

Задание 2:

Создать таблицы, содержащие данные поля и выполнить запрос INSERT и SELECT с условием к полю каждого типа:

- 1) Стандартные типы данных (Числовой, символьный, логический, дата/время)

Скрипт:

```
CREATE TABLE standard_types (
    id SERIAL PRIMARY KEY,
    small_value SMALLINT,
    int_value INTEGER,
    big_value BIGINT,
    real_value REAL,
    double_value DOUBLE PRECISION,
    numeric_value NUMERIC(10,2),
    char_value CHAR(5),
    varchar_value VARCHAR(50),
    text_value TEXT,
    bool_value BOOLEAN,
    date_value DATE,
    time_value TIME,
    timetz_value TIME WITH TIME ZONE,
    timestamp_value TIMESTAMP,
    timestampz_value TIMESTAMP WITH TIME ZONE,
    interval_value INTERVAL);

INSERT INTO standard_types (
    small_value, int_value, big_value, real_value, double_value, numeric_value,
    char_value, varchar_value, text_value,
    bool_value,
    date_value, time_value, timetz_value, timestamp_value, timestampz_value, interval_value
) VALUES (
    5, 100, 999999999,
    3.14, 2.718, 12345.67,
    'abc ', 'hello', 'длинный текст',
    TRUE,
    '2024-05-01', '12:30', '12:30+03',
    '2024-05-01 12:30', '2024-05-01 12:30+03',
    '2 days'
);
SELECT * FROM standard_types
WHERE int_value > 50 AND bool_value = TRUE AND date_value > '2024-01-01';
вывод скрипта:
```

id	small_value	int_value	big_value	real_value	double_value	numeric_value	char_value	varchar_value	text_value	bool_value	date_value	time_value	timetz_value	timestamp_value	timestampz_value	interval_value
1	5	100	999999999	3.14	2.718	12345.67	abc	hello	длинный текст	t	2024-05-01	12:30:00	12:30:00+03	2024-05-01 12:30:00	2024-05-01 09:30:00+03	2 days
2	5	100	999999999	3.14	2.718	12345.67	abc	hello	длинный текст	t	2024-05-01	12:30:00	12:30:00+03	2024-05-01 12:30:00	2024-05-01 09:30:00+03	2 days

2) Перечисления

Скрипт:

```
CREATE TYPE mood AS ENUM ('happy', 'neutral', 'sad');
```

```
CREATE TABLE enum_table (
    id SERIAL PRIMARY KEY,
```

```
    feeling mood  
);  
  
INSERT INTO enum_table (feeling) VALUES ('happy');
```

```
SELECT * FROM enum_table WHERE feeling = 'happy';
```

Вывод скрипта:

```
postgres=# CREATE TYPE mood AS ENUM ('happy', 'neutral', 'sad');  
CREATE TABLE enum_table (  
    id SERIAL PRIMARY KEY,  
    feeling mood  
);  
  
INSERT INTO enum_table (feeling) VALUES ('happy');  
  
SELECT * FROM enum_table WHERE feeling = 'happy';  
CREATE TYPE  
CREATE TABLE  
INSERT 0 1  
 id | feeling  
----+-----  
  1 | happy  
(1 row)
```



3) Массивы

Скрипт:

```
CREATE TABLE array_types (  
    id SERIAL PRIMARY KEY,  
    int_array INTEGER[],  
    text_array TEXT[],  
    bool_array BOOLEAN[]  
);  
INSERT INTO array_types (int_array, text_array, bool_array)  
VALUES ('{1,2,3}', '{"a","b","c"}', '{true,false,true}');  
SELECT * FROM array_types
```

```
WHERE 2 = ANY(int_array);
```

вывод скрипта:

```
postgres=# CREATE TABLE array_types (
    id SERIAL PRIMARY KEY,
    int_array INTEGER[],
    text_array TEXT[],
    bool_array BOOLEAN[]
);
INSERT INTO array_types (int_array, text_array, bool_array)
VALUES ('{1,2,3}', '{"a","b","c"}', '{true,false,true}');
SELECT * FROM array_types
WHERE 2 = ANY(int_array);
CREATE TABLE
INSERT 0 1
 id | int_array | text_array | bool_array
----+-----+-----+-----+
  1 | {1,2,3}   | {a,b,c}   | {t,f,t}
(1 row)
```

4) XML и JSON

Скрипт:

```
CREATE TABLE xml_json_types (
    id SERIAL PRIMARY KEY,
    xml_value XML,
    json_value JSON,
    jsonb_value JSONB
);
INSERT INTO xml_json_types (xml_value, json_value, jsonb_value)
VALUES (
    '<user><name>Ivan</name></user>',
    '{"name":"Ivan","age":20}',
    '{"active":true,"roles":["admin","dev"]}'
);
SELECT * FROM xml_json_types
```

```
WHERE jsonb_value->>'active' = 'true';
```

Вывод скрипта:

```
postgres=# CREATE TABLE xml_json_types (
    id SERIAL PRIMARY KEY,
    xml_value XML,
    json_value JSON,
    jsonb_value JSONB
);
INSERT INTO xml_json_types (xml_value, json_value, jsonb_value)
VALUES (
    '<user><name>Ivan</name></user>',
    '{"name": "Ivan", "age": 20}',
    '{"active": true, "roles": ["admin", "dev"]}'           ↴
);
SELECT * FROM xml_json_types
WHERE jsonb_value->>'active' = 'true';
CREATE TABLE
INSERT 0 1
      id |          xml_value          |       json_value       |          jsonb_value
-----+-----+-----+-----+
      1 | <user><name>Ivan</name></user> | {"name": "Ivan", "age": 20} | {"roles": ["admin", "dev"], "active": true}
(1 row)
```

5) Составные типы

Скрипт:

```
CREATE TYPE address_type AS (
    city TEXT,
    street TEXT,
    house INT
);
```

```
CREATE TYPE person_type AS (
    firstname TEXT,
    lastname TEXT,
    age INT
);
```

```
CREATE TABLE composite_types (
    id SERIAL PRIMARY KEY,
    address address_type,
    person person_type
);
INSERT INTO composite_types (address, person)
VALUES (
    ROW('Москва', 'Тверская', 12),
    ROW('Иван', 'Петров', 25)
);
SELECT * FROM composite_types
WHERE (address).city = 'Москва';
```

Вывод скрипта:

```
postgres=# CREATE TYPE address_type AS (
    city TEXT,
    street TEXT,
    house INT
);

CREATE TYPE person_type AS (
    firstname TEXT,
    lastname TEXT,
    age INT
);

CREATE TABLE composite_types (
    id SERIAL PRIMARY KEY,
    address address_type,
    person person_type
);
INSERT INTO composite_types (address, person)
VALUES (
    ROW('Москва', 'Тверская', 12),
    ROW('Иван', 'Петров', 25)
);
SELECT * FROM composite_types
WHERE (address).city = 'Москва';
CREATE TYPE
CREATE TYPE
CREATE TABLE
INSERT 0 1
 id |      address          |      person
----+-----+-----+
  1 | (Москва,Тверская,12) | (Иван,Петров,25)
(1 row)
```

6) Прочие типы: денежный, двоичный, геометрический, битовые строки, UUID

Скрипт:

```
CREATE EXTENSION IF NOT EXISTS "uuid-ossp";
```

```
CREATE TABLE other_types (
    id SERIAL PRIMARY KEY,
```

```
    money_value MONEY,
    binary_value BYTEA,
```

```
    point_value POINT,
    polygon_value POLYGON,
```

```
    bit_fixed BIT(8),
    bit_var VARBIT(16),
```

```
    uuid_value UUID,
    inet_value INET,
    mac_value MACADDR
```

```
);
```

```
INSERT INTO other_types (
    money_value, binary_value,
    point_value, polygon_value,
    bit_fixed, bit_var,
    uuid_value, inet_value, mac_value
)
```

```
VALUES (
```

```
    '1500.50',
    decode('48656C6C6F','hex'),
```

```
    POINT(3,5),
```

```
    '((0,0),(3,0),(3,3),(0,3))',
```

```
    B'10101010',
```

```
    B'1011',
```

```
    uuid_generate_v4(),
```

```
    '192.168.1.1',
```

```
    '08:00:2b:01:02:03'
```

```
);
```

```
SELECT * FROM other_types WHERE money_value > '1000';
```

Вывод скрипта:

id	money_value	binary_value	point_value	polygon_value	bit_fixed	bit_var	uuid_value	inet_value	mac_value
1	\$1,500.50	\x48656c6c6f	(3,5)	((0,0),(3,0),(3,3),(0,3))	10101010	1011	b7548ff4-8138-489d-a622-aedfd1a3623c	192.168.1.1	08:00:2b:01:02:03