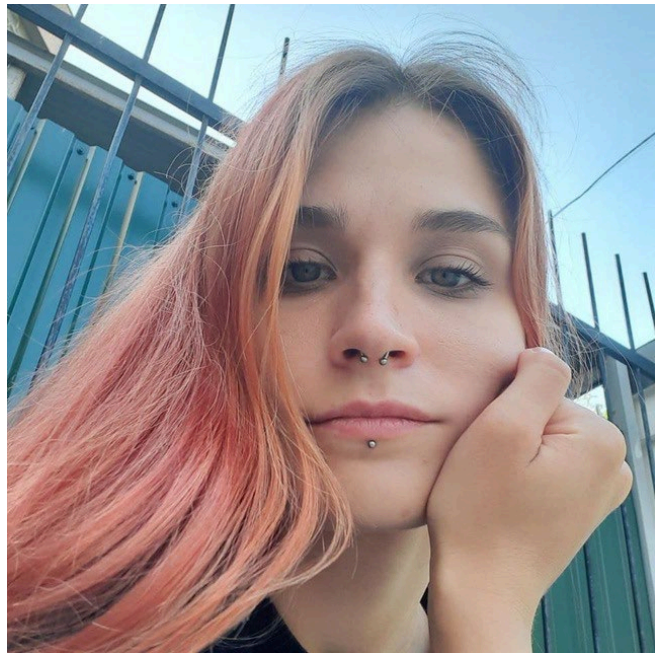
PostgreSQL

Додаткові можливості SQL та PostgreSQL

Introduction



Єрмольонок Яна
Back-end Developer

yana.yermolonok

Тема уроку

Додаткові можливості SQL та PostgreSQL

План уроку

1. Віконні функції.
2. Використання json та xml.

Віконні функції

Function Description	
`row_number` () → `bigint` Returns the number of the current row within its partition, counting from 1.	
`rank` () → `bigint` Returns the rank of the current row, with gaps; that is, the `row_number` of the first row in its peer group.	
`dense_rank` () → `bigint` Returns the rank of the current row, without gaps; this function effectively counts peer groups.	
`percent_rank` () → `double precision` Returns the relative rank of the current row, that is (`rank` - 1) / (total partition rows - 1). The value thus ranges from 0 to 1 inclusive.	
`cume_dist` () → `double precision` Returns the cumulative distribution, that is (number of partition rows preceding or peers with current row) / (total partition rows). The value thus ranges from 1/*N* to 1.	
`ntile` (*num_buckets* `integer`) → `integer` Returns an integer ranging from 1 to the argument value, dividing the partition as equally as possible.	
`lag` (*value* `anycompatible` [, *offset* `integer` [, *default* `anycompatible`]]) → `anycompatible` Returns *value* evaluated at the row that is *offset* rows before the current row within the partition; if there is no such row, instead returns *default* (which must be of a type compatible with *value*). Both *offset* and *default* are evaluated with respect to the current row. If omitted, *offset* defaults to 1 and *default* to `NULL`.	
`lead` (*value* `anycompatible` [, *offset* `integer` [, *default* `anycompatible`]]) → `anycompatible` Returns *value* evaluated at the row that is *offset* rows after the current row within the partition; if there is no such row, instead returns *default* (which must be of a type compatible with *value*). Both *offset* and *default* are evaluated with respect to the current row. If omitted, *offset* defaults to 1 and *default* to `NULL`.	
`first_value` (*value* `anyelement`) → `anyelement` Returns *value* evaluated at the row that is the first row of the window frame.	
`last_value` (*value* `anyelement`) → `anyelement` Returns *value* evaluated at the row that is the last row of the window frame.	
`nth_value` (*value* `anyelement`, *n* `integer`) → `anyelement` Returns *value* evaluated at the row that is the *n*'th row of the window frame (counting from 1); returns `NULL` if there is no such row.	

JSON операторы

Operator Description Example(s)

`json -> integer → json`
`jsonb -> integer → jsonb`
Extracts *n*'th element of JSON array (array elements are indexed from zero, but negative integers count from the end).
`'[{"a":"foo"},{"b":"bar"},{"c":"baz"}]'::json -> 2 → {"c":"baz"}`
`'[{"a":"foo"},{"b":"bar"},{"c":"baz"}]'::json -> -3 → {"a":"foo"}`

`json -> text → json`
`jsonb -> text → jsonb`
Extracts JSON object field with the given key.
`'{"a": {"b":"foo"}}'::json -> 'a' → {"b":"foo"}`

`json ->> integer → text`
`jsonb ->> integer → text`
Extracts *n*'th element of JSON array, as `text`.
`'[1,2,3]'::json ->> 2 → 3`

`json ->> text → text`
`jsonb ->> text → text`
Extracts JSON object field with the given key, as `text`.
`'{"a":1,"b":2}'::json ->> 'b' → 2`

`json #> text[] → json`
`jsonb #> text[] → jsonb`
Extracts JSON sub-object at the specified path, where path elements can be either field keys or array indexes.
`'{"a": {"b": ["foo","bar"]}}'::json #> '{a,b,1}' → "bar"`

`json #>> text[] → text`
`jsonb #>> text[] → text`
Extracts JSON sub-object at the specified path as `text`.
`'{"a": {"b": ["foo","bar"]}}'::json #>> '{a,b,1}' → bar`

Функції створення json

- to_json(anyelement)
- array_to_json(anyarray[,boolean])
- row_to_json(record[,boolean])
- json_build_array(VARIADIC "any")
- json_build_object(VARIADIC "any")
- json_object(text[])
- json_object(,)

Функції створення Xml

- Xmlcomment
- Xmlconcat
- Xmlelement
- Xmlforest
- Xmlpi
- Xmlroot
- Xmlagg

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
    <to> Tove</to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don't forget me this weekend!</body>
</note>
```

Підсумки уроку

1. Віконні функції (window functions) дозволяють проводити маніпуляції між рядками, які повертаються одним SQL-запитом.
2. Типи Json та Xml для зручного зберігання даних.

TestProvider Pearson CyberBionic systematics Microsoft Partner Windows Azure EBA

Перевірка знань

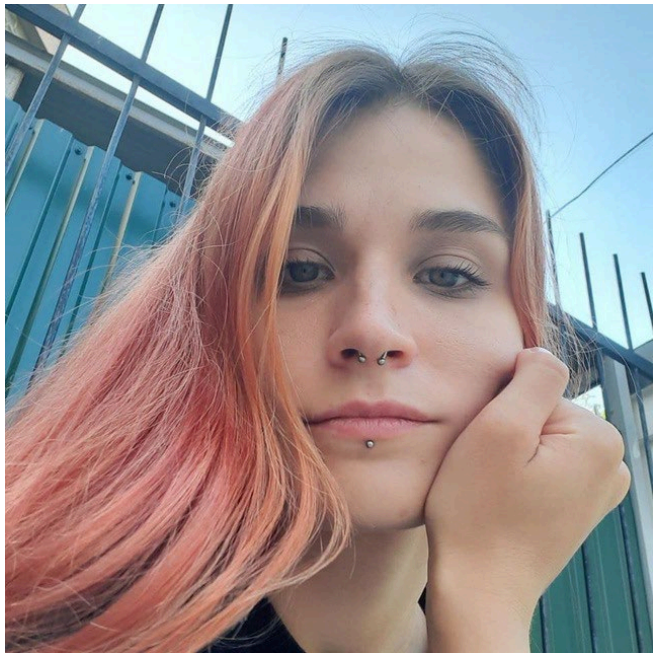
TestProvider.com

TestProvider – це online-сервіс перевірки знань з інформаційних технологій. За його допомогою ви можете оцінити свій рівень та виявити слабкі місця. Він буде корисним як у процесі вивчення технології, так і для загальної оцінки знань IT-спеціаліста.

Перевірте, як ви засвоїли даний матеріал на TestProvider.com

Успішне проходження фінального тестування дозволить вам отримати відповідний Сертифікат.

Дякую за увагу! До нових зустрічей!



Єрмольонок Яна
Back-end Developer