PostgreSQL JSON Cheat Sheet

www.databasestar.com

JSON Example

```
{
  “color”: “black”,
  “drawers”: [
    {
      “side”: “left”,
      “height”: “30cm”
    },
    {
      “side”: “left”,
      “height”: “40cm”
    }
  ],
  “material”: “metal”
}
```

Data Types

JSON: regular JSON

JSONB: JSON Binary. The **recommended data type**.

Creating a JSON Field

Create Table with JSONB field:

```
CREATE TABLE product (
  id INT,
  product_name CHARACTER VARYING(200),
  attributes JSONB
);
```

Create Table with JSON field:

```
CREATE TABLE product (
  id INT,
  product_name CHARACTER VARYING(200),
  attributes JSON
);
```

Insert JSON Data

Insert statement:

```
INSERT INTO product (id, product_name, attributes)
VALUES (
1,
'Chair',
'{"color":"brown", "material":"wood",
"height":"60cm"}'
);
```

Insert array:

```
INSERT INTO product (id, product_name, attributes)
VALUES (
3,
'Side Table',
'{"color":"brown", "material":["metal", "wood"]}'
);
```

Insert with JSONB_BUILD_OBJECT:

```
INSERT INTO product (id, product_name, attributes)
VALUES (
4,
'Small Table',
JSONB_BUILD_OBJECT(
  'color', 'black', 'material', 'plastic'
)
);
```

Other functions for inserting:

- TO_JSON and TO_JSONB
- ARRAY_TO_JSON
- ROW_TO_JSON
- JSON_BUILD_ARRAY and JSONB_BUILD_ARRAY
- JSON_OBJECT and JSONB_OBJECT

Selecting

Select with key and value:
(displays a value such as "blue" **with surrounding quotes)**

```
SELECT
id,
product_name,
attributes -> 'color' AS color_key
FROM product;
```

Select with key and value:
(displays a value such as "blue" **without surrounding quotes)**

```
SELECT
id,
product_name,
attributes ->> 'color' AS color_key
FROM product;
```

Select an array value with key and value:

```
SELECT
id,
product_name,
attributes -> 'drawers' -> 1 AS drawer_value
FROM product;
```

Select an array value with key and value as object or as text

```
SELECT
id,
product_name,
attributes #> '{drawers, 1}' AS drawers_element,
attributes #>> '{drawers, 1}' AS drawers_text
FROM product;
```

Filtering

Filtering a value with key and value:

```
SELECT
id,
product_name,
attributes
FROM product
WHERE attributes ->> 'color' = 'brown';
```

Filtering where a key exists:

```
SELECT
id,
product_name,
attributes
FROM product
WHERE attributes ? 'drawers' = true;
```

Split Data into Rows

Split each element into separate rows:

```
SELECT
id,
product_name,
JSONB_EACH(attributes)
FROM product;
```

Get all keys:

```
SELECT
id,
product_name,
JSONB_OBJECT_KEYS(attributes)
FROM product;
```

Updating

Update field by concatenating:

```
UPDATE product
SET attributes =
attributes || '{"width":"100cm"}'
WHERE id = 1;
```

Update field using JSONB_SET:

```
UPDATE product
SET attributes =
JSONB_SET(attributes, '{height}', '"75cm"')
WHERE id = 1;
```

Deleting

Delete based on filter:

```
DELETE FROM product
WHERE attributes ->> 'color' = 'brown';
```

Remove attribute from field:

```
UPDATE product
SET attributes = attributes - 'height'
WHERE id = 1;
```