Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт

Кафедра «Информатика»
кафедра

ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ № 2

Базы данных
Тема

Преподаватель _____ Е. П. Моргунов
Подпись, дата Инициалы, Фамилия

Студент КИ19-17/1Б, №031939174 _____ А. К. Никитин
Номер группы, зачетной книжки Подпись, дата Инициалы, Фамилия

Красноярск 2021

1 Ход выполнения

1.1 Задание 1

```
postgres=# CREATE TABLE test_numeric
postgres-# ( measurement numeric(5, 2),
postgres(# description text
postgres(# );
CREATE TABLE
```

Рисунок 1 – Создание таблицы с типом numeric

```
postgres=# INSERT INTO test_numeric
postgres-# VALUES ( 999.9999, 'Какое-то измерение ' );
ОШИБКА:  переполнение поля numeric
ПОДРОБНОСТИ:  Поле с точностью 5, порядком 2 должно округляться до абсолютного значения меньше чем 10^3.
```

Рисунок 2 – Ошибка с numeric

Ошибка возникает по причине того, что новое значение при округление превосходит поле precision типа данных numeric.

1.2 Задание 3

```
postgres=# SELECT 'NaN'::numeric > 123456789;
 ?column?
----------
 t
(1 строка)
```

Рисунок 3 – Проверка сравнения NaN с другим числом

1.3 Задание 5

```
demo=# SELECT count(*) FROM aircrafts;
 count
-------
     9
(1 строка)

demo=# DELETE FROM aircrafts WHERE range < 0;
DELETE 0
demo=# SELECT count(*) FROM aircrafts;
 count
-------
     9
(1 строка)

demo=#
```

Рисунок 4 – Удаление строки по невыполнимому условию

1.1 Задание 5

```
postgres=# SELECT 'Inf'::double precision > 1E+308;
 ?column?
----------
 t
(1 строка)

postgres=# SELECT '-Inf'::double precision < 1E-308;
 ?column?
----------
 t
(1 строка)
```

Рисунок 4 – Сравнение бесконечности и отрицательной бесконечности с наибольшими числами double

1.1 Задание 7

```
postgres=# CREATE TABLE test_serial
postgres-# ( id serial,
postgres(# name text
postgres(# );
CREATE TABLE
postgres=# INSERT INTO test_serial ( name ) VALUES ( 'Вишневая' );
INSERT 0 1
postgres=# INSERT INTO test_serial ( name ) VALUES ( 'Грушевая' );
INSERT 0 1
postgres=# INSERT INTO test_serial ( name ) VALUES ( 'Зеленая' );
INSERT 0 1
postgres=# SELECT * FROM test_serial
postgres-# ;
 id |   name
----+----------
  1 | Вишневая
  2 | Грушевая
  3 | Зеленая
(3 строки)
```

Рисунок 5 – Создание таблицы с полем типа serial

```
postgres=# INSERT INTO test_serial ( id, name ) VALUES ( 10, 'Прохладная' );
INSERT 0 1
postgres=# SELECT * FROM test_serial
;
 id |    name
----+------------
  1 | Вишневая
  2 | Грушевая
  3 | Зеленая
 10 | Прохладная
(4 строки)
```

Рисунок 6 – Добавление строки с заданным вручную полем типа serial

```
postgres=# INSERT INTO test_serial ( name ) VALUES ( 'Луговая' );
INSERT 0 1
postgres=# SELECT * FROM test_serial
;
 id |    name
----+------------
  1 | Вишневая
  2 | Грушевая
  3 | Зеленая
 10 | Прохладная
  4 | Луговая
(5 строк)
```

Рисунок 7 – Добавление строки с незаданным полем типа serial

1.1 Задание 9

В PostgreSQL используется григорианский календарь

1.1 Задание 11

```
postgres=# SELECT current_timestamp;
       current_timestamp
-------------------------------
 2021-02-23 19:11:00.254946+07
(1 строка)

postgres=# SELECT current_timestamp::time( 0 );
 current_timestamp
-------------------
 19:11:03
(1 строка)

postgres=# SELECT current_timestamp::time( 3 );
 current_timestamp
-------------------
 19:11:04.973
(1 строка)
```

Рисунок 8 – Точность вывода значений типа timestamp

```
postgres=# SELECT 'P0001-02-03T04:05:06.789012'::interval;
              interval
-------------------------------------
 1 year 2 mons 3 days 04:05:06.789012
(1 строка)

postgres=# SELECT 'P0001-02-03T04:05:06.789012'::interval( 0 );
          interval
------------------------------
 1 year 2 mons 3 days 04:05:07
(1 строка)

postgres=# SELECT 'P0001-02-03T04:05:06.789012'::interval( 3 );
             interval
---------------------------------
 1 year 2 mons 3 days 04:05:06.789
(1 строка)
```

Рисунок 9 – Точность вывода значений типа interval

```
postgres=# SELECT '2016-09-12'::date;
    date
------------
 2016-09-12
(1 строка)

postgres=# SELECT '2016-09-12'::date( 3 );
ОШИБКА:  у типа "date" не может быть модификаторов
СТРОКА 1: SELECT '2016-09-12'::date( 3 );
                                 ^
```

Рисунок 10 – Невозможность задать точность типу данных date

У типа данных date нет точности, так как оно не может быть нецелым.

1.2 Задание 13

```
postgres=# SHOW datestyle;
 DateStyle
-----------
 ISO, DMY
(1 строка)

postgres=# \q
postgres@brain:~$ PGDATESTYLE="Postgres" psql -d test -U postgres
psql (11.5)
Введите "help", чтобы получить справку.

test=# SHOW datestyle;
   DateStyle
---------------
 Postgres, DMY
(1 строка)
```

Рисунок 11 – Смена стиля даты

1.1 Задание 15

```
postgres=# SELECT to_char( current_timestamp, 'dd.mm.yy' );
 to_char
----------
 23.02.21
(1 строка)
```

Рисунок 12 – Преобразование типа данных timestamp к char по шаблону

1.1 Задание 17

```
                  ^
postgres=# SELECT '21.5:15:16'::time;
ОШИБКА:  неверный синтаксис для типа time: "21.5:15:16"
СТРОКА 1: SELECT '21.5:15:16'::time;
                  ^
```

Рисунок 13 – Ошибка допустимости значения типа данных

1.1 Задание 19

```
postgres=# SELECT ( '20:34:35'::time - '19:44:45'::time );
 ?column?
----------
 00:49:50
(1 строка)
```

Рисунок 14 – Вычитание времени

```
                          ^
postgres=# SELECT ( '20:34:35'::time + '19:44:45'::time );
ОШИБКА:  оператор не уникален: time without time zone + time without time zone
СТРОКА 1: SELECT ( '20:34:35'::time + '19:44:45'::time );
                                    ^
ПОДСКАЗКА:  Не удалось выбрать лучшую кандидатуру оператора. Возможно, вам следует добавить явные приведения типов.
postgres=#
```

Рисунок 15 – Попытка сложения времени

1.1 Задание 21

```
postgres=# SELECT ( '2016-01-31'::date + '1 mon'::interval ) AS new_date;
      new_date
---------------------
 2016-02-29 00:00:00
(1 строка)

postgres=# SELECT ( '2016-02-29'::date + '1 mon'::interval ) AS new_date;
      new_date
---------------------
 2016-03-29 00:00:00
(1 строка)
```

Рисунок 16 – Прибавление месяца в последний день изначального месяца

Таким образом, происходит сложение количества дней изначального месяца с новым месяцем, но не далее конца следующего месяца.

1.1 Задание 23

```
postgres=# SELECT ( '2016-09-16'::date - '2015-09-01'::date );
 ?column?
----------
      381
(1 строка)

postgres=# SELECT ( '2016-09-16'::timestamp - '2015-09-01'::timestamp );
 ?column?
----------
 381 days
(1 строка)
```

Рисунок 17 – Вычитание данных типа date и timestamp

Во втором случае идет уточнение результата “days”, так как тип данных timestamp оперирует и датой, и временем, и уточнение необходимо для ясности восприятия.

1.1 Задание 25

```
postgres=# SELECT ( date_trunc( 'week',timestamp '1999-11-27 12:34:56.987654' ) );
     date_trunc
---------------------
 1999-11-22 00:00:00
(1 строка)

postgres=# SELECT ( date_trunc( 'month',timestamp '1999-11-27 12:34:56.987654' ) );
     date_trunc
---------------------
 1999-11-01 00:00:00
(1 строка)

postgres=# SELECT ( date_trunc( 'year',timestamp '1999-11-27 12:34:56.987654' ) );
     date_trunc
---------------------
 1999-01-01 00:00:00
(1 строка)

postgres=# SELECT ( date_trunc( 'dec',timestamp '1999-11-27 12:34:56.987654' ) );
     date_trunc
---------------------
 1990-01-01 00:00:00
(1 строка)

postgres=# SELECT ( date_trunc( 'cent',timestamp '1999-11-27 12:34:56.987654' ) );
     date_trunc
---------------------
 1901-01-01 00:00:00
(1 строка)

postgres=# SELECT ( date_trunc( 'mil',timestamp '1999-11-27 12:34:56.987654' ) );
     date_trunc
---------------------
 1001-01-01 00:00:00
(1 строка)

postgres=# SELECT ( date_trunc( 'microseconds',timestamp '1999-11-27 12:34:56.987654' ) );
         date_trunc
----------------------------
 1999-11-27 12:34:56.987654
(1 строка)

postgres=# SELECT ( date_trunc( 'millisecond',timestamp '1999-11-27 12:34:56.987654' ) );
       date_trunc
-------------------------
 1999-11-27 12:34:56.987
(1 строка)

postgres=# SELECT ( date_trunc( 'second',timestamp '1999-11-27 12:34:56.987654' ) );
     date_trunc
---------------------
 1999-11-27 12:34:56
(1 строка)

postgres=# SELECT ( date_trunc( 'minute',timestamp '1999-11-27 12:34:56.987654' ) );
     date_trunc
---------------------
 1999-11-27 12:34:00
(1 строка)

postgres=# SELECT ( date_trunc( 'hour',timestamp '1999-11-27 12:34:56.987654' ) );
     date_trunc
---------------------
 1999-11-27 12:00:00
(1 строка)

postgres=# SELECT ( date_trunc( 'day',timestamp '1999-11-27 12:34:56.987654' ) );
     date_trunc
---------------------
 1999-11-27 00:00:00
(1 строка)
```

Рисунок 18 – Использование функции date_trunc

1.1 Задание 27

```
postgres=# SELECT extract('microseconds' from timestamp '1999-11-27 12:34:56.123459');
 date_part
-----------
  56123459
(1 строка)

postgres=# SELECT extract('ms' from timestamp '1999-11-27 12:34:56.123459');
 date_part
-----------
 56123.459
(1 строка)

postgres=# SELECT extract('sec' from timestamp '1999-11-27 12:34:56.123459');
 date_part
-----------
 56.123459
(1 строка)

postgres=# SELECT extract('min' from timestamp '1999-11-27 12:34:56.123459');
 date_part
-----------
        34
(1 строка)

postgres=# SELECT extract('hour' from timestamp '1999-11-27 12:34:56.123459');
 date_part
-----------
        12
(1 строка)

postgres=# SELECT extract('day' from timestamp '1999-11-27 12:34:56.123459');
 date_part
-----------
        27
(1 строка)

postgres=# SELECT extract('week' from timestamp '1999-11-27 12:34:56.123459');
 date_part
-----------
        47
(1 строка)

postgres=# SELECT extract('month' from timestamp '1999-11-27 12:34:56.123459');
 date_part
-----------
        11
(1 строка)

postgres=# SELECT extract('year' from timestamp '1999-11-27 12:34:56.123459');
 date_part
-----------
      1999
(1 строка)

postgres=# SELECT extract('dec' from timestamp '1999-11-27 12:34:56.123459');
 date_part
-----------
       199
(1 строка)

postgres=# SELECT extract('mil' from timestamp '1999-11-27 12:34:56.123459');
 date_part
-----------
         2
(1 строка)
```

Рисунок 19 – Использование функции extract()

1.1 Задание 29

Последняя операция «SELECT * FROM databases WHERE is_open_source <> 1;» не будет выполняться, так как значение 1 не будет интерпретировано логическим типом.

1.2 Задание 31

```
postgres=# SELECT *, ( current_date::timestamp - birthday::timestamp )::interval
postgres-# FROM birthdays;
    person    |  birthday  |  interval
--------------+------------+------------
 Ken Thompson | 1955-03-23 | 24079 days
 Ben Johnson  | 1971-03-19 | 18239 days
 Andy Gibson  | 1987-08-12 | 12249 days
(3 строки)

postgres=# SELECT *, justify_days( current_date::timestamp - birthday::timestamp )::interval
FROM birthdays;
    person    |  birthday  |       justify_days
--------------+------------+-------------------------
 Ken Thompson | 1955-03-23 | 66 years 10 mons 19 days
 Ben Johnson  | 1971-03-19 | 50 years 7 mons 29 days
 Andy Gibson  | 1987-08-12 | 34 years 9 days
(3 строки)
```

Рисунок 20 – Применение функции justify_days()

1.3 Задание 33

```
postgres=# CREATE TABLE pilots
( pilot_name text,
schedule integer[],
meal text[][]
);
CREATE TABLE
```

Рисунок 21 – Создание новой таблицы с многомерным массивом

```
postgres=# INSERT INTO pilots
VALUES ( 'Ivan', '{ 1, 3, 5, 6, 7 }'::integer[],
'{{ "сосиска", "макароны", "кофе" },
postgres'# { "котлета", "каша", "кофе" },
postgres'# { "сосиска", "каша", "кофе" },
postgres'# { "котлета", "каша", "чай" } }'::text[][]
postgres(# );
INSERT 0 1
postgres=# SELECT * FROM pilots
postgres-# ;
 pilot_name |  schedule   |                                          meal
------------+-------------+-------------------------------------------------------------------------------------
 Ivan       | {1,3,5,6,7} | {{сосиска,макароны,кофе},{котлета,каша,кофе},{сосиска,каша,кофе},{котлета,каша,чай}}
(1 строка)
```

Рисунок 22 – Добавление строчки и вывод таблицы

1.4 Задание 35

Ознакомился с некоторыми операторами и функциями json.

```
postgres=# SELECT '{"a":1, "b":2}'::jsonb ? 'c';
 ?column?
----------
 f
(1 строка)

postgres=# SELECT '{"a":1, "b":2, "c":3}'::jsonb ?| array['b', 'd'];
 ?column?
----------
 t
(1 строка)
```

Рисунок 23 – Использование оператора ? и ?| в json-объектах

1.5 Задание 37

```
postgres=# SELECT '{"a":1, "b":2, "c":3}'::jsonb - 'b';
     ?column?
------------------
 {"a": 1, "c": 3}
(1 строка)
```

Рисунок 24 – Удаление из json