

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

**ФГБОУ ВО «ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ»**

БЕЗОПАСНОСТЬ СЕТЕЙ БАЗ ДАННЫХ

СОЗДАНИЕ БАЗЫ ДАННЫХ В POSTGRESQL

**Семенов Алексей
УБ-02**

**ВОРОНЕЖ
2023**

Функция ROW_NUMBER

```
school=# SELECT ROW_NUMBER() OVER (ORDER BY name_lesson) AS num, name_lesson
school-# FROM lesson;
 num | name_lesson
-----+
  1 | Физика
  2 | ИЗО
  3 | Химия
  4 | Математика
  5 | ПСО
```

функция ROW_NUMBER() используется для нумерации строк таблицы "Lesson" в порядке сортировки по столбцу "name_lesson".

```

school=# SELECT *
school-# FROM (
school(#   SELECT ROW_NUMBER() OVER (ORDER BY name_lesson) AS num, name_lesson
school(#   FROM lesson
school(# ) subquery
school-# WHERE num <= 5;
num | name_lesson
-----+
 1 | Физика
 2 | ИЗО
 3 | Химия
 4 | Математика
 5 | ПСО

```

создаем подзапрос, который вычисляет числовую строку с помощью функции ROW_NUMBER(), а затем внешний запрос выбирает только первые пять строк (где число <= 5).

```

school=# SELECT ROW_NUMBER() OVER (PARTITION BY id_class ORDER BY name_lesson) AS num, name_lesson, id_class
school-# FROM lesson;
num | name_lesson | id_class
-----+-----+-----+
 1 | Химия      | 105
 1 | ИЗО        | 10A
 1 | Физика     | 115
 1 | Математика | 11A
 1 | ПСО        | 9A

```

здесь строки разбиваются на группы по группе "id_class", и каждая группа столбцов нумеруется отдельно внутри группы в порядке сортировки по столбцу "name_lesson".

Функция COALESCE

```

school=# SELECT id_lesson, COALESCE(name_lesson, '') AS name_lesson, COALESCE(fuull_name, '') AS fuull_name,
school-#           COALESCE(id_class, '') AS id_class, start_l_time, end_l_time
school-# FROM Lesson;
id_lesson | name_lesson | fuull_name | id_class | start_l_time | end_l_time
-----+-----+-----+-----+-----+-----+
  1 | Математика | Петров Н.А. | 11A    | 08:00:00  | 08:45:00
  2 | Физика     | Климов Н.А. | 115   | 08:00:00  | 08:45:00
  3 | Химия      | Карасев Д.Д. | 105   | 08:00:00  | 08:45:00
  4 | ИЗО        | Гагарин Д.Д. | 10A    | 08:45:00  | 09:40:00
  5 | ПСО        | Гаврилов В.С. | 9A    | 08:45:00  | 09:40:00

```

функция COALESCE для замены значений NULL в соответствующих столбцах на пустую строку

Числовые функции

```
school=# SELECT ABS(100) X1, ABS(-100) X2, ABS(-100.2) X3;
x1 | x2 | x3
---+---+---
 100 | 100 | 100.2
(1 ё€Ёюр)
```

```
school=# SELECT ABS(100) X1, ABS(-100) X2, ABS(-100.2) X3;
x1 | x2 | x3
---+---+---
 100 | 100 | 100.2
(1 ё€Ёюр)
```

```
school=# SELECT CEIL(100) X1, CEIL(-100) X2,
school-# CEIL(100.2) X3, CEIL(-100.2) X4;
x1 | x2 | x3 | x4
---+---+---+---
 100 | -100 | 101 | -100
(1 ё€Ёюр)
```

```
school=# SELECT FLOOR(100.22) X1, FLOOR(-100.22) X2,
school-# FLOOR(100.99) X3, FLOOR(100.01) X4;
x1 | x2 | x3 | x4
---+---+---+---
 100 | -101 | 100 | 100
(1 ё€Ёюр)
```

```
school=# SELECT TRUNC(100.25678) X1, TRUNC(-100.25678) X2,
school-# TRUNC(100.99) X3, TRUNC(100.25678, 2) X4;
x1 | x2 | x3 | x4
---+---+---+---
 100 | -100 | 100 | 100.25
(1 ё€Ёюр)
```

```
school=# SELECT ROUND(100.25678) X1, ROUND(100.5) X2,
school-# ROUND(100.99) X3, ROUND(100.25678, 2) X4;
x1 | x2 | x3 | x4
---+---+---+---
 100 | 101 | 101 | 100.26
(1 ё€Ёюр)
```

```
school=# SELECT SIGN(100.22) X1, SIGN(-100.22) X2, SIGN(0) X3;
x1 | x2 | x3
---+---+---
 1 | -1 | 0
(1 ё€Ёюр)
```

```
school=# SELECT MOD(10, 3) X1, MOD(10, 2) X2, MOD(100, 98) X3;
x1 | x2 | x3
---+---+---
 1 | 0 | 2
(1 ё€Ёюр)
```

Тригонометрические функции

```
school=# SELECT SIN(0) X1, COS(0) X2, TAN(1) X3, COT(1);
          x1 | x2 |      x3      |      cot
-----+-----+-----+-----+
          0 |  1 | 1.5574077246549023 | 0.6420926159343306
(1 ё€Ёюр)
```

Строковые и символьные функции

```
postgres=# SELECT
postgres-#      REPLACE(' попа была собака', 'собака', 'кошка') AS X1,
postgres-#      REPLACE(' попа была злая собака', 'злая', '') AS X2,
postgres-#      REPLACE(' попа была собака', 'Собака', 'Кошка') AS X3;
          x1 |      x2      |      x3
-----+-----+-----+
    попа была кошка |    попа была собака |    попа была собака
(1 ё€Ёюр)
```

```
postgres=# SELECT LOWER('TeXt DATA') X;
          x
-----
    text data
(1 ё€Ёюр)
```

```
postgres=# SELECT UPPER('TeXt DATA') X;
          x
-----
    TEXT DATA
(1 ё€Ёюр)
```

```

postgres=# SELECT REPLACE('попа была собака', 'собака', 'кошка') X1,
postgres=# REPLACE('попа была злая собака', 'злая', '') X2,
postgres=# REPLACE('попа была собака', 'Собака', 'Кошка') X3;
ШИБКА: тип "6РҮ Є " не существует
ПРОКА 1: SELECT REPLACE(" ІРї Үл< 6РҮ Є ', 6РҮ Є '' , ЕRиE '') X1,
^

postgres=# SELECT TRANSLATE('Test 12345', 'e2', 'E!') X1,
postgres=# TRANSLATE('Test 12345', 'e234', 'E') X2;
   x1    |   x2
-----+-----
TEst 1!345 | TEst 15
1 єЕЇюър)

postgres=# SELECT LTRIM(' TeXt DATA') X1,
postgres=# LTRIM(' _ # TeXt DATA', '#_') X2,
postgres=# LTRIM(' 1234567890 TeXt DATA', ' 1234567890') X3
postgres=# UNION ALL
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
TeXt DATA | TeXt DATA | TeXt DATA
2 єЕЇюъш)

postgres=# CREATE OR REPLACE FUNCTION is_password_correct(
postgres(# password IN char)
postgres=# RETURNS int AS $is_password_correct$
postgres$# BEGIN
postgres$# IF TRANSLATE(password, '0123456789', '**') = password THEN
postgres$# RAISE WARNING
postgres$# Пароль' должен содержать хотя бы одну цифру!';
postgres$# RETURN 0;
postgres$# END IF;
postgres$# RAISE INFO Корректный' пароль!';
postgres$# RETURN 1;
postgres$# END;
postgres$# $is_password_correct$ LANGUAGE plpgsql;
ШИБКА: нераспознанное условие исключения "? аR<m"
ОНТЕКСТ: компиляция функции PL/pgSQL "is_password_correct" в районе строки 4
postgres=# SELECT TRANSLATE('123 455,23', '.', ',') X1,
postgres=# TRANSLATE(' -123 455.23', '.', ',') X2;
   x1    |   x2
-----+-----
123455.23 | -123455.23

```

Функции работы с датой и временем

```
postgres=# SELECT NOW() D1,
postgres-# NOW() + JUSTIFY_INTERVAL('30 DAYS 1 HOUR 2 MINUTE') D2,
postgres-# NOW() - JUSTIFY_INTERVAL('30 DAYS 1 HOUR 2 MINUTE') D3;
      d1           |           d2           |           d3
-----+-----+-----+
2023-05-25 20:43:38.400353+03 | 2023-06-25 21:45:38.400353+03 | 2023-04-25 19:41:38.400353+03
(1 ё€Ёюр)

postgres=# SELECT
postgres-# DATE_TRUNC('HOUR', NOW()) D1,
postgres-# DATE_TRUNC('DAY', NOW()) D2,
postgres-# DATE_TRUNC('MONTH', NOW()) D3;
      d1           |           d2           |           d3
-----+-----+-----+
2023-05-25 20:00:00+03 | 2023-05-25 00:00:00+03 | 2023-05-01 00:00:00+03
(1 ё€Ёюр)

postgres=# SELECT
postgres-# DATE_TRUNC('MONTH', NOW()) D1,
postgres-# DATE_TRUNC('MONTH', NOW())
postgres-# + JUSTIFY_INTERVAL('1 MONTH - 1 DAY') D2;
      d1           |           d2
-----+-----+
2023-05-01 00:00:00+03 | 2023-05-30 00:00:00+03
(1 ё€Ёюр)

postgres=# SELECT
postgres-# CURRENT_DATE D1,
postgres-# AGE(MAKE_TIMESTAMP(2013, 7, 15, 8, 15, 23.5)) D2,
postgres-# AGE(MAKE_DATE(2016, 3, 3)),
postgres-# + MAKE_TIMESTAMP(2013, 7, 15, 8, 15, 23.5) D3;
      d1           |           d2           |           d3
-----+-----+-----+
2023-05-25 | 9 years 10 mons 9 days 15:44:36.5 | 2 years 7 mons 18 days 15:44:36.5
(1 ё€Ёюр)

postgres=# SELECT
postgres-# NOW() D1,
postgres-# EXTRACT(MONTH FROM NOW()) D2,
postgres-# EXTRACT(YEAR FROM NOW()) D3,
postgres-# EXTRACT(MINUTE FROM NOW()) D4;
      d1           |   d2   |   d3   |   d4
-----+-----+-----+-----+
2023-05-25 20:44:12.602209+03 | 5 | 2023 | 44
(1 ё€Ёюр)

postgres=# SELECT NOW() D1,
postgres-# TO_CHAR(NOW(), 'DD.MM.YY HH24:MI') D2;
      d1           |           d2
-----+-----+
2023-05-25 20:44:36.828089+03 | 25.05.23 20:44
(1 ё€Ёюр)

postgres=# SELECT
postgres-# TO_DATE('05 Dec 2000', 'DD Mon YYY') D1,
postgres-# TO_DATE('15.12.2000', 'dd.mm.yy') D2;
      d1           |           d2
-----+-----+
2000-12-05 | 2000-12-15
(1 ё€Ёюр)
```

Вывод: я научился работать в psql и освежил+ обновил свои знания языка sql.