SELECT foo FROM regexp\_split\_to\_table('the quick brown fox jumped over the lazy dog', E'\\\s+') AS foo;

foo

--------

the

quick

brown

fox

jumped

over

the

lazy

dog

(9 rows

SELECT foo FROM regexp\_split\_to\_table('the quick brown fox', E'\\s\*') AS foo;

foo

-----

t

h

e

q

u

i

c

k

b

r

o

w

n

f

o

x

(16 rows)

……………………………………………………………………………………………………………………………………

**PostgreSQL LIKE operator**

SELECT

   'foo' LIKE 'foo', -- true

   'foo' LIKE 'f%', -- true

   'foo' LIKE '\_o\_', -- true

   'bar' LIKE 'b\_'; -- false

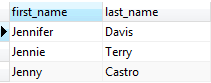
Create table customer(first\_name char(30) not null, last\_name char(30));

insert into customer values('Jenifer', 'Davis'),

('Jennie','Terry'),('Jenny','Castro'),

('Kinberly','Lee'), ('Teresa','Rogers'),('Sherry','Rhodes');

Select \* from customer;



SELECT

   first\_name,

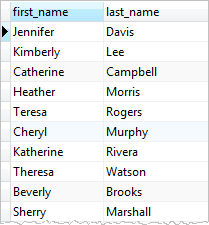
        last\_name

FROM

   customer

WHERE

   first\_name LIKE '%er%'



SELECT

   first\_name,

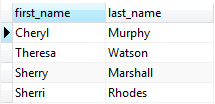
   last\_name

FROM

   customer

WHERE

   first\_name LIKE '\_her%';



SELECT

   first\_name,

   last\_name

FROM

   customer

WHERE

   first\_name NOT LIKE 'Jen%';

## PostgreSQL’s extensions of LIKE operator

PostgreSQL provides the ILIKE operator that acts like the LIKE operator. In addition, the ILIKE operator matches value case-insensitively

SELECT

   first\_name,

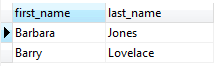
   last\_name

FROM

   customer

WHERE

   first\_name ILIKE 'BAR%';



PostgreSQL also provides some operators that act like the LIKE, NOT LIKE, ILIKE and NOT ILIKE operator as shown below:

* ~~ is equivalent to LIKE
* ~~\* is equivalent to ILIKE
* !~~ is equivalent to NOT LIKE
* !~~\* is equivalent to NOT ILIKE

| **Comparison Operator** | **Description** |
| --- | --- |
| = | Equal |
| <> | Not Equal |
| != | Not Equal |
| > | Greater Than |
| >= | Greater Than or Equal |
| < | Less Than |
| <= | Less Than or Equal |
| [IN ( )](https://www.techonthenet.com/postgresql/in.php) | Matches a value in a list |
| [NOT](https://www.techonthenet.com/postgresql/not.php) | Negates a condition |
| [BETWEEN](https://www.techonthenet.com/postgresql/between.php) | Within a range (inclusive) |
| [IS NULL](https://www.techonthenet.com/postgresql/is_null.php) | NULL value |
| [IS NOT NULL](https://www.techonthenet.com/postgresql/is_not_null.php) | Non-NULL value |
| [LIKE](https://www.techonthenet.com/postgresql/like.php) | Pattern matching with % and \_ |
| [EXISTS](https://www.techonthenet.com/postgresql/exists.php) | Condition is met if subquery returns at least one row |

SELECT \*

FROM employees

WHERE employee\_id BETWEEN 200 AND 300;

SELECT \*

FROM employees

WHERE start\_date BETWEEN '2014-04-01' AND '2014-04-30';

SELECT \*

FROM employees

WHERE employee\_id NOT BETWEEN 500 AND 599;

…………………………………………………………………………………………………………

## CASE

The SQL CASE expression is a generic conditional expression, similar to if/else statements in other programming languages:

CASE WHEN condition THEN result

[WHEN ...]

[ELSE result]

END

## Create table test(a int not null);

## insert into test values(1),(2),(3),(4);

## SELECT \* FROM test;

a

---

1

2

3

SELECT a,

CASE WHEN a=1 THEN 'one'

WHEN a=2 THEN 'two'

ELSE 'other'

END

FROM test;

a | case

---+-------

1 | one

2 | two

3 | other