

JSONB В POSTGRESQL И NOSQL ТРЕНД

сравнение функциональности и производительности

Дмитрий Долгов

February 3, 2016

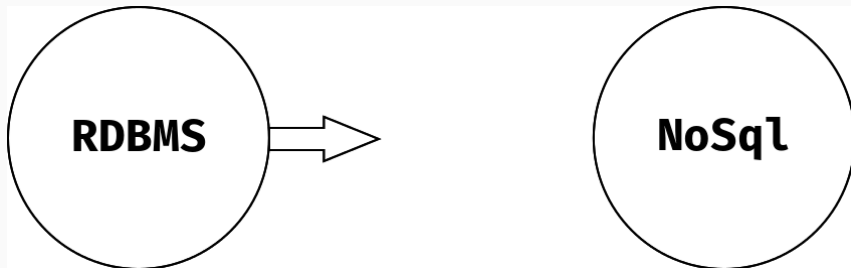
NoSql популярен и это многим не дает покоя.
Это приводит к тому, что многие реляционные базы данных предлагают поддержку тех или иных возможностей, изначально ассоциирующихся с NoSql.



RDBMS



NoSql



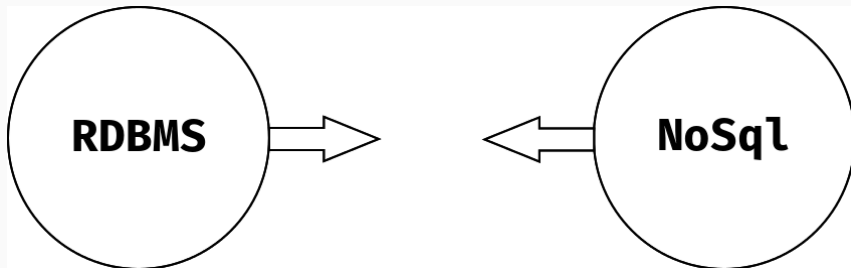
Почему это важно? Каков уровень поддержки хранения слабо-структурированных данных в PostgreSQL?

ФИЛОСОФСКОЕ ВВЕДЕНИЕ

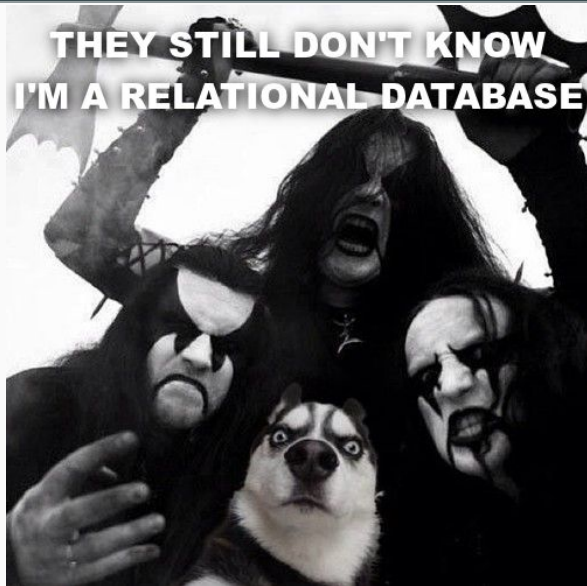


ФИЛОСОФСКОЕ ВВЕДЕНИЕ





ФИЛОСОФСКОЕ ВВЕДЕНИЕ



СРАВНЕНИЕ ФУНКЦИОНАЛА

ОБЗОР ПО КАТЕГОРИЯМ

DB	Native	Select	Modify	Delete	Attributes	Indexing	Search	Conversion	Syntactic
PG	✓	✓	✓	✓	✓	✓	Q	✓	Q
Mysql	✓	✓	✓	✓	✓	Q	Q	✗	Q
Oracle	✗	✓	✗	✗	✗	✗	Q	Q	Q
DB2	✓	✓	✗	✗	✗	✗	✗	Q	✗
MSSql	✗	✓	✗	✗	✗	✗	Q	Q	✗

POSTGRESQL

- Hstore
- Json
- Jsonb + (jsonbx)

PostgreSQL 9.5

МНОГОЛИКИЙ JSONB

```
select '"string"'::jsonb;
```

МНОГОЛИКИЙ JSONB

```
select '"string"'::jsonb;
```

```
select '1'::jsonb;
```

МНОГОЛИКИЙ JSONB

```
select '"string"'::jsonb;
```

```
select '1'::jsonb;
```

```
select 'true'::jsonb;
```


МНОГОЛИКИЙ JSONB

```
select '"string"'::jsonb;
```

```
select '1'::jsonb;
```

```
select 'true'::jsonb;
```

```
select '[1, 2, 3]'::jsonb;
```

МНОГОЛИКИЙ JSONB

```
select '"string"'::jsonb;
```

```
select '1'::jsonb;
```

```
select 'true'::jsonb;
```

```
select '[1, 2, 3]'::jsonb;
```

```
select '["string", 1]'::jsonb;
```

МНОГОЛИКИЙ JSONB

```
select '''string''::jsonb;
```

```
select '1'::jsonb;
```

```
select 'true'::jsonb;
```

```
select '[1, 2, 3]'::jsonb;
```

```
select '["string", 1]'::jsonb;
```

```
select '{"key": {"nested": "value"}}'::jsonb;
```

ПОЛУЧЕНИЕ ДАННЫХ

```
select '{"key": "value"}'::jsonb ->> 'key';
```

```
select '["string", 1]'::jsonb -> -1;
```

```
select '{  
    "key": {"nested_key": "value"}  
'::jsonb #> '{key, nested_key}'
```

ИЗМЕНЕНИЕ ДАННЫХ

```
select jsonb_set(  
    '{"n":null, "a":{"b": 2}}'::jsonb,  
    '{n}',  
    '[1,2,3]'  
);
```

jsonb_set

```
{"a": {"b": 2}, "n": [1, 2, 3]}  
(1 row)
```

УДАЛЕНИЕ ДАННЫХ

```
select  
  '{"a": {"b": [1, 2, 3]}}'::jsonb  
  #-  
  '{a, b, -1}';  
  ?column?
```

```
  {"a": {"b": [1, 2]}}  
(1 row)
```

```
select jsonb_object_keys(  
    '{"key": "value"}'::jsonb  
);
```

```
    jsonb_object_keys
```

```
    key
```

```
(1 row)
```

```
select jsonb_typeof('1'::jsonb);
```

```
jsonb_typeof
```

```
number
```

```
(1 row)
```


- GIN индекс для @> <@ ?
- jsonb_path_ops
- jsquery: jsonb_path_value, jsonb_path_key

- Содержит ли jsonb объект указанных ключ?
- jquery

КОНВЕРТИРОВАНИЕ

```
select * from test_agg;
```

id	data
1	value1
2	value2

(2 rows)

```
select jsonb_pretty(jsonb_agg(test_agg)) from test_agg;
```

```
      jsonb_pretty
```

```
[
  {
    "id": 1,
    "data": "value1"
  },
  {
    "id": 2,
    "data": "value2"
  }
]
```

(1 row)

```
select array_to_json(  
    ARRAY [  
        jsonb '{"a":1}',  
        jsonb '{"b":[2,3]}'  
    ]  
);  
    array_to_json  
-----  
 [{"a": 1}, {"b": [2, 3]}]  
(1 row)
```

```
update some_table set jsonb_data =  
    jsonb_set(jsonb_data, '{a, a1, a2}', '42');
```

VS

```
update some_table  
    set jsonb_data['a']['a1']['a2'] = 42;
```

MYSQL

MySql 5.7.7, тип данных JSON

ВОЗМОЖНЫЕ ВИДЫ

```
select cast('"string"' as json);
```


ВОЗМОЖНЫЕ ВИДЫ

```
select cast('"string"' as json);
```

```
select cast('["string", 1]' as json);
```

ВОЗМОЖНЫЕ ВИДЫ

```
select cast('"string"' as json);
```

```
select cast('["string", 1]' as json);
```

```
select cast('{"key": {"nested": "value"}}' as json);
```

ПОЛУЧЕНИЕ ДАННЫХ

```
select json_extract('{ "key": "value" }', '$.*');
```

```
select cast('{ "key": "value" }' as json) -> 'key';
```

ИЗМЕНЕНИЕ ДАННЫХ

```
select json_set(  
    '{"n":null, "a":{"b": 2}}',  
    '$.n',  
    '[1,2,3]',  
    '$.a',  
    1  
);
```

```
{"a": 1, "n": "[1,2,3]"}
```

1 row in set (0.01 sec)

УДАЛЕНИЕ ДАННЫХ

```
select json_remove(  
    '{"a": {"b": [1, 2, 3]}}',  
    '$.a.b[2]'  
)
```

```
{"a": {"b": [1, 2]}}  
1 row in set (0.01 sec)
```

```
select json_type('1');
```

```
INTEGER
```

```
1 row in set (0.01 sec)
```

Нет методов для получения ключей, значений и проч.
Есть методы для получения длины или глубины json.

Тип `json` на прямую не индексируется, в качестве `workaround` предлагается создавать `generated` поля с `json_extract`.

- Поиск в пути \$, *
- Поиск по значению json_search

```
select json_search(  
    '{"a": "test", "b": [1, 2, "test2"]}',  
    'all', 'test%'  
);
```

```
["$a", "$b[2]"]  
1 row in set (0.00 sec)
```

ORACLE

Oracle 12.1.0.2, тип данных JSON
Требует **WITH UNIQUE KEYS**

ПОЛУЧЕНИЕ ДАННЫХ

```
SELECT json.document.Nested.key FROM json_data json;
```

```
SELECT json_value(document, '$.Number' RETURNING NUMBER) FROM  
    json_document;
```

ПОЛНОТЕКСТОВЫЙ ПОИСК

```
CREATE INDEX json_search_idx ON json_data (document)
  INDEXTYPE IS CTXSYS.CONTEXT
  PARAMETERS (
    'section group CTXSYS.JSON_SECTION_GROUP SYNC (ON COMMIT)'
  );

SELECT document FROM json_data WHERE json_textcontains(
  document, '$.Array.Description', 'Some description'
);
```

→ AL32UTF8

→ WE8ISO8859P1

```
SELECT * FROM json_data json,  
  json_table(  
    json.document, '$'  
    COLUMNS (  
      json_number NUMBER PATH '$.Number'  
    )  
  );
```

DB2

DB2 11 for z/OS, тип данных JSON/BSON

ПОЛУЧЕНИЕ ДАННЫХ

```
SELECT JSON_VAL(DATA, 'PO.productName', 's:10') FROM JSONPO;
```

```
SYSTOOLS.BSON2JSON(DATA);  
SYSTOOLS.JSON2BSON(DATA);
```

MSSQL

Sql Server 2016, тип данных JSON (внутри NVARCHAR)

```
SELECT JSON_VALUE(jsonInfo, '$.info.address[0].town');
```


```
SELECT * FROM OPENJSON(json, N'$');
```


```
SELECT field1, field2, field3  
FROM table1  
FOR JSON PATH, ROOT("RootKey");
```


ПРОИЗВОДИТЕЛЬНОСТЬ

ЗАКЛЮЧЕНИЕ

CONTACTS

 github.com/erthalion/jsonbx

 @erthalion

 9erthalion6 at gmail dot com

QUESTIONS?