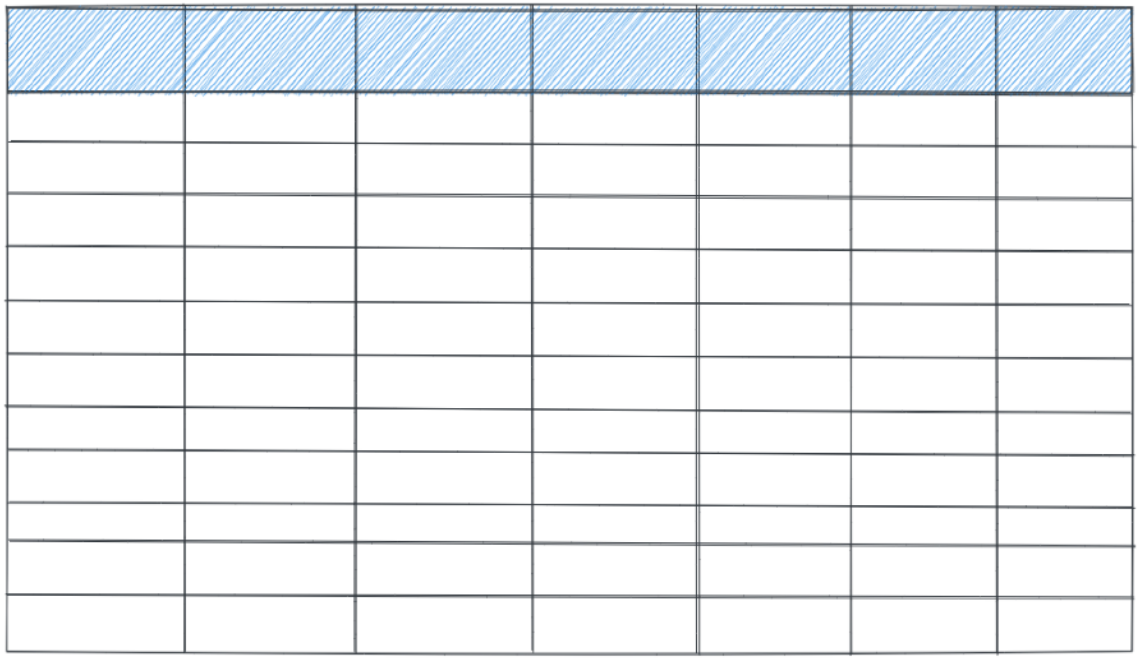
aiven

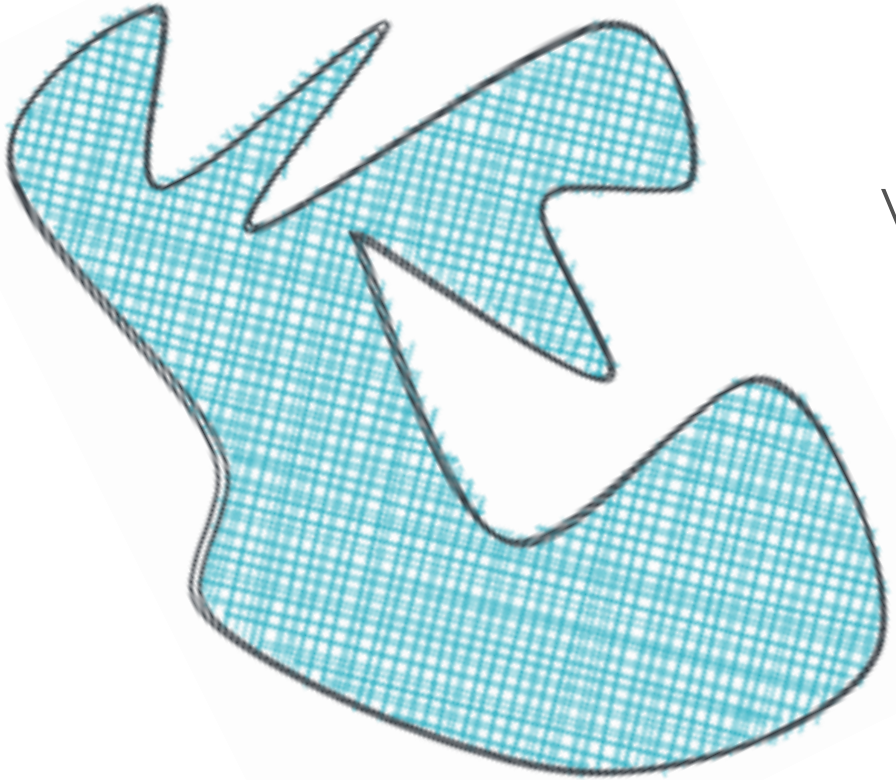
Learn PostgreSQL JSON with Pizza🍕

Francesco Tisiot - Senior Developer Advocate

@ftisiot

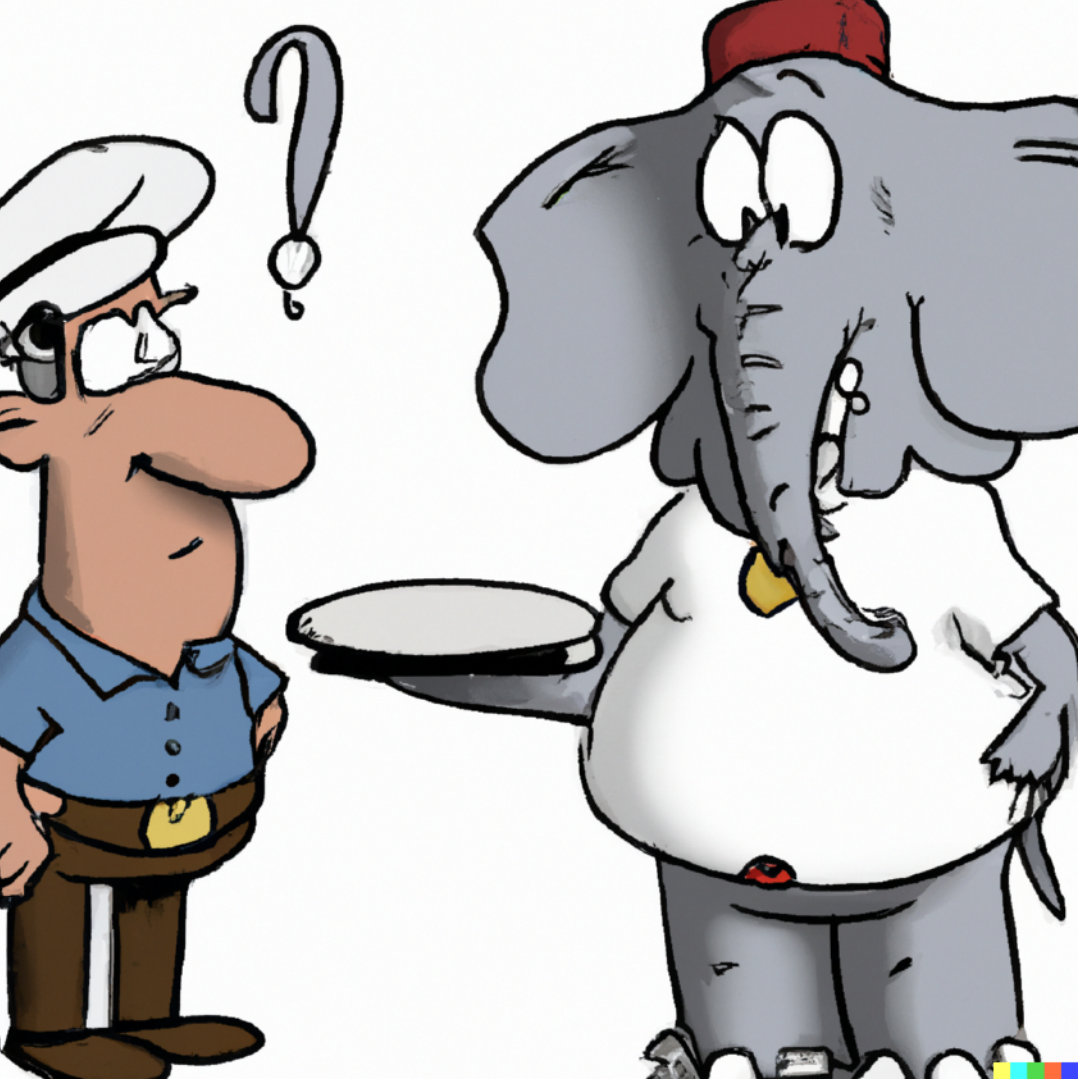
Why?





Variable Payload

Nested Fields



Why in PostgreSQL ?

@ftisiot | @aiven_io

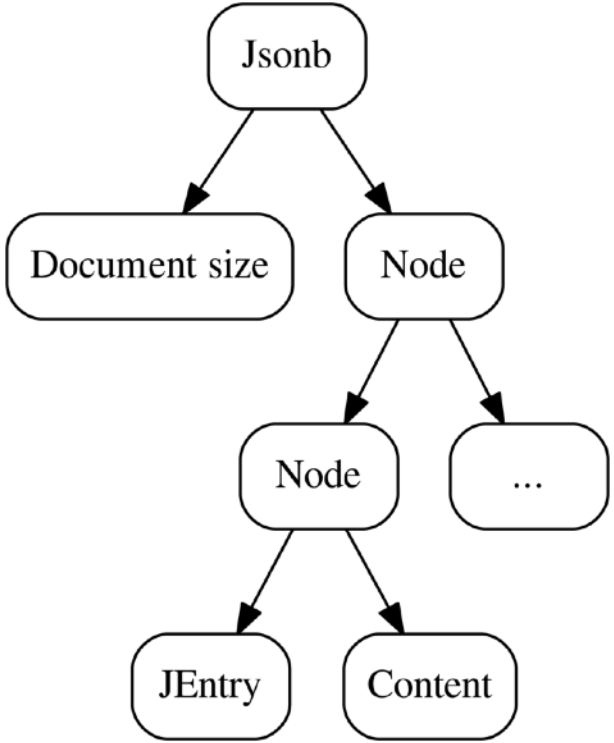
How?

PostgreSQL
JSON
JSONB

JSON	JSONB
Saved as Text	Saved in Binary Format
Minimal JSON Validation	Advanced Computation

https://erthalion.info/2017/12/21/advanced-json-benchmarks/

JSONB Structure



https://ftisiot.net/postgresqljson/what-are-the-differences-json-jsonb-postgresql/

Quality	JSON	JSONB
Data storage format	raw text	custom binary
Write	faster to write, only minimal JSON validation	slower to write due to binary conversion overhead
Read	slower to read, reparsing needed	faster to read
White spaces	preserved	removed
Duplicated keys	preserved	only last valued is kept
Order of keys	preserved	can be altered
Index support	no	yes

JSONB Caveats

❌ Column Stats

⚠️ Larger Footprint

How?

https://ftisiot.net/postgresqljson/main/

This series covers how to solve common problems on JSON datasets with PostgreSQL® and it includes (links will appear once the target pages are up):

Common JSON questions

- What are the differences between JSON or JSONB in PostgreSQL?
- How to load JSON data in PostgreSQL?

Parse JSON

- How to extract a field from a JSON object in PostgreSQL?
- How to get the JSON field types in PostgreSQL?
- How to check if JSON contains in PostgreSQL?
- How does JSON path work in PostgreSQL?
- How to parse JSON arrays in PostgreSQL?
- How to parse JSON keys in PostgreSQL?

Tabulate JSON to record or recordset

- How to tabulate a JSON to a record in PostgreSQL?
- How to tabulate a JSON to a recordset in PostgreSQL?

Convert to JSON

- How to convert a row to JSON in PostgreSQL?
- How to create a JSON object from keys and values arrays in PostgreSQL?
- How to create a JSON object from array of key/value pairs in PostgreSQL?
- How to build a JSON array from a list of elements in PostgreSQL?
- How to convert an array to a JSON array in PostgreSQL?

Prettify JSON output

- How to prettify the JSON output in PostgreSQL?

Edit JSON

```
{
    "id": 778,
    "shop": "Luigis Pizza",
    "name": "Edward Olson",
    "phoneNumbers":
        ["(935)503-3765x4154","(935)12345"],
    "address": "Unit 9398 Box 2056\nDPO AP 24022",
    "image": null,
    "pizzas": [
        {
            "pizzaName": "Salami",
            "additionalToppings": ["🥓", "🌶"]
        },
        {
            "pizzaName": "Margherita",
            "additionalToppings": ["🍌", "🌶", "🍍"]
        }
    ]
}
```

```
create table test(
  id serial,
  json_data jsonb
  );
```

Load Data

```
insert into test(json_data) values (
'{
    "id": 778,
    "shop": "Luigis Pizza",
    "name": "Edward Olson",
    "phoneNumbers":
        ["(935)503-3765x4154","(935)12345"],
    "address": "Unit 9398 Box 2056\nDPO AP 24022",
    "image": null,
    "pizzas": [
        {
            "pizzaName": "Salami",
            "additionalToppings": ["🥓", "🌶"]
        },
        {
            "pizzaName": "Margherita",
            "additionalToppings": ["🍌", "🌶", "🍍"]
        }
    ]
}');
```

Load Data
(From File)

https://stackoverflow.com/questions/43855930/postgresql-json-to-columns-error-character-with-value-must-be-escaped

```
\copy <TABLE_NAME> from program 'sed -e ''s/\\/\\\\/g'' <FILE_NAME>.json';
```

Extract Data

`->` Extract as JSON(B) object

```
select
    json_data -> 'id' id,
    json_data -> 'name' name
    from test;
```

Result

```
 id  |      name
-----+----------------
 778 | "Edward Olson"
```

`->>` Extract as Text

```
select
    json_data ->> 'id' id,
    json_data ->> 'name' order_name
from test;
```

Result:

```
 id  |  order_name
-----+--------------
 778 | Edward Olson
```

Extract from Array

—> Extract an item from an array

```
select
    json_data -> 'pizzas' -> 1 as second_pizza
    from test;
```

In the above query

- `json_data -> 'pizzas'` extracts the `pizzas` field
- the additional `-> 1` extracts the second pizza (index starts from `0`)

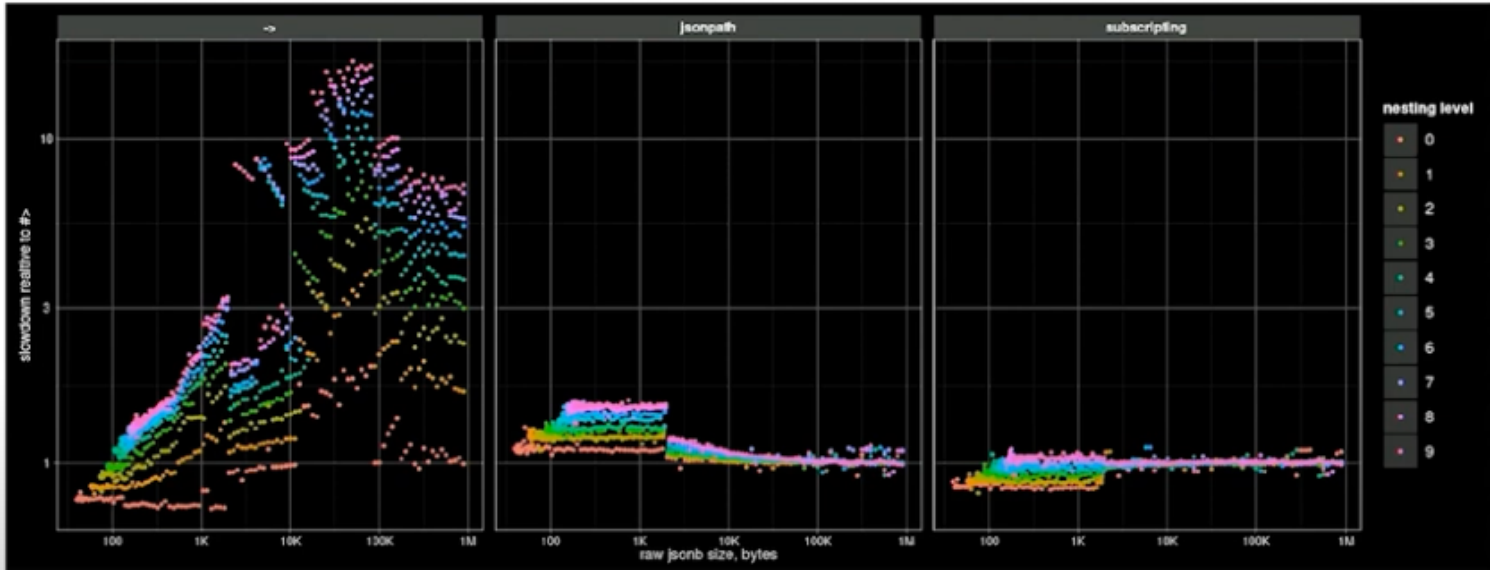
Result

```
                              second_pizza
-----------------------------------------------------------------------
 {"pizzaName": "Margherita", "additionalToppings": ["🍌", "🌶️", "🍗"]}
(1 row)
```

Nested

`->` `...` `->` `...` `->`

Extraction can be nested



https://www.youtube.com/watch?v=v_s-DH4PEVA

JSON path

jsonb_extract_path

```
select
    jsonb_extract_path(json_data, 'pizzas', '1', 'pizzaName') second_pizza_name
from test;
```

Result

```
 second_pizza_name
-------------------
 "Margherita"
(1 row)
```

Subscripting

[]

To get the second pizza name in the order you can use the `[]` operator

```
select
    json_data['pizzas']['1']['pizzaName'] second_pizza_name
from test;
```

Result

```
 second_pizza_name
-------------------
 "Margherita"
(1 row)
```

Tabulate

jsonb_populate_recordset

To tabulate a JSON document to a table recordset defining the list of fields you can use the `json_to_recordset` (`jsonb_to_recordset` in the example for JSONB column). To tabulate the JSON document:

```
select p.* from test
cross join lateral
    jsonb_to_recordset(json_data -> 'pizzas')
    as p("pizzaName" text, "additionalToppings" text[]);
```

Result

```
 pizzaName  | additionalToppings
------------+--------------------
 Salami     | {🐗,🌶}
 Margherita | {🍋,🌶,🍆}
```

Index

GIN Index

Create GIN Index

```
create index json_data_gin on test using gin (json_data);
```

Query Using Index

Retrieve all rows with a value

To retrieve all rows having the `shop` name `Luigis Pizza`, you can use the `@>` operator:

```
select * from test where json_data @> '{"shop":"Luigis Pizza"}';
```

Result

```
 id |
----+-----------------------------------------------------------------------------------------------------
  1 | {"id": 778, "name": "Edward Olson", "shop": "Luigis Pizza", "image": null, "pizzas": [{"pizzaName":
(1 row)
```

Checking with `EXPLAIN`:

```
explain analyse select * from test where json_data @> '{"shop":"Luigis Pizza"}';
```

Result shows the usage of the index:

```
                                                                                                  QUERY PLAN
-------------------------------------------------------------------------------------------------------------------------------------------------------
 Bitmap Heap Scan on test  (cost=41.00..42.01 rows=1 width=394) (actual time=0.398..0.399 rows=1 loops=1)
   Recheck Cond: (json_data @> '{"shop": "Luigis Pizza"}'::jsonb)
   Heap Blocks: exact=1
   ->  Bitmap Index Scan on json_data_gin  (cost=0.00..41.00 rows=1 width=0) (actual time=0.389..0.390 ro
         Index Cond: (json_data @> '{"shop": "Luigis Pizza"}'::jsonb)
 Planning Time: 0.087 ms
 Execution Time: 0.470 ms
(7 rows)
```

⚠️Not all queries work

```
explain analyse select * from test where json_data -> 'pizzas' @> '{"pizzaName":"Margherita"}';
```

You get a sequence scan:

```
                                                    QUERY PLAN
-------------------------------------------------------------------------------------------------------------------
 Seq Scan on test  (cost=0.00..470.13 rows=69 width=394) (actual time=2.064..2.064 rows=0 loops=1)
   Filter: ((json_data -> 'pizzas'::text) @> '{"pizzaName": "Margherita"}'::jsonb)
   Rows Removed by Filter: 6939
 Planning Time: 0.056 ms
 Execution Time: 2.085 ms
(5 rows)
```

Create GIN Index on an item

```
CREATE INDEX pizzas_gin ON test
USING GIN ((json_data -> 'pizzas'));
```

Now Index is used

To filter for a specific item value, like `pizzas` having `pizzaName` equal to `Margherita` you can use the `@>` operator:

```
explain analyse select * from test where json_data -> 'pizzas' @> '{"pizzaName":"Margherita"}';
```

Results

```
 id |
----+-----------------------------------------------------------------------------------------------------
  1 | {"id": 778, "name": "Edward Olson", "shop": "Luigis Pizza", "image": null, "pizzas": [{"pizzaName":
(1 row)
```

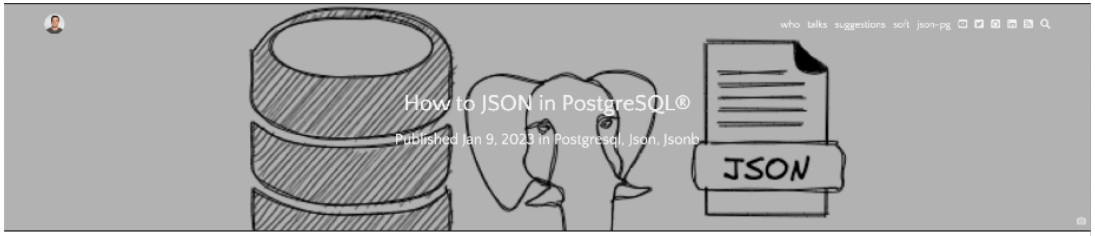
Checking with `EXPLAIN`:

```
explain analyse select * from test where json_data @> '{"pizzas": [{"pizzaName":"Margherita"}]}';
```

Results, the index is used:

```
                                                        QUERY PLAN
-----------------------------------------------------------------------------------------------------------
 Bitmap Heap Scan on test  (cost=43.00..44.01 rows=1 width=394) (actual time=0.493..0.494 rows=1 loops=1)
   Recheck Cond: (json_data @> '{"pizzas": [{"pizzaName": "Margherita"}]}'::jsonb)
   Heap Blocks: exact=1
   ->  Bitmap Index Scan on json_data_gin  (cost=0.00..43.00 rows=1 width=0) (actual time=0.483..0.483 rc
         Index Cond: (json_data @> '{"pizzas": [{"pizzaName": "Margherita"}]}'::jsonb)
 Planning Time: 0.186 ms
 Execution Time: 0.523 ms
(7 rows)
```

https://ftisiot.net/postgresqljson/main/



This series covers how to solve common problems on JSON datasets with PostgreSQL® and it includes (links will appear once the target pages are up):

Common JSON questions

- What are the differences between JSON or JSONB in PostgreSQL?
- How to load JSON data in PostgreSQL?

Parse JSON

- How to extract a field from a JSON object in PostgreSQL?
- How to get the JSON field types in PostgreSQL?
- How to check if JSON contains in PostgreSQL?
- How does JSON path work in PostgreSQL?
- How to parse JSON arrays in PostgreSQL?
- How to parse JSON keys in PostgreSQL?

Tabulate JSON to record or recordset

- How to tabulate a JSON to a record in PostgreSQL?
- How to tabulate a JSON to a recordset in PostgreSQL?

Convert to JSON

- How to convert a row to JSON in PostgreSQL?
- How to create a JSON object from keys and values arrays in PostgreSQL?
- How to create a JSON object from array of key/value pairs in PostgreSQL?
- How to build a JSON array from a list of elements in PostgreSQL?
- How to convert an array to a JSON array in PostgreSQL?

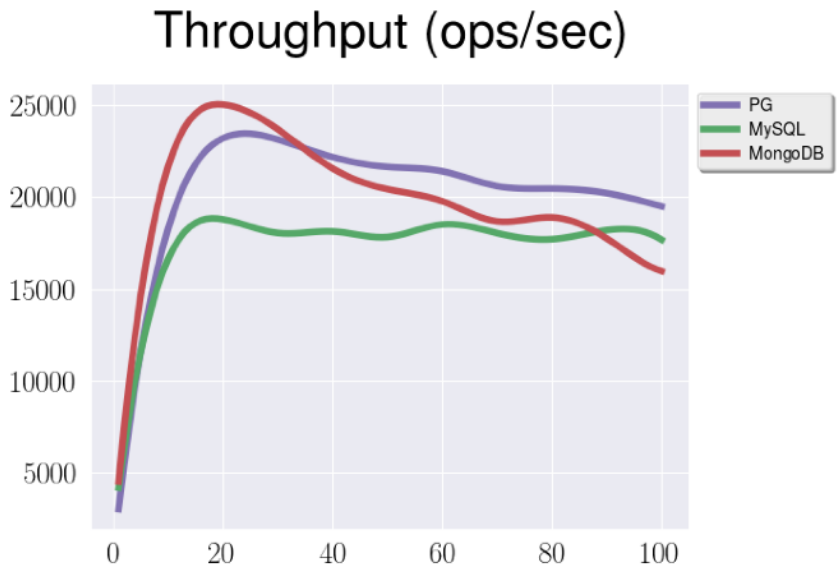
Prettify JSON output

- How to prettify the JSON output in PostgreSQL?

Edit JSON

Benchmarks

https://erthalion.info/2017/12/21/advanced-json-benchmarks/



Code Optimisations

```
SELECT * FROM test_table WHERE data @> '{"key": "123"}'::jsonb;
```

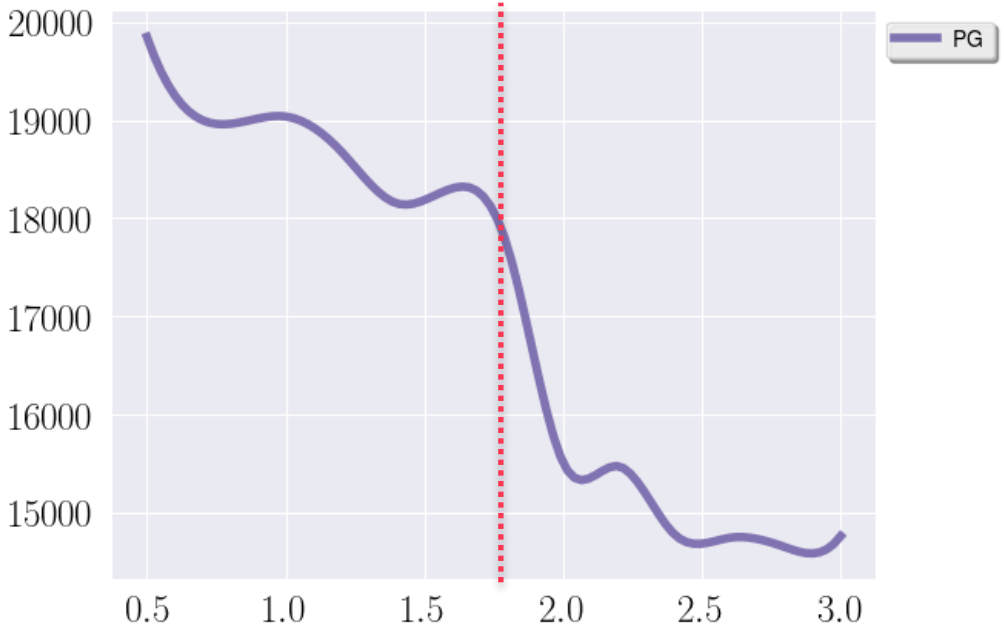
```
SELECT * FROM test_table WHERE data @> jsonb_build_object('key', '123');
```

⚠

TOAST

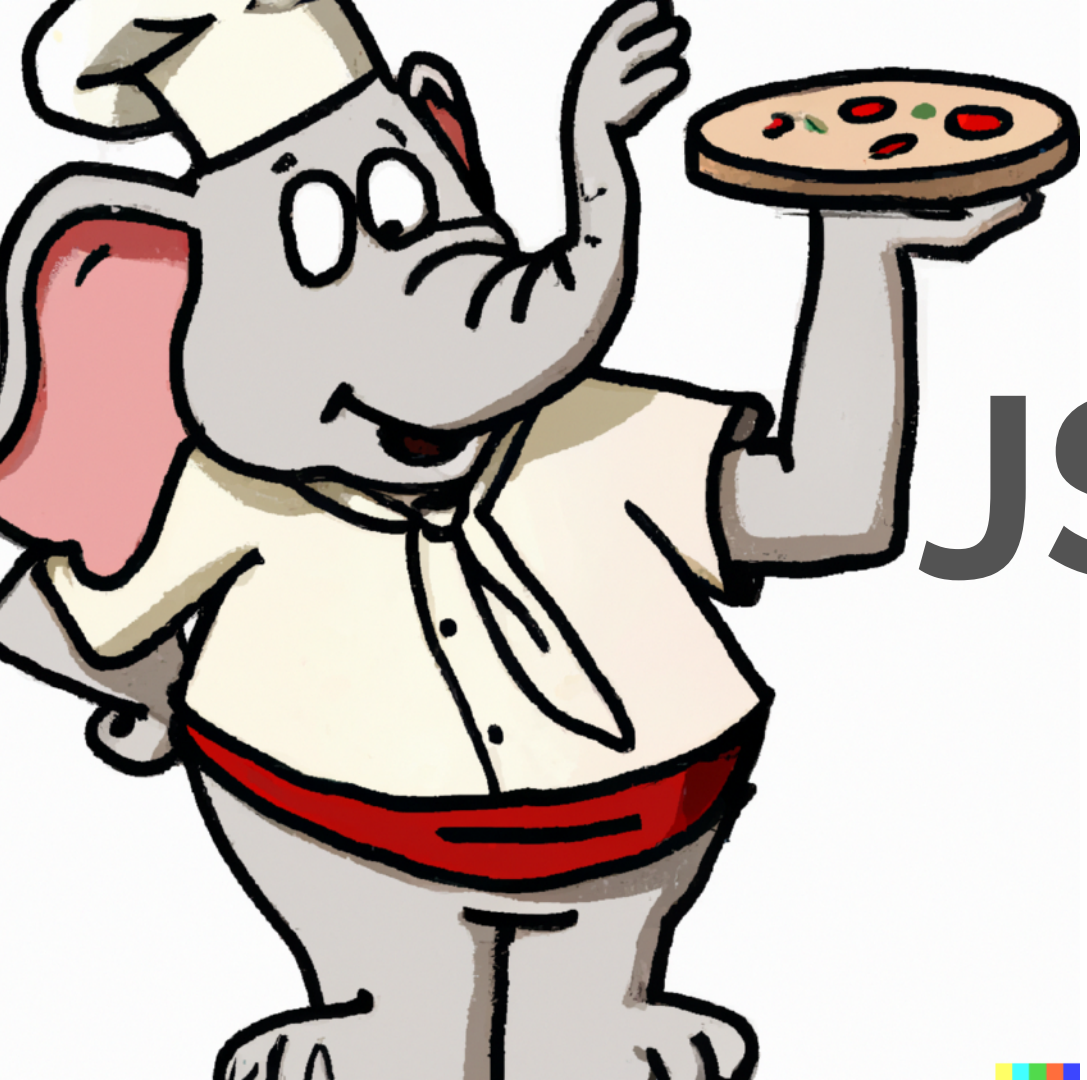
TOAST

~2kB



https://ftisiot.net/postgresqljson/what-are-the-differences-json-jsonb-postgresql/

Quality	JSON	JSONB
Data storage format	raw text	custom binary
Write	faster to write, only minimal JSON validation	slower to write due to binary conversion overhead
Read	slower to read, reparsing needed	faster to read
White spaces	preserved	removed
Duplicated keys	preserved	only last valued is kept
Order of keys	preserved	can be altered
Index support	no	yes



Use JSON(B) Now