**Віконні функції**

ALTER TABLE products ADD COLUMN price DECIMAL (11, 2);

UPDATE products SET price = 10000 WHERE id = 2;

**AVG**

SELECT AVG (price) FROM products;

**AVG with GROUP BY**

SELECT c.name, AVG (p.price)

FROM products p

INNER JOIN categroies c ON c.id = p.category\_id

GROUP BY c.name;

SELECT c.name, AVG (p.price)

FROM products p

INNER JOIN categroies c ON c.id = p.category\_id

GROUP BY c.name;

SELECT p.name product\_name, p.price, c.name category\_name, AVG (p.price) OVER ( PARTITION BY c.name)

FROM products p

INNER JOIN categories c ON c.id = p.category\_id;

**WINDOW**

Якщо ми використовуємо декілька віконних функцій у запиті:

SELECT

wf1() OVER(PARTITION BY c1 ORDER BY c2),

wf2() OVER(PARTITION BY c1 ORDER BY c2)

FROM table\_name;

ми можемо використати WINDOWпропозицію, щоб скоротити запит, як показано в наступному запиті:

SELECT

wf1() OVER w,

wf2() OVER w,

FROM table\_name

WINDOW w AS (PARTITION BY c1 ORDER BY c2);

Також можна використовувати WINDOWпропозицію, навіть якщо ви викликаєте одну віконну функцію в запиті:

SELECT wf1() OVER w FROM table\_name

WINDOW w AS (PARTITION BY c1 ORDER BY c2);

**Функція ROW\_NUMBER()**

SELECT

p.name product\_name,

c.name category\_name,

p.price,

ROW\_NUMBER () OVER (

PARTITION BY c.name

ORDER BY

p.price

)

FROM

products p

INNER JOIN categories c ON p.category\_id = c.id;

**Функція RANK()**

SELECT p.name product\_name, c.name category\_name, p.price,

RANK() OVER ( PARTITION BY c.name ORDER BY p.price)

FROM products p

INNER JOIN categories c ON p.category\_id = c.id;

**Функція FIRST\_VALUE()**

SELECT

p.name product\_name,

c.name category\_name,

p.price,

FIRST\_VALUE(price) OVER (

PARTITION BY c.name

ORDER BY p.price)

AS lowest\_price\_per\_group

FROM products p

INNER JOIN categories c ON p.category\_id = c.id;

**JSON**

CREATE TABLE orders (

id GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,

info json NOT NULL

);

INSERT INTO orders (info)

VALUES('{ "customer": "John Doe", "items": {"product": "Beer","qty": 6}}');

INSERT INTO orders (info)

VALUES('{ "customer": "Lily Bush", "items": {"product": "Diaper","qty": 24}}'),

('{ "customer": "Josh William", "items": {"product": "Toy Car","qty": 1}}'),

('{ "customer": "Mary Clark", "items": {"product": "Toy Train","qty": 2}}');

**Оператори JSON**

SELECT info -> 'customer' AS customer

FROM orders;

SELECT info ->> 'customer' AS customer

FROM orders;

SELECT info -> 'items' ->> 'product' as product

FROM orders

ORDER BY product;

SELECT info ->> 'customer' AS customer

FROM orders

WHERE info -> 'items' ->> 'product' = 'Diaper';

SELECT info ->> 'customer' AS customer,

info -> 'items' ->> 'product' AS product

FROM orders

WHERE CAST ( info -> 'items' ->> 'qty' AS INTEGER) = 2;

SELECT

MIN (CAST (info -> 'items' ->> 'qty' AS INTEGER)),

MAX (CAST (info -> 'items' ->> 'qty' AS INTEGER)),

SUM (CAST (info -> 'items' ->> 'qty' AS INTEGER)),

AVG (CAST (info -> 'items' ->> 'qty' AS INTEGER))

FROM orders;

**Функції PostgreSQL JSON**

SELECT json\_each (info)

FROM orders;

SELECT json\_object\_keys (info->'items')

FROM orders;

SELECT json\_typeof (info->'items');

SELECT json\_typeof (info->'items'->'qty')

FROM orders;

**Функції XML**

CREATE TABLE test (y int, x xml, z text);

INSERT INTO test VALUES (1, «<foo>abc<foo>», «abc»), (2, «<bar/», «bar»);

SELECT xmlcomment(z) FROM test;

SELECT xmlpi(name php, „echo «hello world»;);