МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ»

Факультет Управление и информатика в технологических системах
Кафедра Информационная безопасность
Специальность 10.05.03 «Информационная безопасность автоматизированных систем»

ДОПОЛНИТЕЛЬНЫЕ ВОЗМОЖНОСТИ POSTGRESQL

Выполнил студент гр. УБ-01
Зенищева Дарья Леонидовна

Воронеж – 2023

Функция ROW_NUMBER

Функция ROW_NUMBER генерирует порядковый номер строки запроса.

Например:

```
SELECT ROW_NUMBER() OVER (ORDER BY fio) Num, fio
FROM  rykovoditel;
```

```
domcult=# SELECT ROW_NUMBER() OVER (ORDER BY fio) Num, fio
domcult-# FROM  rykovoditel;
 num |             fio
-----+-----------------------------
   1 | Gynkin Sergie Aleksandrovich
   2 | Ivanova Sofia Ivanovna
   3 | Melnikova Ksenia Vitalievna
   4 | Sapsay Ivan Alexeyevich
(4 строки)
```

В данном запросе производится нумерация извлекаемых строк. Также ROW_NUMBER может применяться для ограничения количества обрабатываемых строк. Так, данный запрос извлекает первые пять строк данных:

```
SELECT * FROM (
SELECT ROW_NUMBER() OVER (ORDER BY fio) Num, fio
FROM rykovoditel
) rykovoditel
WHERE Num <= 3;
```

```
domcult=# SELECT * FROM (
domcult(# SELECT ROW_NUMBER() OVER (ORDER BY fio) Num, fio
domcult(# FROM rykovoditel
domcult(# ) rykovoditel
domcult-# WHERE Num <= 3;
 num |             fio
-----+-----------------------------
   1 | Gynkin Sergie Aleksandrovich
   2 | Ivanova Sofia Ivanovna
   3 | Melnikova Ksenia Vitalievna
(3 строки)
```

Функция COALESCE

Функция COALESCE, как правило, применяется чаще всего. Функция принимает несколько параметров:

COALESCE(value [, ...])

Функция возвращает значение первого аргумента, значение которого не равно NULL. Данная функция вернет NULL только в том случае, если все аргументы имеют значение NULL.

SELECT fio, stajwork,

COALESCE(stajwork, 0) staj_work_and_null

FROM rykovoditel;

```
domcult=# SELECT fio, stajwork,
domcult-# COALESCE(stajwork, 0) staj_work_and_null
domcult-# FROM rykovoditel;
            fio              | stajwork | staj_work_and_null
-----------------------------+----------+--------------------
 Melnikova Ksenia Vitalievna |        1 |                  1
 Ivanova Sofia Ivanovna      |        2 |                  2
 Sapsay Ivan Alexeyevich     |        4 |                  4
 Gynkin Sergie Aleksandrovich |       1 |                  1
(4 строки)
```

Числовые функции

Функция ABS

Функция ABS(n) возвращает абсолютное значение числа n.
SELECT ABS(100) X1, ABS(-100) X2, ABS(-100.2) X3;

```
domcult=#  SELECT ABS(100) X1, ABS(-100) X2, ABS(-100.2) X3;
 x1  | x2  |  x3
-----+-----+-------
 100 | 100 | 100.2
(1 строка)
```

Функция CEIL

Функция CEIL(n) возвращает наименьшее целое, большее или равное переданному в качестве параметра числу n.

SELECT CEIL(101) X1, CEIL(-101) X2,

CEIL(101.2) X3, CEIL(-101.2) X4;

```
domcult=# SELECT CEIL(101) X1, CEIL(-101) X2,
domcult-# CEIL(101.2) X3, CEIL(-101.2) X4;
 x1  |  x2  | x3  |  x4
-----+------+-----+------
 101 | -101 | 102 | -101
(1 строка)
```

Функция FLOOR

Функция FLOOR(n) возвращает наибольшее целое, меньшее или равное переданному в качестве параметра числу n.

SELECT FLOOR(110.22) X1, FLOOR(-120.22) X2,

FLOOR(130.99) X3, FLOOR(140.01) X4;

```
domcult=# SELECT FLOOR(110.22) X1, FLOOR(-120.22) X2,
domcult-# FLOOR(130.99) X3, FLOOR(140.01) X4;
 x1  |  x2  | x3  | x4
-----+------+-----+-----
 110 | -121 | 130 | 140
(1 строка)
```

Функция TRUNC

Функция TRUNC(n[, m]) возвращает число n, усеченное до m

знаков после десятичной точки. Параметр m может не указываться – в этом случае n усекается до целого.

SELECT TRUNC(200.35678) X1, TRUNC(-200.35678) X2,

TRUNC(200.79) X3, TRUNC(200.35678, 2) X4;

```
domcult=# SELECT TRUNC(200.35678) X1, TRUNC(-200.35678) X2,
domcult-# TRUNC(200.79) X3, TRUNC(200.35678, 2) X4;
 x1  |  x2  | x3  |   x4
-----+------+-----+--------
 200 | -200 | 200 | 200.35
(1 строка)
```

Функция ROUND

Функция ROUND(n[,m]) возвращает число n, округленное до m знаков после десятичной точки по правилам математического округления. Параметр m может не указываться – в этом случае n округляется до целого.

SELECT ROUND(200.45678) X1, ROUND(200.5) X2,

ROUND(200.79) X3, ROUND(200.65678, 2) X4;

```
domcult=# SELECT ROUND(200.45678) X1, ROUND(200.5) X2,
domcult-# ROUND(200.79) X3, ROUND(200.65678, 2) X4;
 x1  | x2  | x3  |   x4
-----+-----+-----+--------
 200 | 201 | 201 | 200.66
(1 строка)
```

Функция SIGN

Функция SIGN(n) определяет знак числа. Если n положительное, то функция возвращает 1. Если отрицательное – возвращается –1. Если равно нулю, то возвращается 0.

SELECT SIGN(200.22) X1, SIGN(-200.22) X2, SIGN(0) X3;

```
domcult=# SELECT SIGN(200.22) X1, SIGN(-200.22) X2, SIGN(0) X3;
 x1 | x2 | x3
----+----+----
  1 | -1 |  0
(1 строка)
```

Функция MOD

Функция MOD(n, m) возвращает остаток от деления n на m.

SELECT MOD(7, 3) X1, MOD(10, 2) X2, MOD(110, 98) X3;

```
domcult=# SELECT MOD(7, 3) X1, MOD(10, 2) X2, MOD(110, 98) X3;
 x1 | x2 | x3
----+----+----
  1 |  0 | 12
(1 строка)
```

Функция POWER

Функция POWER(n, m) возводит число n в степень m. Степень может быть дробной и отрицательной, что существенно расширяет возможности данной функции.

SELECT POWER(7, 2) X1, POWER(170, 0.5) X2,

POWER(190, 0.33333333) X3, POWER(150, -0.33333333) X4;

```
domcult=# SELECT POWER(7, 2) X1, POWER(170, 0.5) X2,
domcult-# POWER(190, 0.33333333) X3, POWER(150, -0.33333333) X4;
 x1 |         x2         |         x3         |         x4
----+--------------------+--------------------+--------------------
 49 | 13.038404810405297 | 5.7488969783961604 | 0.1882072089196646
(1 строка)
```

Функция SQRT

Функция SQRT(n) возвращает квадратный корень от числа n.

SELECT SQRT(100) X;

```
domcult=# SELECT SQRT(100) X;
 x
----
 10
(1 строка)
```

Функции EXP и LN

Функция EXP(n) возводит e в степень n, а функция LN(n) вычисляет натуральный логарифм от n (при этом значение n должно быть больше нуля).

SELECT EXP(1) X1, LN(3) X2, LN(EXP(2)) X3;

```
domcult=# SELECT EXP(1) X1, LN(3) X2, LN(EXP(2)) X3;
        x1         |         x2         | x3
-------------------+--------------------+----
 2.718281828459045 | 1.0986122886681098 |  2
(1 строка)
```

Попытка передать функции LN отрицательное значение приводит к возникновению ошибки «Вычислить логарифм отрицательного числа нельзя».

Функция LOG

Функция LOG(n, m) производит вычисление логарифма m по основанию n.

SELECT LOG(2, 16) X1, LOG(10, 100) X2;

```
domcult=# SELECT LOG(2, 16) X1, LOG(10, 100) X2;
         x1         |         x2
--------------------+--------------------
 4.0000000000000000 | 2.0000000000000000
(1 строка)
```

Тригонометрические функции

PostgreSQL поддерживает вычисление основных тригонометрических функций:

• SIN(n) – синус n (где n – угол в радианах);

• COS(n) – косинус n (где n – угол в радианах);

• TAN(n) – тангенс n (где n – угол в радианах);

• COT(n) – котангенс n (где n – угол в радианах).

SELECT SIN(0) X1, COS(0) X2, TAN(0) X3, COT(0);

```
domcult=# SELECT SIN(0) X1, COS(0) X2, TAN(0) X3, COT(0);
 x1 | x2 | x3 |   cot
----+----+----+----------
  0 |  1 |  0 | Infinity
(1 строка)
```

Строковые и символьные функции

Функция CONCAT

Функция CONCAT(str1, str2) выполняет конкатенацию строк str1 и str2. Если один из аргументов равен NULL, то он воспринимается как пустая строка. Если оба аргумента равны NULL, то функция возвращает NULL.

SELECT CONCAT('how','are','you') X1,

CONCAT('Test',NULL) X2,

CONCAT(NULL, 'Test') X3,

CONCAT(NULL,NULL)X4;

```
postgres=# SELECT CONCAT( 'how','are','you') X1,
postgres-# CONCAT('Test',NULL) X2,
postgres-# CONCAT(NULL, 'Test') X3,
postgres-# CONCAT(NULL, NULL) X4;
    x1     |  x2  |  x3  | x4
-----------+------+------+----
 howareyou | Test | Test |
(1 строка)
```

Функция LOWER

Функция LOWER(str) преобразует все символы строки str в строчные.

SELECT LOWER(’HoW aRe YoU’) X;

```
postgres=# SELECT LOWER('HoW aRe YoU') X;
      x
-------------
 how are you
(1 строка)
```

Функция UPPER

Функция UPPER(str) преобразует все символы строки str в прописные.

SELECT UPPER(’HoW aRe YoU’) X;

```
postgres=# SELECT UPPER('HoW aRe YoU') X;
      x
-------------
 HOW ARE YOU
(1 строка)
```

Функция INITCAP

Функция INITCAP(str) возвращает строку str, в которой первые буквы всех слов преобразованы в прописные. Функция удобна для форматирования полного имени при построении отчетов.

SELECT INITCAP (‘GynKina AlIna’) X;

```
postgres=# SELECT INITCAP ('GynKina AlIna') X;
       x
---------------
 Gynkina Alina
(1 строка)
```

Функции LTRIM и RTRIM

Функция LTRIM(str [,set]) удаляет все символы с начала строки до первого символа, которого нет в наборе символов set.

По умолчанию set состоит из одного пробела и может не указываться. Функция RTRIM(str [,set]) аналогична LTRIM, но удаляет символы, начиная от конца строки.

SELECT LTRIM(' TeXt DATA') X1,

LTRIM(' _ # TeXt DATA', ' #_') X2,

LTRIM(' 1234567890 TeXt DATA', ' 1234567890') X3;

```
postgres=# SELECT LTRIM(' TeXt DATA') X1,
postgres-# LTRIM(' _ # TeXt DATA', ' #_') X2,
postgres-# LTRIM(' 1234567890 TeXt DATA', ' 1234567890') X3;
    x1     |    x2     |    x3
-----------+-----------+-----------
 TeXt DATA | TeXt DATA | TeXt DATA
(1 строка)
```

SELECT RTRIM('TeXt DATA ') X1,

RTRIM('TeXt DATA _ # ', ' #_') X2,

RTRIM('TeXt DATA 1234567890 ', ' 1234567890') X3;

```
postgres=# SELECT RTRIM('TeXt DATA ') X1,
postgres-# RTRIM('TeXt DATA _ # ', ' #_') X2,
postgres-# RTRIM('TeXt DATA 1234567890 ', ' 1234567890') X3;
    x1     |    x2     |    x3
-----------+-----------+-----------
 TeXt DATA | TeXt DATA | TeXt DATA
(1 строка)
```

Функция TRANSLATE

Функция TRANSLATE(str, from_mask, to_mask) анализирует строку str и заменяет в ней все символы, встречающиеся в строке from_mask, на соответствующие символы из to_mask.

SELECT TRANSLATE ('Test 54321', 'e32', 'E!') X1,

TRANSLATE ('Test 54321', 'e324', ’E') X2;

```
postgres=# SELECT TRANSLATE ('Test 54321', 'e32', 'E!') X1,
postgres-# TRANSLATE ('Test 54321', 'e324', 'E') X2;
    x1     |   x2
-----------+---------
 TEst 54!1 | TEst 51
(1 строка)
```

Функция SUBSTR

Функция SUBSTR(str, m [,n]) возвращает фрагмент строки str, начиная с символа m длиной n символов. Длину можно не указывать – в этом случае возвращается строка от символа m и до конца строки str. Нумерация символов идет с 1. Если указать m равное 0, то копирование все равно начнется с первого символа. Задание отрицательного значения m приводит к тому, что символы отсчитываются от конца строки, а не от начала. Задание значений m, превышающих по абсолютному значению длину строки, приводит к тому, что функция возвращает NULL.

SELECT SUBSTR(’ how are you’, 13) X1,

SUBSTR(’ how are you’, -1) X2,

SUBSTR(’ test text’, 2, 3) X3,

SUBSTR(’ how are you’, 150) X4;

```
postgres=# SELECT SUBSTR(’ how are you’, 13) X1,
postgres-# SUBSTR(’ how are you’, -1) X2,
postgres-# SUBSTR(’ test text’, 2, 3) X3,
postgres-# SUBSTR(’ how are you’, 150) X4;
 x1 |      x2      | x3  | x4
----+--------------+-----+----
    |  how are you | tes |
```

Функция LENGTH

Функция LENGTH(str) возвращает длину строки str в символах.

Для пустой строки функция вернет 0, а для значения NULL – NULL.

SELECT LENGTH(’how are you’) X1,

LENGTH(’’) X2,

LENGTH(NULL) X3;

```
postgres=# SELECT LENGTH(’how are you’) X1,
postgres-# LENGTH(’’) X2,
postgres-# LENGTH(NULL) X3;
 x1 | x2 | x3
----+----+----
 11 |  0 |
```

Функция ASCII

Функция ASCII(str) возвращает ASCII-код первого символа строки str в случае применения кодировок ASCII и UTF-8.

SELECT ASCII(’Tes’) X1;

```
postgres=# SELECT ASCII(’Tes’) X1;
 x1
----
 84
```

Функция CHR

Функция CHR(n) возвращает символ по его коду.

SELECT CHR(84) X1,

CHR(83) X2,

CHR(80) X3,

CHR(81) X4;

```
postgres=# SELECT CHR(84) X1,
postgres-# CHR(83) X2,
postgres-# CHR(80) X3,
postgres-# CHR(81) X4;
 x1 | x2 | x3 | x4
----+----+----+----
 T  | S  | P  | Q
```

Функции работы с датой и временем

Функция NOW

Это одна из самых часто употребляемых функций, она возвращает текущую дату и время по часам сервера.

SELECT NOW();
Функция JUSTIFY_INTERVAL

```
postgres=# SELECT NOW();  Функция JUSTIFY_INTERVAL;
              now
-------------------------------
 2023-05-06 00:38:02.487794+03
```

Функция DATE_TRUNC

Функция DATE_TRUNC(timestamp) используется для обрезки даты или интервала (DATE_TRUNC(interval)) до определенной точности.

SELECT DATE_TRUNC(’HOUR’, NOW()) D1,

DATE_TRUNC(’DAY’, NOW()) D2,

DATE_TRUNC(’MONTH’, NOW()) D3;

```
postgres=# SELECT DATE_TRUNC(’HOUR’, NOW()) D1,
postgres-# DATE_TRUNC(’DAY’, NOW()) D2,
postgres-# DATE_TRUNC(’MONTH’, NOW()) D3;
           d1           |           d2           |           d3
------------------------+------------------------+------------------------
 2023-05-06 00:00:00+03 | 2023-05-06 00:00:00+03 | 2023-05-01 00:00:00+03
```

Для получения дат, соответствующих началу и концу месяца необходимо использовать функции DATE_TRUNC и JUSTIFY_INTERVAL.

SELECT DATE_TRUNC(’MONTH’, NOW()) D1,

DATE_TRUNC(’MONTH’, NOW())

+ JUSTIFY_INTERVAL(’3 MONTH - 1 DAY’) D2;

```
postgres=# SELECT DATE_TRUNC(’MONTH’, NOW()) D1,
postgres-# DATE_TRUNC(’MONTH’, NOW())
postgres-# + JUSTIFY_INTERVAL(’3 MONTH - 1 DAY’) D2;
           d1           |           d2
------------------------+------------------------
 2023-05-01 00:00:00+03 | 2023-07-30 00:00:00+03
```

Данные функции также могут быть использованы для определения количества дней в заданном месяце.

SELECT NOW() D1,

TO_CHAR(DATE_TRUNC(’MONTH’, NOW())

+ JUSTIFY_INTERVAL(’3 MONTH - 1 DAY’), ’DD’) D2;

```
postgres=# SELECT NOW() D1,
postgres-# TO_CHAR(DATE_TRUNC(’MONTH’, NOW())
postgres(# + JUSTIFY_INTERVAL(’3 MONTH - 1 DAY’), ’DD’) D2;
              d1               | d2
-------------------------------+----
 2023-05-06 00:42:59.549016+03 | 30
```

Функция AGE

Функция AGE([end_date,]start_date) возвращает разницу между датами, обозначенными как end_date и start_date. Если параметр end_date опущен, то используется значение глобальной переменной CURRENT_DATE, которая содержит текущую дату (тип date, дата без времени).

SELECT CURRENT_DATE D1,

AGE(MAKE_TIMESTAMP(2022, 8, 25, 4, 15, 20.5)) D2,

AGE(MAKE_DATE(2019, 2, 7),

MAKE_TIMESTAMP(2023, 3, 15, 7, 15, 23.5)) D3;

```
postgres=# SELECT CURRENT_DATE D1,
postgres-# AGE(MAKE_TIMESTAMP(2022, 8, 25, 4, 15, 20.5)) D2,
postgres-# AGE(MAKE_DATE(2019, 2, 7),
postgres(# MAKE_TIMESTAMP(2023, 3, 15, 7, 15, 23.5)) D3;
     d1     |            d2             |                  d3
------------+---------------------------+--------------------------------------
 2023-05-06 | 8 mons 11 days 19:44:39.5 | -4 years -1 mons -8 days -07:15:23.5
```

Функция EXTRACT

Функция EXTRACT(field FROM timestamp) извлекает элемент даты field из значения типа timestamp. Также существует функция EXTRACT(field FROM interval) для работы со значениями типа interval.

SELECT NOW() D1,

EXTRACT(MONTH FROM NOW()) D2,

EXTRACT(YEAR FROM NOW()) D3,

EXTRACT(MINUTE FROM NOW()) D4;

```
postgres=# SELECT NOW() D1,
postgres-# EXTRACT(MONTH FROM NOW()) D2,
postgres-# EXTRACT(YEAR FROM NOW()) D3,
postgres-# EXTRACT(MINUTE FROM NOW()) D4;
              d1               | d2 |  d3  | d4
-------------------------------+----+------+----
 2023-05-06 00:45:17.11234+03 |  5 | 2023 | 45
```

Функция TO_DATE

Функция TO_DATE(str, mask) преобразует строку str в дату. Преобразование ведется по маске mask.

SELECT TO_DATE('12 Dec 2020', 'DD Mon YYY') D1,

TO_DATE('12.12.2020', 'dd.mm.yy') D2;

```
postgres=# SELECT TO_DATE('12 Dec 2020', 'DD Mon YYY') D1,
postgres-# TO_DATE('12.12.2020', 'dd.mm.yy') D2;
     d1     |     d2
------------+------------
 2020-12-12 | 2020-12-12
```

Функция TO_CHAR

Функция TO_CHAR(date, mask) преобразует дату date в символьную строку в соответствии с заданной маской.

Пример:

SELECT NOW() D1,

TO_CHAR(NOW(), ’DD.MM.YY HH24:MI’) D2;