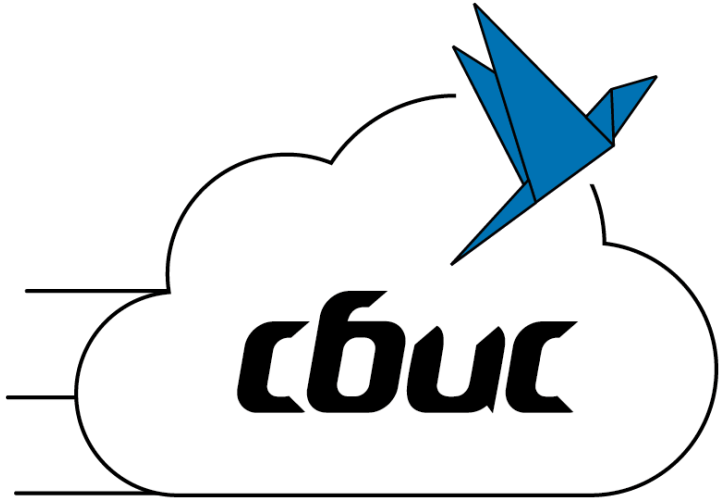


PostgreSQL для начинающих

#2: Простые SELECT

Кирилл Боровиков / Компания «Тензор», технический директор / explain.tensor.ru, sbis.ru

SELECT – это просто!

```
SELECT
   *
FROM
   имя_таблицы;
```

```
TABLE имя_таблицы;
```

https://postgrespro.ru/docs/postgresql/15/sql-select#SQL-TABLE

SELECT – это сложно...

```
[ WITH [ RECURSIVE ] запрос_WITH [, ...] ]
SELECT [ ALL | DISTINCT [ ON ( выражение [, ...] ) ] ]
    [ * | выражение [ [ AS ] имя_результата ] [, ...] ]
    [ FROM элемент_FROM [, ...] ]
    [ WHERE условие ]
    [ GROUP BY [ ALL | DISTINCT ] элемент_группирования [, ...] ]
    [ HAVING условие ]
    [ WINDOW имя_окна AS ( определение_окна ) [, ...] ]
    [ { UNION | INTERSECT | EXCEPT } [ ALL | DISTINCT ] выборка ]
    [ ORDER BY выражение [ ASC | DESC | USING оператор ] [ NULLS { FIRST | LAST } ] [, ...] ]
    [ LIMIT { число | ALL } ]
    [ OFFSET начало [ ROW | ROWS ] ]
    [ FETCH { FIRST | NEXT } [ число ] { ROW | ROWS } { ONLY | WITH TIES } ]
    [ FOR { UPDATE | NO KEY UPDATE | SHARE | KEY SHARE } [ OF имя_таблицы [, ...] ] [ NOWAIT | SKIP LOCKED ] [...] ]
```

https://postgrespro.ru/docs/postgresql/15/sql-select

https://postgrespro.ru/docs/postgresql/15/queries

SELECT – это сложно...

```
[ WITH [ RECURSIVE ] запрос_WITH [, ...] ]
SELECT [ ALL | DISTINCT [ ON ( выражение [, ...] ) ] ]
    [ * | выражение [ [ AS ] имя_результата ] [, ...] ]
    [ FROM элемент_FROM [, ...] ]
    [ WHERE условие ]
    [ GROUP BY [ ALL | DISTINCT ] элемент_группирования [, ...] ] -- PostgreSQL 14
    [ HAVING условие ]
    [ WINDOW имя_окна AS ( определение_окна ) [, ...] ]
    [ { UNION | INTERSECT | EXCEPT } [ ALL | DISTINCT ] выборка ]
    [ ORDER BY выражение [ ASC | DESC | USING оператор ] [ NULLS { FIRST | LAST } ] [, ...] ]
    [ LIMIT { число | ALL } ]
    [ OFFSET начало [ ROW | ROWS ] ]
    [ FETCH { FIRST | NEXT } [ число ] { ROW | ROWS } { ONLY | WITH TIES } ] -- PostgreSQL 13
    [ FOR { UPDATE | NO KEY UPDATE | SHARE | KEY SHARE } [ OF имя_таблицы [, ...] ] [ NOWAIT | SKIP LOCKED ] [...] ]
```

https://postgrespro.ru/docs/postgresql/15/sql-select

https://postgrespro.ru/docs/postgresql/15/queries

VALUES

```
VALUES ( выражение [, ...] ) [, ...]
    [ ORDER BY выражение_сортировки [ ASC | DESC | USING оператор ] [, ...] ]
    [ LIMIT { число | ALL } ]
    [ OFFSET начало [ ROW | ROWS ] ]
    [ FETCH { FIRST | NEXT } [ число ] { ROW | ROWS } ONLY ]
```

https://postgrespro.ru/docs/postgresql/15/sql-values

VALUES

```
VALUES (1);
```

```
column1
integer
      1
```

```
VALUES (1, 2);
```

```
column1 | column2
integer | integer
      1 |       2
```

```
VALUES (1), (2);
```

```
column1
integer
      1
      2
```

```
VALUES
  (1, 2.5, 'alpha')
, (4, 5.5, 'beta');
```

```
column1 | column2 | column3
integer | numeric | text
      1 |     2.5 | alpha
      4 |     5.5 | beta
```

```
VALUES
  (1, 2.5, 'alpha')
, (4, 5.5);
```

```
ERROR:  VALUES lists must all be the same length
LINE 3: , (4, 5.5)
```

VALUES

```
VALUES
  (1, 2.5e+0, 'al' || 'pha')
, (2 + 2, NULL, 'beta');
```

```
column1 | column2 | column3
integer | numeric | text
      1 |     2.5 | alpha
      4 |         | beta
```

Операторы в PostgreSQL

Имя оператора образует последовательность не более чем NAMEDATALEN-1 (по умолчанию 63) символов из следующего списка:

+ - * / < > = ~ ! @ # % ^ & | ` ?

Однако для имён операторов есть ещё несколько ограничений:

- Сочетания символов **--** и **/*** не могут присутствовать в имени оператора, так как они будут обозначать начало комментария.
- Многосимвольное имя оператора не может заканчиваться знаком **+** или **-**, если только оно не содержит также один из этих символов: **~ ! @ # % ^ & | ` ?**
  Например, **@-** — допустимое имя оператора, а ***-** — нет. Благодаря этому ограничению, PostgreSQL может разбирать корректные SQL-запросы без пробелов между компонентами.

https://postgrespro.ru/docs/postgresql/15/sql-syntax-lexical#SQL-SYNTAX-OPERATORS

Операторы в PostgreSQL

```
VALUES (|/ 36, ||/ 125, @ -1, 'al' || 'pha');
```

```
column1          | column2          | column3 | column4
double precision | double precision | integer | text
               6 |                5 |       1 | alpha
```

```
VALUES (sqrt(36), cbrt(125), abs(-1), concat('al', 'pha'));
```

https://postgrespro.ru/docs/postgresql/15/functions-math
https://postgrespro.ru/docs/postgresql/15/typeconv-oper

Операторы в PostgreSQL

Оператор/элемент	Очерёдность	Описание
`.`	слева-направо	разделитель имен таблицы и столбца
`::`	слева-направо	приведение типов в стиле PostgreSQL
`[ ]`	слева-направо	выбор элемента массива
`+ -`	справа-налево	унарный плюс, унарный минус
`^`	слева-направо	возведение в степень
`* / %`	слева-направо	умножение, деление, остаток от деления
`+ -`	слева-направо	сложение, вычитание
(любой другой оператор)	слева-направо	все другие встроенные и пользовательские операторы
`BETWEEN IN LIKE ILIKE SIMILAR`		проверка диапазона, проверка членства, сравнение строк
`< > = <= >= <>`		операторы сравнения
`IS ISNULL NOTNULL`		`IS TRUE`, `IS FALSE`, `IS NULL`, `IS DISTINCT FROM` и т. д.
`NOT`	справа-налево	логическое отрицание
`AND`	слева-направо	логическая конъюнкция
`OR`	слева-направо	логическая дизъюнкция

https://postgrespro.ru/docs/postgresql/15/sql-syntax-lexical#SQL-PRECEDENCE

ORDER BY

```
ORDER BY
  выражение -- номер столбца выборки, имя поля или выражение
    [ ASC | DESC | USING оператор ] -- порядок сортировки [ASC]
    [ NULLS { FIRST | LAST } ] -- куда относить NULL-значения [ASC LAST, DESC FIRST]
[, ...]
```

https://postgrespro.ru/docs/postgresql/15/sql-select#SQL-ORDERBY

https://postgrespro.ru/docs/postgresql/15/queries-order

ORDER BY

```
VALUES
  (1, 2, 1)
, (2, 1, 2)
, (1, 1, 3);
```

```
column1 | column2 | column3
integer | integer | integer
      1 |       2 |       1
      2 |       1 |       2
      1 |       1 |       3
```

```
VALUES
  (1, 2, 1)
, (2, 1, 2)
, (1, 1, 3)
ORDER BY
  1; -- сортировка по первому столбцу
```

```
column1 | column2 | column3
integer | integer | integer
      1 |       2 |       1
      1 |       1 |       3
      2 |       1 |       2
```

ORDER BY

```
VALUES
  (1, 2, 1)
, (2, 1, 2)
, (1, 1, 3)
ORDER BY
  1 ASC; -- по возрастанию
```

```
column1 | column2 | column3
integer | integer | integer
      1 |       1 |       3
      1 |       2 |       1
      2 |       1 |       2
```

```
VALUES
  (1, 2, 1)
, (2, 1, 2)
, (1, 1, 3)
ORDER BY
  1 DESC; -- по убыванию
```

```
column1 | column2 | column3
integer | integer | integer
      2 |       1 |       2
      1 |       1 |       3
      1 |       2 |       1
```

ORDER BY

```
VALUES
  (1, 2, 1)
, (2, 1, 2)
, (1, 1, 3)
ORDER BY
  1 ASC, 2 ASC; -- по возрастанию #1, #2
```

```
column1 | column2 | column3
integer | integer | integer
      1 |       1 |       3
      1 |       2 |       1
      2 |       1 |       2
```

```
VALUES
  (1, 2, 1)
, (2, 1, 2)
, (1, 1, 3)
ORDER BY
  1 DESC, 2 DESC; -- по убыванию #1, #2
```

```
column1 | column2 | column3
integer | integer | integer
      2 |       1 |       2
      1 |       2 |       1
      1 |       1 |       3
```

ORDER BY

```
VALUES
  (1, 2, 1)
, (2, 1, 2)
, (1, 1, 3)
ORDER BY
  column1 + column2 -- сложное выражение
, column3; -- второе выражение
```

```
column1 | column2 | column3
integer | integer | integer
      1 |       1 |       3 -- 1+1= 2
      1 |       2 |       1 -- 1+2= 3
      2 |       1 |       2 -- 2+1= 3
```

```
VALUES
  (1, 2, 1)
, (2, 1, 2)
, (1, 1, 3)
ORDER BY
  (column1, column2); -- тоже выражение
```

```
column1 | column2 | column3
integer | integer | integer
      1 |       1 |       3 -- (1,1)
      1 |       2 |       1 -- (1,2)
      2 |       1 |       2 -- (2,1)
```

ORDER BY

```
VALUES (1), (NULL), (2)
ORDER BY 1 ASC; -- [NULLS LAST]
```

```
column1
integer
      1
      2
      -
```

```
VALUES (1), (NULL), (2)
ORDER BY 1 ASC NULLS FIRST;
```

```
column1
integer
      -
      1
      2
```

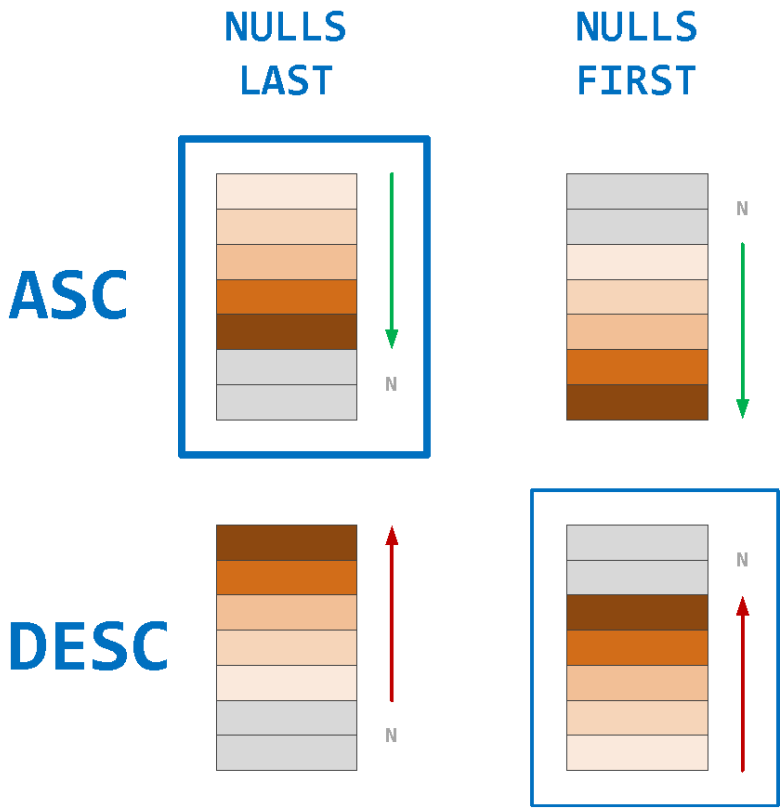
```
VALUES (1), (NULL), (2)
ORDER BY 1 DESC NULLS LAST;
```

```
column1
integer
      2
      1
      -
```

```
VALUES (1), (NULL), (2)
ORDER BY 1 DESC; -- [NULLS FIRST]
```

```
column1
integer
      -
      2
      1
```

ORDER BY



ORDER BY

```
VALUES
  ('ёж')
, ('ель')
, ('ял')
, ('alpha')
, ('Яков')
ORDER BY
  column1; -- [USING <]
```

```
alpha
ель
ёж
Яков
ял
```

```
VALUES
  ('ёж')
, ('ель')
, ('ял')
, ('alpha')
, ('Яков')
ORDER BY
  column1 USING ~<~;
```

```
alpha
Яков
ель
ял
ёж
```

ORDER BY

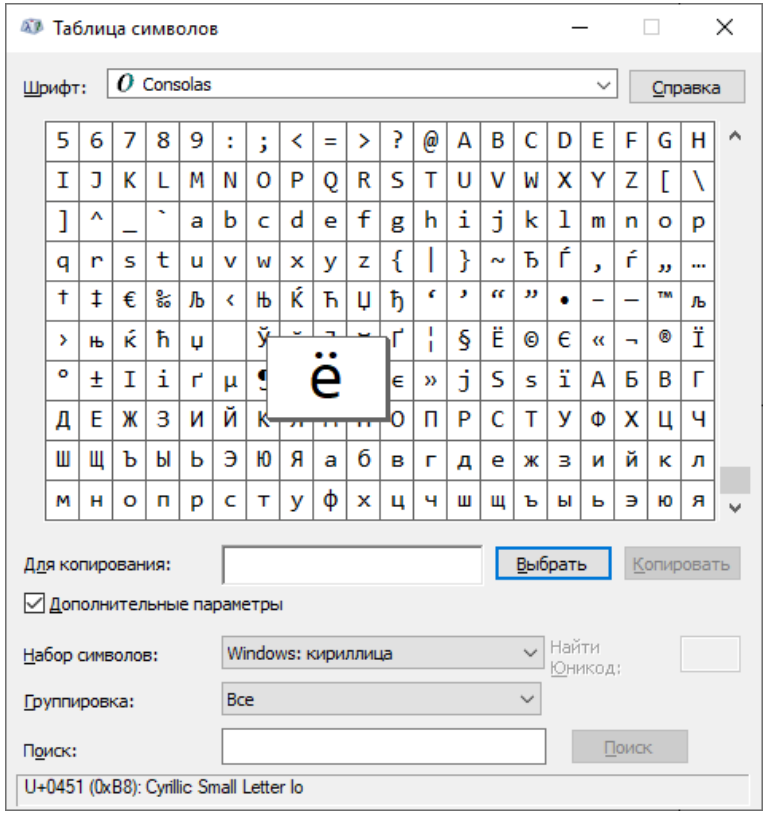


Таблица символов
Шрифт:
Consolas
Справка
ё
Для копирования:
Выбрать
Копировать
Дополнительные параметры
Набор символов:
Windows: кириллица
Найти Юникод:
Группировка:
Все
Поиск:
Поиск
U+0451 (0xB8): Cyrillic Small Letter Io

ORDER BY

```
VALUES
  ('ёж')
, ('ель')
, ('ял')
, ('alpha')
, ('Яков')
ORDER BY
  column1 COLLATE "C";
```

```
alpha
Яков
ель
ял
ёж
```

```
VALUES
  ('ёж')
, ('ель')
, ('ял')
, ('alpha')
, ('Яков')
ORDER BY
  column1 USING ~<~;
```

```
alpha
Яков
ель
ял
ёж
```

https://postgrespro.ru/docs/postgresql/15/collation

LIMIT

```
[ LIMIT { число | ALL } ]
[ OFFSET начало [ ROW | ROWS ] ]
[ FETCH { FIRST | NEXT } [ число ] { ROW | ROWS } { ONLY | WITH TIES } ]
```

https://postgrespro.ru/docs/postgresql/15/sql-select#SQL-LIMIT

https://postgrespro.ru/docs/postgresql/15/queries-limit

LIMIT

```
VALUES
  ('a')
, ('b')
, ('c')
LIMIT ALL; -- или LIMIT NULL
```

```
a
b
c
```

```
VALUES
  ('a')
, ('b')
, ('c')
LIMIT 2; -- или даже '2'
```

```
a
b
```

OFFSET

```
VALUES
  ('a')
, ('b')
, ('c')
OFFSET 0; -- или OFFSET NULL
```

```
a
b
c
```

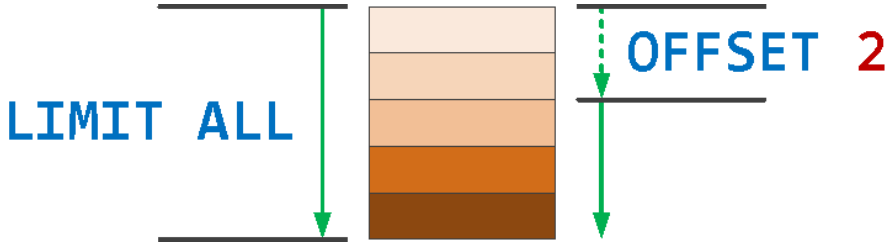
```
VALUES
  ('a')
, ('b')
, ('c')
OFFSET 1; -- или даже '1'
```

```
b
c
```

OFFSET

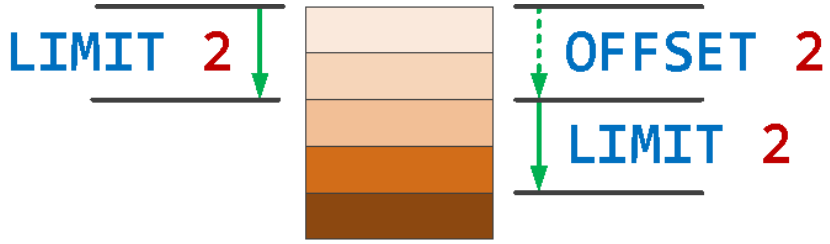
```
VALUES
  ('a')
, ('b')
, ('c')
OFFSET 1;
```

```
b
c
```




```
VALUES
  ('a')
, ('b')
, ('c')
LIMIT 1
OFFSET 1;
```

```
b
```



FETCH

```
VALUES
  ('a')
, ('b')
, ('c')
LIMIT 1
OFFSET 1;
```

```
b
```

```
VALUES
  ('a')
, ('b')
, ('c')
OFFSET 1 ROWS
FETCH FIRST 1 ROWS ONLY;
```

```
b
```

FETCH

```
VALUES
  ('a', 1)
, ('b', 2)
, ('a', 3)
, ('b', 4)
ORDER BY 1
LIMIT 1;
```

```
column1 | column2
text    | integer
      a |       1
```

```
VALUES
  ('a', 1)
, ('b', 2)
, ('a', 3)
, ('b', 4)
ORDER BY 1
FETCH FIRST 1 ROWS ONLY;
```

```
column1 | column2
text    | integer
      a |       1
```

FETCH

```
VALUES
  ('a', 1)
, ('b', 2)
, ('a', 3)
, ('b', 4)
ORDER BY 1
FETCH FIRST 1 ROWS
  ONLY;
```

```
column1 | column2
text    | integer
      a |       1
```

```
VALUES
  ('a', 1)
, ('b', 2)
, ('a', 3)
, ('b', 4)
ORDER BY 1
FETCH FIRST 1 ROWS
  WITH TIES; -- !!!
```

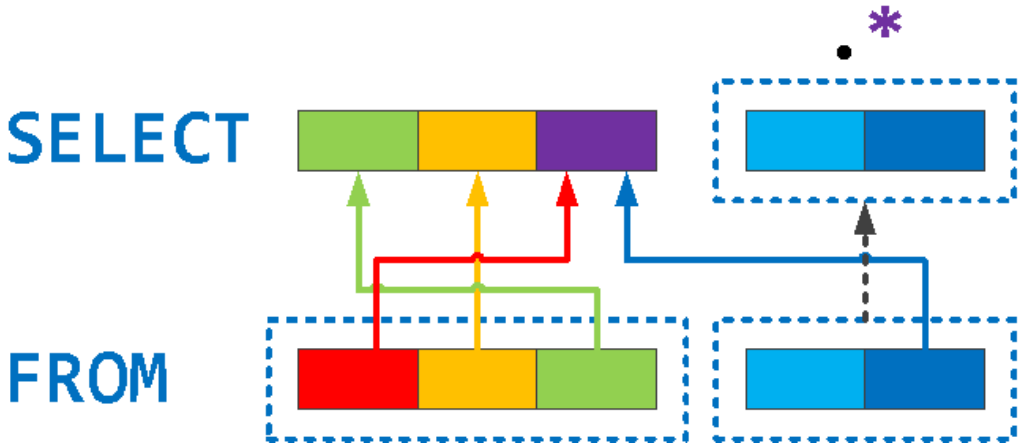
```
column1 | column2
text    | integer
      a |       1
      a |       3
```

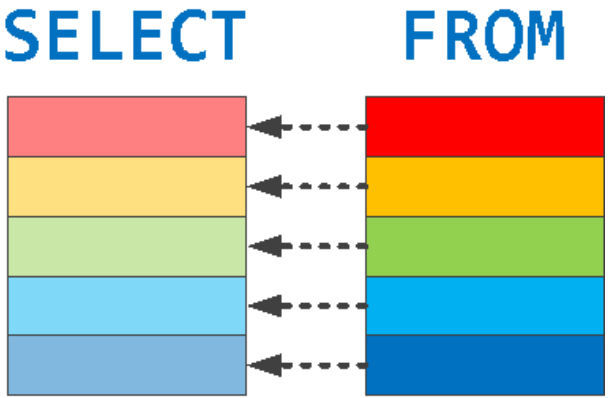
https://habr.com/ru/companies/tensor/articles/520294/

SELECT

```
SELECT
  [ * | выражение [ [ AS ] имя_результата ] [, ...] ]
[ FROM элемент_FROM [, ...] ]
```

https://postgrespro.ru/docs/postgresql/15/sql-select#SQL-SELECT-LIST





SELECT

```
SELECT
  'a'
, 1;
```

```
?column? | ?column?
text     | integer
       a |       1
```

```
SELECT
  'a' AS s -- AS писать необязательно
, 1 i;
```

```
s    | i
text | integer
   a |       1
```

```
SELECT
  random();
```

```
random
double precision
0.35806417754693065
```

```
SELECT
  generate_series(1, 3) i;
```

```
i
integer
      1
      2
      3
```

SELECT

```
SELECT
  * -- все столбцы FROM
, T.* -- столбцы FROM-элемента
, T.column1 str -- конкретный столбец
FROM (
  VALUES
    ('a', 1)
  , ('b', 2)
  , ('a', 3)
  , ('b', 4)
) T; -- тут тоже можно не писать AS
```

column1	column2	column1	column2	str
text	integer	text	integer	text
a	1	a	1	a
b	2	b	2	b
a	3	a	3	a
b	4	b	4	b

SELECT

```sql
SELECT
  T.*
FROM (
  VALUES
    ('a', 1)
  , ('b', 2)
  , ('a', 3)
  , ('b', 4)
) T(str, i);
```

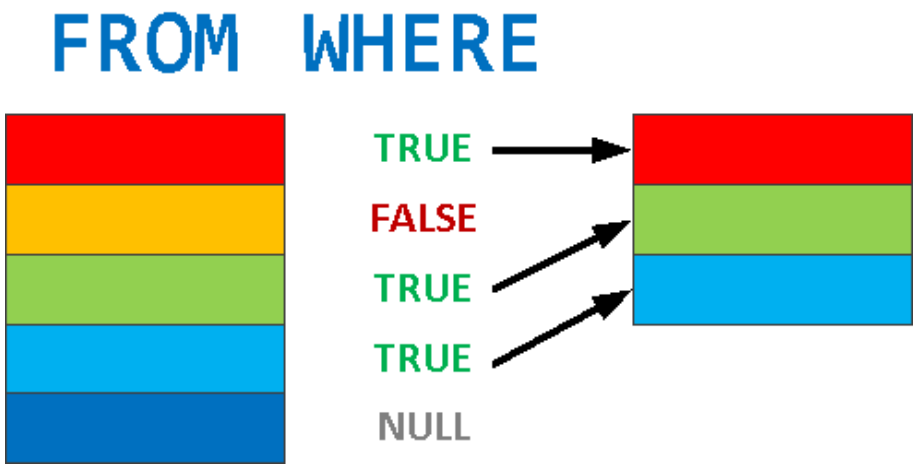
```
str  | i
text | integer
   a |       1
   b |       2
   a |       3
   b |       4
```

WHERE

```
WHERE условие
```

https://postgrespro.ru/docs/postgresql/15/sql-select#SQL-WHERE

https://postgrespro.ru/docs/postgresql/15/queries-table-expressions#QUERIES-WHERE



WHERE

```
SELECT
  *
FROM (
  VALUES
    ('a', 1)
  , ('b', 2)
  , ('a', 3)
  , ('b', 4)
) T(str, i)
WHERE
  str = 'a';
```

```
column1 | column2
text    | integer
      a |       1
      a |       3
```

WHERE

```
WHERE условие1 AND условие2
```

WHERE

WHERE *условие1* AND *условие2*

https://habr.com/ru/companies/tensor/articles/494776/

Заметьте, что это отличается от «оптимизации» вычисления логических операторов слева направо, реализованной в некоторых языках программирования.

Как следствие, в сложных выражениях не стоит использовать функции с побочными эффектами. Особенно опасно рассчитывать на порядок вычисления или побочные эффекты в предложениях **WHERE** и **HAVING**, так как эти предложения тщательно оптимизируются при построении плана выполнения. Логические выражения (сочетания **AND**/**OR**/**NOT**) в этих предложениях могут быть видоизменены любым способом, допустимым законами Булевой алгебры.

Когда порядок вычисления важен, его можно зафиксировать с помощью конструкции **CASE** (см. Раздел 9.18). Например, такой способ избежать деления на ноль в предложении **WHERE** ненадёжен:

```
SELECT ... WHERE x > 0 AND y/x > 1.5;
```

Безопасный вариант:

```
SELECT ... WHERE CASE WHEN x > 0 THEN y/x > 1.5 ELSE FALSE END;
```

Применяемая так конструкция **CASE** защищает выражение от оптимизации, поэтому использовать её нужно только при необходимости. (В данном случае было бы лучше решить проблему, переписав условие как `y > 1.5 * x`.)

https://postgrespro.ru/docs/postgresql/15/sql-expressions#SYNTAX-EXPRESS-EVAL

DISTINCT

```
SELECT DISTINCT список_выборки ...
SELECT DISTINCT ON (выражение [, выражение ...]) список_выборки ...
```

https://postgrespro.ru/docs/postgresql/15/sql-select#SQL-DISTINCT

https://postgrespro.ru/docs/postgresql/15/queries-select-lists#QUERIES-DISTINCT

DISTINCT

```
SELECT DISTINCT ON(str)
  *
FROM (
  VALUES
    ('a', 1), ('b', 2)
  , ('a', 3), ('b', 4) -- неполные клоны
) T(str, i);
```

```
str  | i
text | integer
   a |       1
   b |       2
```

```
SELECT DISTINCT ON(str)
  *
FROM (
  VALUES
    ('a', 1), ('b', 2)
  , ('a', 3), ('b', 4)
) T(str, i)
ORDER BY
  str, i DESC; -- определяем сортировку
```

```
str  | i
text | integer
   a |       3
   b |       4
```

DISTINCT

```sql
SELECT DISTINCT
  *
FROM (
  VALUES
    ('a', 1), ('b', 2)
  , ('a', 1), ('b', 2) -- полные клоны
) T(str, i);
```

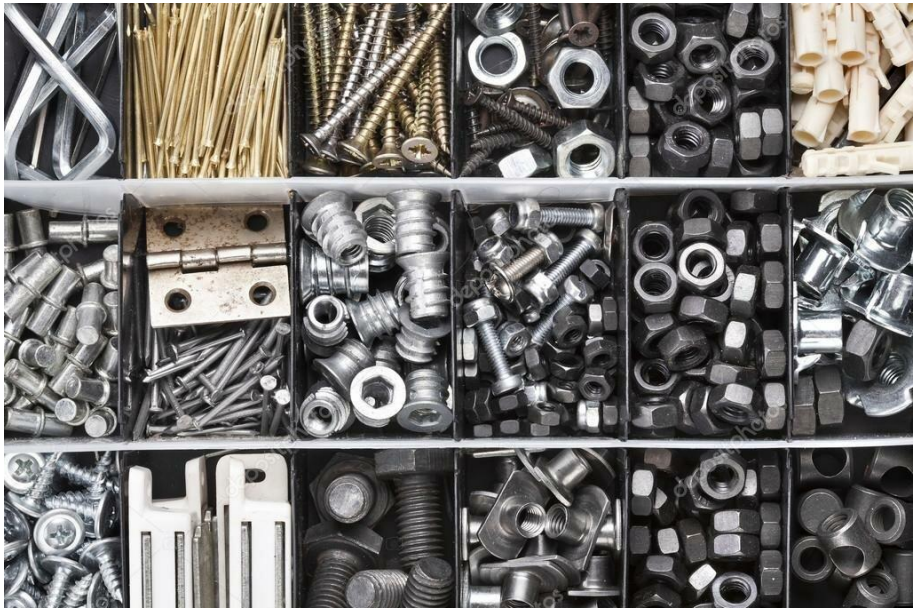
```
str  | i
text | integer
   a |       1
   b |       2
```

GROUP BY

```
GROUP BY элемент_группирования [, ...]
```

https://postgrespro.ru/docs/postgresql/15/sql-select#SQL-GROUPBY

https://postgrespro.ru/docs/postgresql/15/queries-table-expressions#QUERIES-GROUP



count, **min**, **max**, **sum**, **avg**

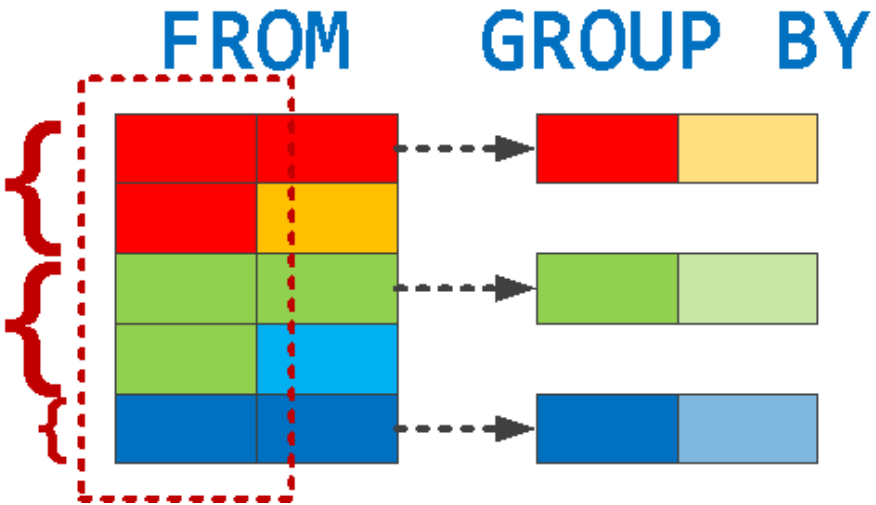
https://postgrespro.ru/docs/postgresql/15/tutorial-agg

https://postgrespro.ru/docs/postgresql/15/functions-aggregate

GROUP BY

```
SELECT
  str
, min(i)
, max(i)
, sum(i)
FROM (
  VALUES
    ('a', 1), ('b', 2)
  , ('a', 3), ('b', 4)
) T(str, i)
GROUP BY
  1; -- str
```

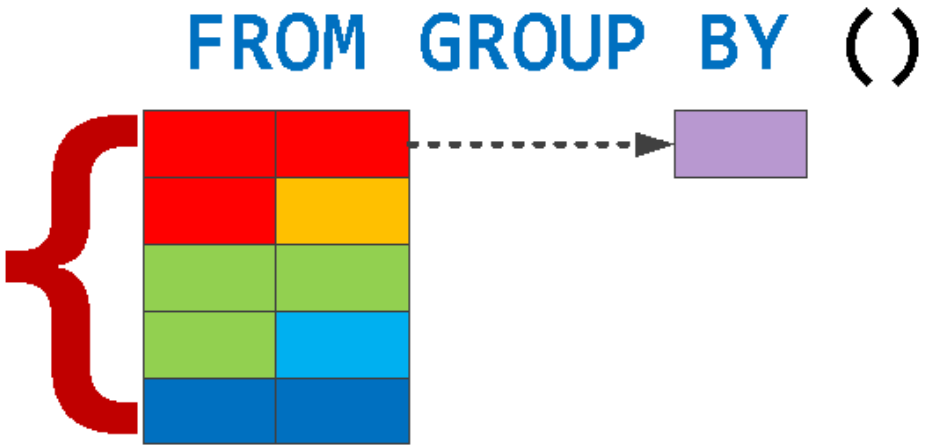
str	min	max	sum
text	integer	integer	integer
a	1	3	4
b	2	4	6



GROUP BY

```sql
SELECT
  min(i)
, max(i)
, sum(i)
FROM (
  VALUES
    ('a', 1), ('b', 2)
  , ('a', 3), ('b', 4)
) T(str, i)
GROUP BY
  (); -- магия!
```

```
min     | max     | sum
integer | integer | integer
      1 |       4 |      10
```



GROUP BY

```
SELECT
  str
, sum(i)
FROM (
  VALUES
    ('a', 1), ('b', 2)
  , ('a', 3), ('b', 4)
) T(str, i)
GROUP BY
  1
ORDER BY
  2 DESC; -- по sum(i)
```

```
str  | sum
text | integer
   b |       6
   a |       4
```

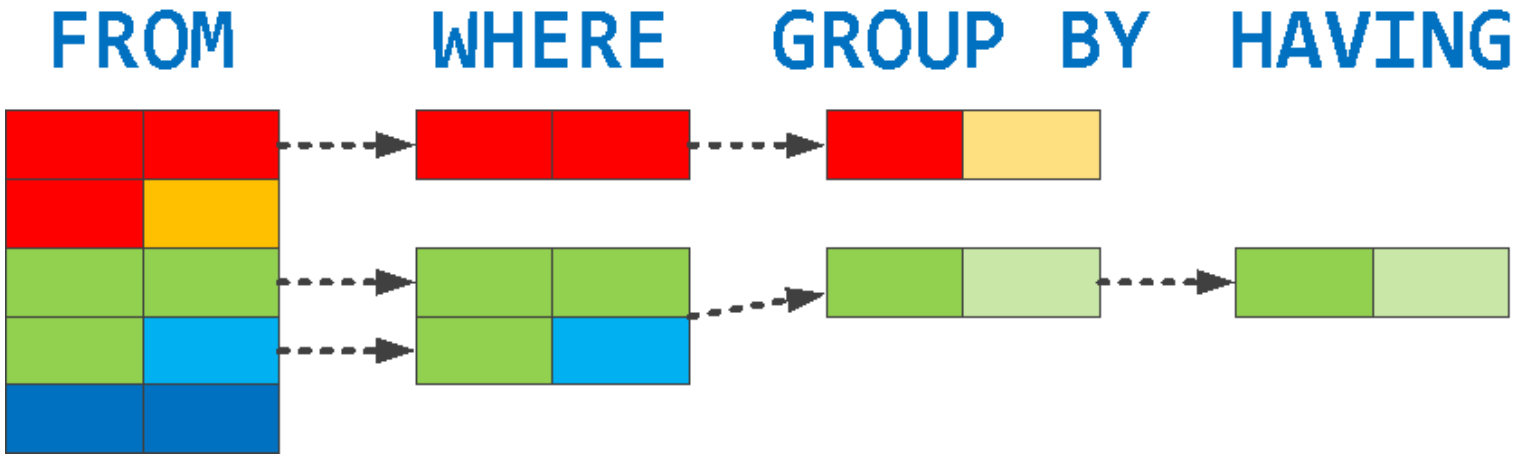
```
SELECT
  str
FROM ...
GROUP BY
  1
ORDER BY
  sum(i) DESC;
```

HAVING

HAVING ***условие***

https://postgrespro.ru/docs/postgresql/15/sql-select#SQL-HAVING

https://postgrespro.ru/docs/postgresql/15/queries-table-expressions#QUERIES-GROUP



HAVING

```
SELECT
  str
, sum(i)
FROM (
  VALUES
    ('a', 1), ('b', 2)
  , ('a', 3), ('b', 4)
) T(str, i)
GROUP BY
  1
HAVING
  sum(i) > 5;
```

```
str  | sum
text | integer
   b |       6
```

```
SELECT
  *
FROM (
  SELECT
    str
  , sum(i)
  FROM ...
  GROUP BY
    1
) T
WHERE
  sum > 5;
```

SELECT – это просто!

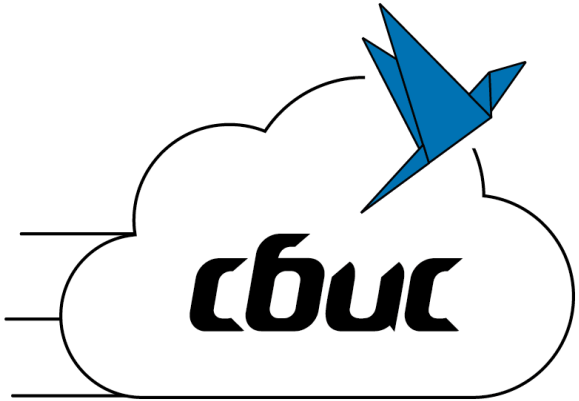
```
SELECT [ DISTINCT [ ON ( выражение [, ...] ) ] ]
    [ * | выражение [ [ AS ] имя_результата ] [, ...] ]
    [ FROM элемент_FROM [, ...] ]
    [ WHERE условие ]
    [ GROUP BY элемент_группирования [, ...] ]
    [ HAVING условие ]
    [ ORDER BY выражение [ ASC | DESC | USING оператор ] [ NULLS { FIRST | LAST } ] [, ...] ]
    [ LIMIT { число | ALL } ]
    [ OFFSET начало [ ROW | ROWS ] ]
    [ FETCH { FIRST | NEXT } [ число ] { ROW | ROWS } { ONLY | WITH TIES } ]
```

https://postgrespro.ru/docs/postgresql/15/sql-select

https://postgrespro.ru/docs/postgresql/15/queries

SELECT – это просто!

```
   FROM
-> WHERE
-> GROUP BY
-> HAVING
-> ORDER BY
-> DISTINCT
-> OFFSET
-> LIMIT
-> FETCH
```



Спасибо за внимание!

Боровиков Кирилл

kilor@tensor.ru / https://n.sbis.ru/explain

sbis.ru / tensor.ru