Autenticação
 - ✓ Logs
 - ✓ Início e fim de queries
 - ✓ PL/pgSQL - início/fim de funções e de cada linha presente nelas

Alternativas

Pgadmin



```
Debugger - inventory_in_stock
File Debug View Help
v_out    INTEGER;
BEGIN
  -- AN ITEM IS IN-STOCK IF THERE ARE EITHER NO ROWS IN :
  -- FOR THE ITEM OR ALL ROWS HAVE return_date POPULATED
  SELECT count(*) INTO v_rentals
  FROM rental
  WHERE inventory_id = p_inventory_id;
  IF v_rentals = 0 THEN
    RETURN TRUE;
  END IF;
  SELECT COUNT(rental_id) INTO v_out
  FROM inventory LEFT JOIN rental USING(inventory_id)
  WHERE inventory_id = p_inventory_id;

```

Stack pane
inventory_in_stock(p_inventory_id=1)@8

Name	Type	Value
v_rentals	integer	NULL
v_out	integer	NULL

Parameters Local Variables

Paused at line 8 Ln 8 Col 1 Ch 196

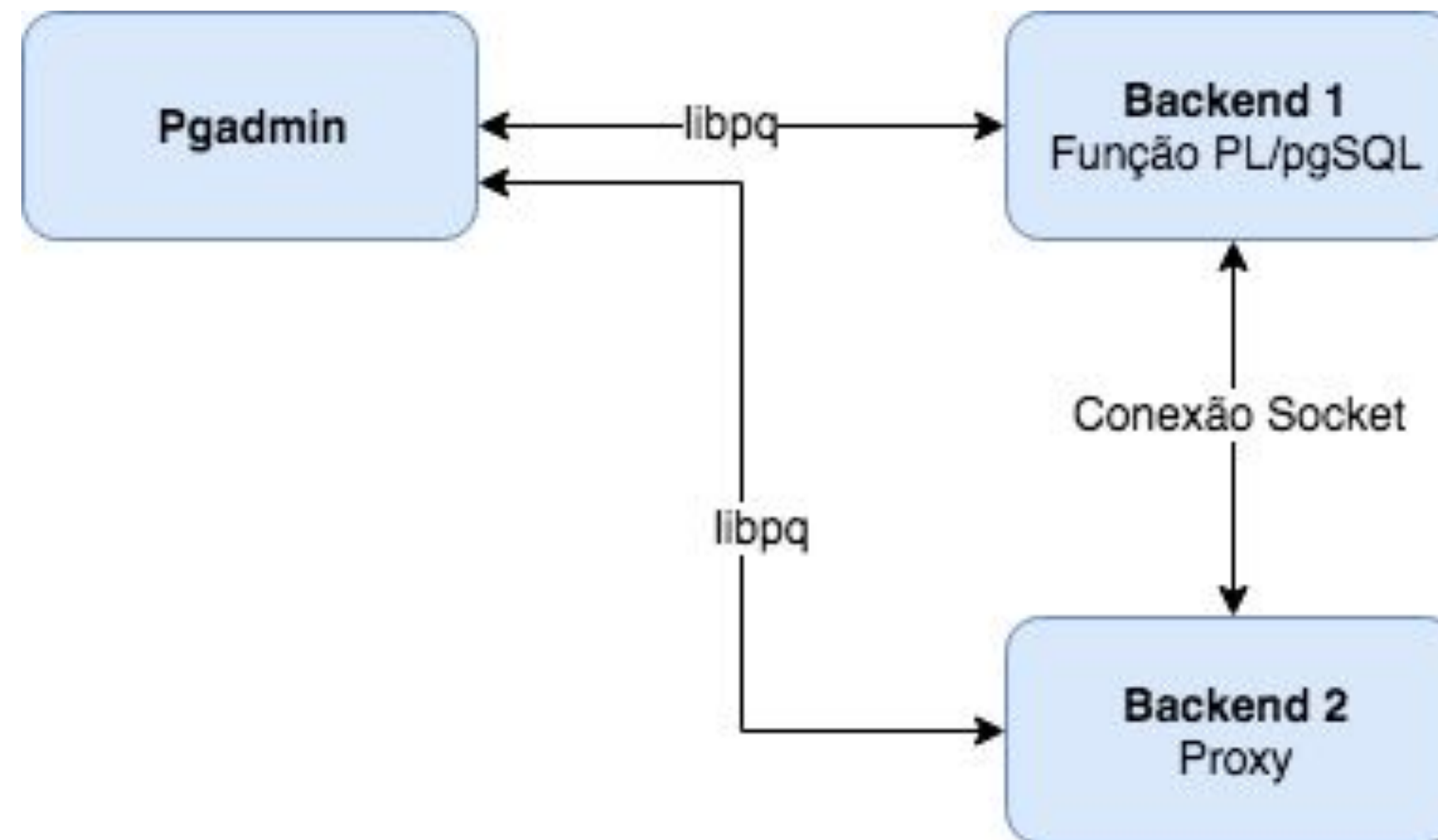
Alternativas

Pgadmin

- ✓ Requerimentos
 - ✓ Necessário ser superuser para usar o depurador
 - ✓ Plugin: plugin_debugger.so
 - ✓ Extensão: pldbgapi
- ✓ Funcionalidades
 - ✓ Execução passo a passo ou com breakpoints
 - ✓ Tabela de variáveis
 - ✓ Pilha de execução

Alternativas

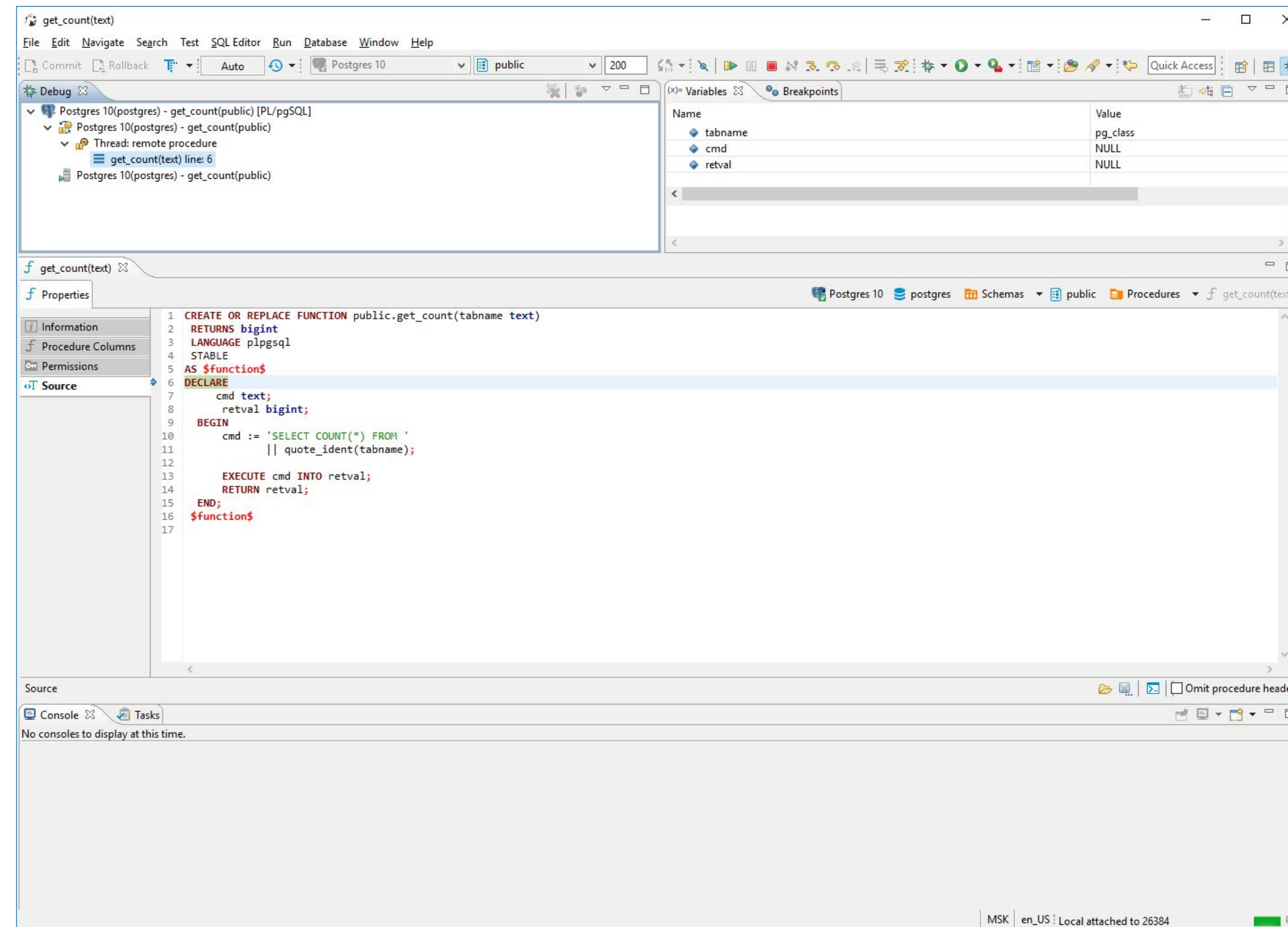
Pgadmin



Alternativas

DBeaver

- ✓ Utiliza mesmos plugin e extension do Pgadmin



Depurador OmniDB

Motivações

- ✓ Controle sobre funcionalidades e desenvolvimento
- ✓ Garantia de continuidade

Depurador OmniDB

Objetivos

- ✓ Implementar um depurador simples mas extensível
- ✓ Utilizar apenas recursos do PostgreSQL (sem bibliotecas externas)
- ✓ Funcionalidades do depurador:
 - ✓ Execução passo a passo ou com breakpoints
 - ✓ Tabela de variáveis
 - ✓ Estatísticas de execução de cada passo



Depurador OmniDB

Detalhes de Implementação

- ✓ Utilizar Hooks de PL/pgSQL para controlar a depuração
- ✓ Hooks devem pausar a cada linha da função e esperar comando do OmniDB
- ✓ OmniDB deve dizer quando Hooks podem continuar execução
- ✓ OmniDB e Hooks devem estar de alguma forma sincronizados. **Como sincronizar?**

Depurador OmniDB

Advisory Locks

- ✓ Locks adquiridos e gerenciados pelo usuário
- ✓ Utilizados para emular estratégias de Lock na própria aplicação
- ✓ Permite bloquear operações banco através de locks em tabelas e seus registros
- ✓ Utilizado pelo depurador do OmniDB para sincronizar interface com hooks PL/pgSQL

Depurador OmniDB

Advisory Locks

	id	nome
1	1	nome 1
2	2	nome 2
3	3	nome 3

Conexão 1

Conexão 2

1 `SELECT pg_advisory_lock(1) FROM tabela WHERE id = 1`

.

2 .

`SELECT pg_advisory_lock(1) FROM tabela WHERE id = 1`

3 .

x

4 `SELECT pg_advisory_unlock(1) FROM tabela WHERE id = 1`

x

5 `SELECT pg_advisory_lock(1) FROM tabela WHERE id = 1`

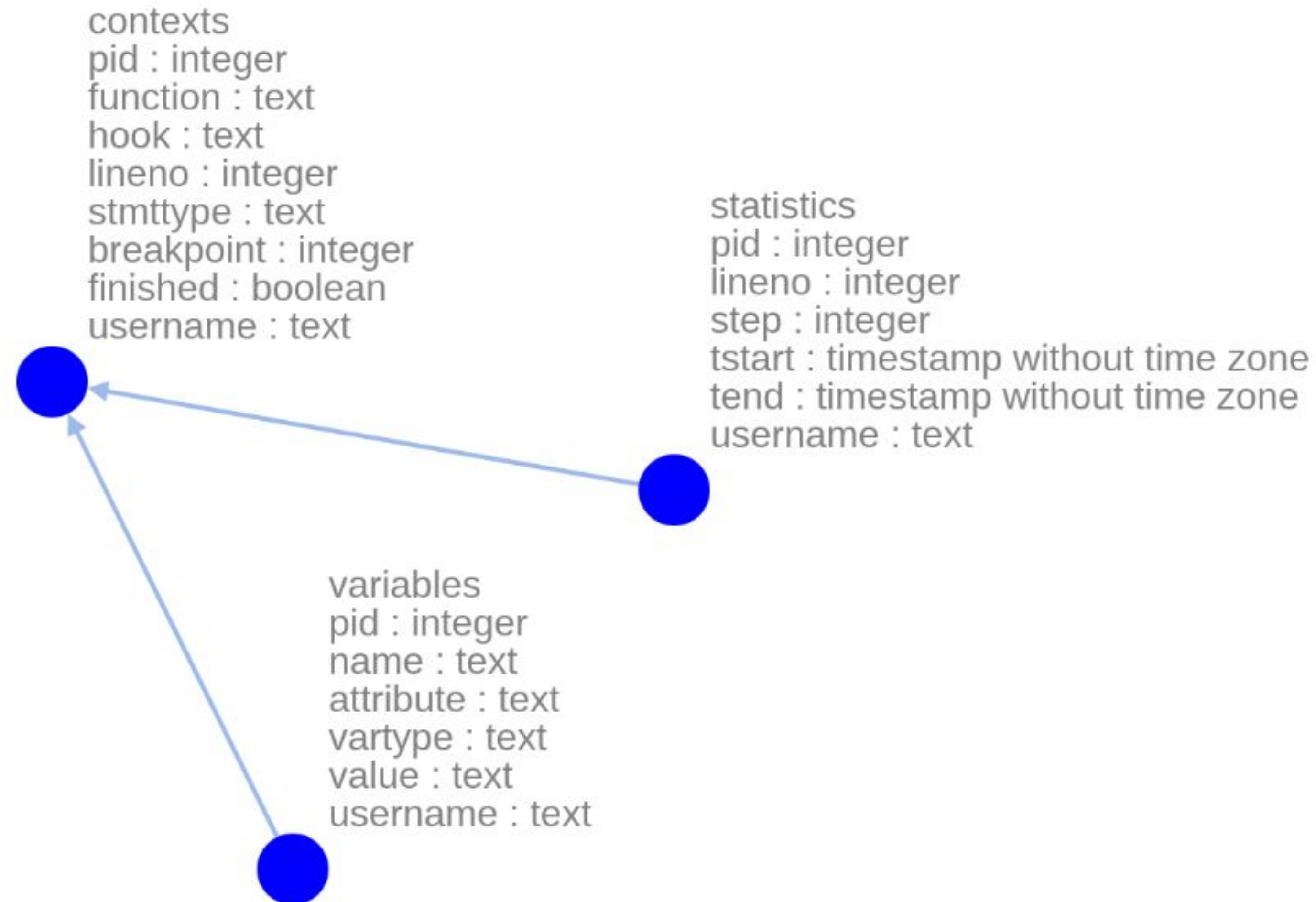
.

6 x

.

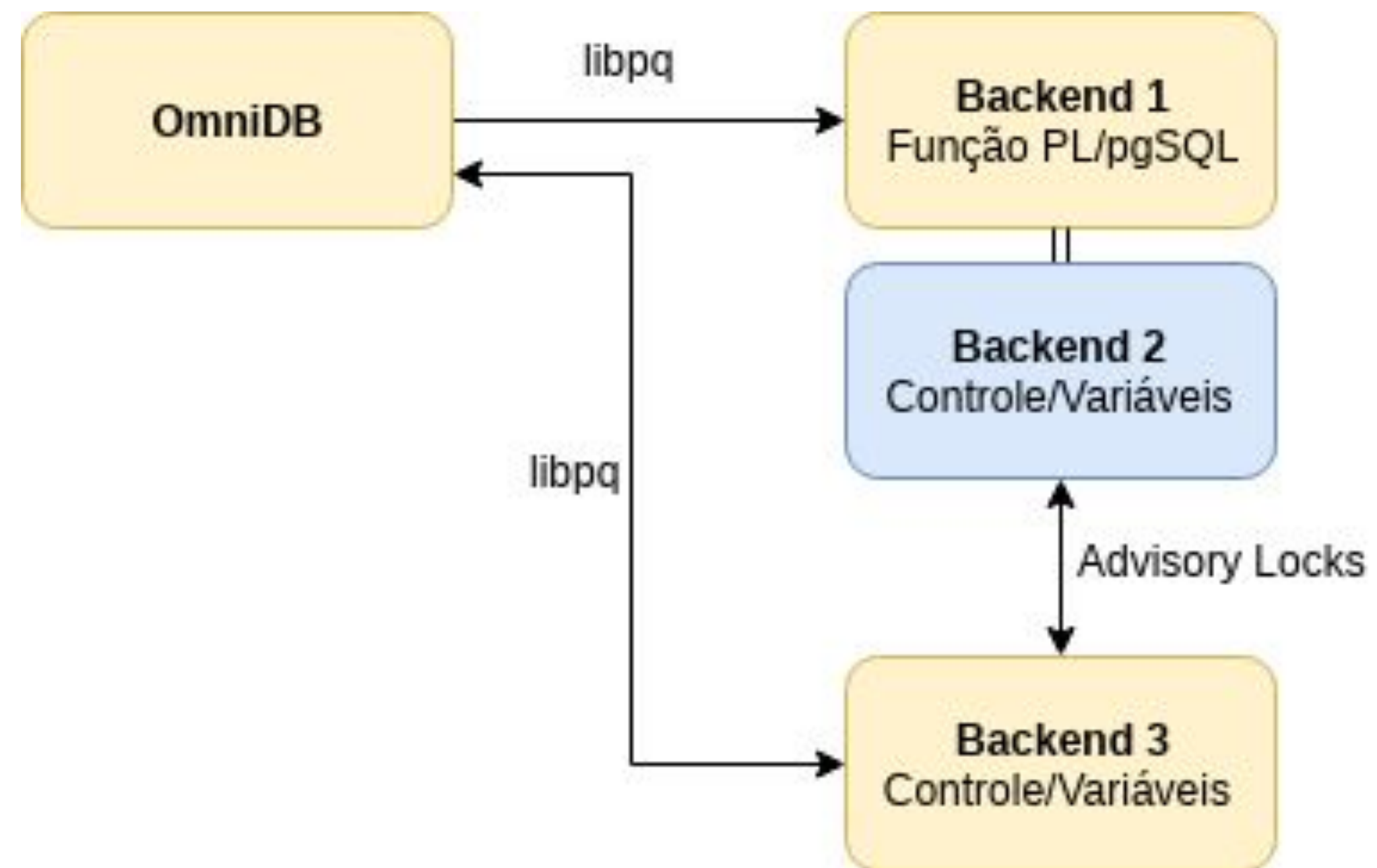
Depurador OmniDB

Tabelas de Controle



Depurador OmniDB

Arquitetura



Depurador OmniDB

Requerimentos

- ✓ Plugin: omnidb_plugin.so / omnidb_plugin.dll
- ✓ Criação do schema com tabelas de controle



Depurador OmniDB

Demonstração...

Depurador OmniDB

Próximos Passos...

- ✓ Step Into (Depurar chamadas de funções dentro de funções)
- ✓ Múltiplos Breakpoints
- ✓ Call Stack (pilha de chamadas)
- ✓ Call Graph (grafo de chamadas)
- ✓ Alterar valores de variáveis
- ✓ Depuração de procedures (PostgreSQL 11)



Obrigado!

rafael.castro@2ndquadrant.com