

??? ?????????? ? ?????????? JSON/JSONB ?? C# ? PL/Python ?????????? ?????? Npgsql?

Передача данных в формате **JSON / JSONB** из C# в PL/Python — один из наиболее эффективных подходов: C# сериализует объект в JSON, Npgsql передает его в PostgreSQL как native-тип `jsonb`, а PL/Python автоматически или с помощью стандартного модуля `json` превращает его в родной словарь (`dict`) или список (`list`).

1. ?????????? ?????????? ? PostgreSQL (PL/Python)

Функция принимает тип `jsonb` (или `json`). В PL/Python аргументы типа `json/jsonb` передаются в виде обычных **строк**, поэтому внутри их нужно распарсить с помощью стандартного модуля `json`.

SQL

```
CREATE OR REPLACE FUNCTION process_user_data(payload jsonb)
RETURNS TABLE(user_id int, status text, is_valid boolean) AS $$
    import json

    # PL/Python получает json/jsonb как строку
    data = json.loads(payload)

    users = data.get("users", [])
    result = []

    for user in users:
        u_id = user.get("id")
        email = user.get("email", "")

        # Простая проверка/обработка
        is_valid = "@" in email
        status = "Active" if user.get("age", 0) >= 18 else "Pending"
```

```

        result.append({
            "user_id": u_id,
            "status": status,
            "is_valid": is_valid
        })

    return result

$$ LANGUAGE plpython3u;

```

2. ?????? ?? C# ? ????????? Npgsql ? System.Text.Json

В современных версиях Npgsql передача JSON выполняется через встроенную поддержку `System.Text.Json` или явное указание типа `NpgsqlDbType.Jsonb`.

???????? 1: ???????????????? `System.Text.Json` (?????????????????)

C#
■ ■

```

using System;
using System.Collections.Generic;
using System.Text.Json;
using System.Threading.Tasks;
using Npgsql;
using NpgsqlTypes;

public class JsonPostgresExample
{
    private const string ConnectionString =
        "Host=localhost;Database=mydb;Username=postgres;Password=secret";

    public async Task ProcessDataAsync()
    {
        // 1. Формируем C# объект для передачи
        var requestData = new
        {
            users = new[]
            {
                new { id = 101, email = "alex@example.com", age = 25 },
                new { id = 102, email = "invalid_email", age = 16 }
            }
        };
    }
}

```

```

    }
};

// 2. Сериализуем объект в строку JSON
string jsonPayload = JsonSerializer.Serialize(requestData);

await using var conn = new NpgsqlConnection(ConnectionString);
await conn.OpenAsync();

const string sql = "SELECT user_id, status, is_valid FROM
process_user_data(@payload)";

await using var cmd = new NpgsqlCommand(sql, conn);

// 3. Добавляем параметр с явным указанием NpgsqlDbType.Jsonb
cmd.Parameters.Add(new NpgsqlParameter("payload", NpgsqlDbType.Jsonb)
{
    Value = jsonPayload
});

// 4. Читаем результат
await using var reader = await cmd.ExecuteReaderAsync();
while (await reader.ReadAsync())
{
    var userId = reader.GetInt32(0);
    var status = reader.GetString(1);
    var isValid = reader.GetBoolean(2);

    Console.WriteLine($"ID: {userId} | Статус: {status} | Валиден: {isValid}");
}
}
}

```

3. ?????????? ??????????: ????????? JSONB ?? PL/Python ? C#

Если функция PL/Python должна вернуть сложный JSON-объект обратно в C#:

????????? ? PostgreSQL:

SQL
■ ■

```

CREATE OR REPLACE FUNCTION get_system_stats()
RETURNS jsonb AS $$
    import json

    stats = {
        "server_time": "2026-09-12",
        "metrics": {
            "cpu_usage": 42.5,
            "active_connections": 12
        },
        "tags": ["prod", "primary"]
    }

    # Возвращаем JSON-строку
    return json.dumps(stats)
$$ LANGUAGE plpython3u;

```

??? ?????? ? C#:

C#


```

public async Task GetStatsAsync()
{
    await using var conn = new NpgsqlConnection(ConnectionString);
    await conn.OpenAsync();

    const string sql = "SELECT get_system_stats()";
    await using var cmd = new NpgsqlCommand(sql, conn);

    var rawJson = (string)await cmd.ExecuteScalarAsync();

    // Десериализуем JSON в C# объект
    using var doc = JsonDocument.Parse(rawJson);
    var cpu = doc.RootElement
        .GetProperty("metrics")
        .GetProperty("cpu_usage")
        .GetDouble();

    Console.WriteLine($"CPU Usage: {cpu}%");
}

```

????????? ???????? ??? ?????? ? JSON/JSONB:

1. **Тип параметра в С#:** При передаче строкового JSON всегда указывайте `NpgsqlDbType.Jsonb` (или `NpgsqlDbType.Json`), иначе Npgsql передаст строку как обычный `text`, и Postgres выдаст ошибку несоответствия типов.

2. Сериализация внутри PL/Python:

- При **приеме** `jsonb` в Python аргумент является строкой `str` — применяйте `json.loads(arg)`.
- При **возврате** `jsonb` отправляйте строку — применяйте `json.dumps(obj)`.

Revision #1

Created 2026-09-28 06:20:20 UTC by Nikolay

Updated 2026-09-28 06:22:21 UTC by Nikolay