# Практика

1. Получите описание таблицы pg_class.
2. Получите *подробное* описание представления pg_tables.
3. Создайте базу данных и временную таблицу в ней.
   Получите полный список схем в базе, включая системные.
4. Получите список представлений в схеме information_schema.
5. Какие запросы выполняет следующая команда psql?
   ```
   \d+ pg_views
   ```

### 1. Описание pg_class

```
=> \d pg_class
```

```
                    Table "pg_catalog.pg_class"
       Column        |     Type     | Collation | Nullable | Default
---------------------+--------------+-----------+----------+---------
 oid                 | oid          |           | not null |
 relname             | name         |           | not null |
 relnamespace        | oid          |           | not null |
 reltype             | oid          |           | not null |
 reloftype           | oid          |           | not null |
 relowner            | oid          |           | not null |
 relam               | oid          |           | not null |
 relfilenode         | oid          |           | not null |
 reltablespace       | oid          |           | not null |
 relpages            | integer      |           | not null |
 reltuples           | real         |           | not null |
 relallvisible       | integer      |           | not null |
 reltoastrelid       | oid          |           | not null |
 relhasindex         | boolean      |           | not null |
 relisshared         | boolean      |           | not null |
 relpersistence      | "char"       |           | not null |
 relkind             | "char"       |           | not null |
 relnatts            | smallint     |           | not null |
 relchecks           | smallint     |           | not null |
 relhasrules         | boolean      |           | not null |
 relhastriggers      | boolean      |           | not null |
 relhassubclass      | boolean      |           | not null |
 relrowsecurity      | boolean      |           | not null |
 relforcerowsecurity | boolean      |           | not null |
 relispopulated      | boolean      |           | not null |
 relreplident        | "char"       |           | not null |
 relispartition      | boolean      |           | not null |
 relrewrite          | oid          |           | not null |
 relfrozenxid        | xid          |           | not null |
 relminmxid          | xid          |           | not null |
 relacl              | aclitem[]    |           |          |
 reloptions          | text[]       | C         |          |
 relpartbound        | pg_node_tree | C         |          |
Indexes:
    "pg_class_oid_index" PRIMARY KEY, btree (oid)
    "pg_class_relname_nsp_index" UNIQUE CONSTRAINT, btree (relname, relnamespace)
    "pg_class_tblspc_relfilenode_index" btree (reltablespace, relfilenode)
```

### 2. Подробное описание pg_tables

```
=> \d+ pg_tables
```

```
                        View "pg_catalog.pg_tables"
   Column    |  Type   | Collation | Nullable | Default | Storage | Description
-------------+---------+-----------+----------+---------+---------+-------------
 schemaname  | name    |           |          |         | plain   |
 tablename   | name    |           |          |         | plain   |
 tableowner  | name    |           |          |         | plain   |
 tablespace  | name    |           |          |         | plain   |
 hasindexes  | boolean |           |          |         | plain   |
 hasrules    | boolean |           |          |         | plain   |
 hastriggers | boolean |           |          |         | plain   |
 rowsecurity | boolean |           |          |         | plain   |
View definition:
 SELECT n.nspname AS schemaname,
    c.relname AS tablename,
    pg_get_userbyid(c.relowner) AS tableowner,
    t.spcname AS tablespace,
    c.relhasindex AS hasindexes,
    c.relhasrules AS hasrules,
    c.relhastriggers AS hastriggers,
    c.relrowsecurity AS rowsecurity
   FROM pg_class c
     LEFT JOIN pg_namespace n ON n.oid = c.relnamespace
     LEFT JOIN pg_tablespace t ON t.oid = c.reltablespace
  WHERE c.relkind = ANY (ARRAY['r'::"char", 'p'::"char"]);
```

### 3. Полный список схем

```
=> CREATE DATABASE data_catalog;
```

```
CREATE DATABASE
```

```
=> \c data_catalog
```

```
You are now connected to database "data_catalog" as user "student".
```

```
=> CREATE TEMP TABLE t(n integer);
```

```
CREATE TABLE
```

```
=> \dnS
```

```
            List of schemas
        Name        |       Owner
--------------------+-------------------
 information_schema | postgres
 pg_catalog         | postgres
 pg_temp_4          | postgres
 pg_toast           | postgres
 pg_toast_temp_4    | postgres
 public             | pg_database_owner
(6 rows)
```

Временная таблица расположена в схеме pg_temp_N, где N — некоторое число. Такие схемы создаются для каждого сеанса, в котором появляются временные объекты, поэтому их может быть несколько. Имя схемы для временных объектов текущего сеанса можно получить, обратившись к системной функции:

```
=> SELECT pg_my_temp_schema()::regnamespace;
```

```
 pg_my_temp_schema
-------------------
 pg_temp_4
(1 row)
```

Однако в большинстве случаев точное имя схемы знать не нужно, поскольку при необходимости к временному объекту можно обратиться, используя имя схемы pg_temp:

```
=> SELECT * FROM pg_temp.t;
```

```
 n
---
(0 rows)
```

Предназначение некоторых других схем нам уже известно, а с оставшимися (pg_toast*) познакомимся позже.

---

### 4. Список представлений в information_schema

Используем шаблон:

```
=> \dv information_schema.*
```

```
                            List of relations
       Schema       |                  Name                   | Type |  Owner
--------------------+-----------------------------------------+------+----------
 information_schema | _pg_foreign_data_wrappers               | view | postgres
 information_schema | _pg_foreign_servers                     | view | postgres
 information_schema | _pg_foreign_table_columns               | view | postgres
 information_schema | _pg_foreign_tables                      | view | postgres
 information_schema | _pg_user_mappings                       | view | postgres
 information_schema | administrable_role_authorizations       | view | postgres
 information_schema | applicable_roles                        | view | postgres
 information_schema | attributes                              | view | postgres
 information_schema | character_sets                          | view | postgres
 information_schema | check_constraint_routine_usage          | view | postgres
 information_schema | check_constraints                       | view | postgres
 information_schema | collation_character_set_applicability   | view | postgres
 information_schema | collations                              | view | postgres
 information_schema | column_column_usage                     | view | postgres
 information_schema | column_domain_usage                     | view | postgres
 information_schema | column_options                          | view | postgres
 information_schema | column_privileges                       | view | postgres
 information_schema | column_udt_usage                        | view | postgres
 information_schema | columns                                 | view | postgres
 information_schema | constraint_column_usage                 | view | postgres
 information_schema | constraint_table_usage                  | view | postgres
 information_schema | data_type_privileges                    | view | postgres
 information_schema | domain_constraints                      | view | postgres
 information_schema | domain_udt_usage                        | view | postgres
 information_schema | domains                                 | view | postgres
 information_schema | element_types                           | view | postgres
 information_schema | enabled_roles                           | view | postgres
 information_schema | foreign_data_wrapper_options            | view | postgres
 information_schema | foreign_data_wrappers                   | view | postgres
 information_schema | foreign_server_options                  | view | postgres
 information_schema | foreign_servers                         | view | postgres
 information_schema | foreign_table_options                   | view | postgres
 information_schema | foreign_tables                          | view | postgres
 information_schema | information_schema_catalog_name         | view | postgres
 information_schema | key_column_usage                        | view | postgres
 information_schema | parameters                              | view | postgres
 information_schema | referential_constraints                 | view | postgres
 information_schema | role_column_grants                      | view | postgres
 information_schema | role_routine_grants                     | view | postgres
 information_schema | role_table_grants                       | view | postgres
 information_schema | role_udt_grants                         | view | postgres
 information_schema | role_usage_grants                       | view | postgres
 information_schema | routine_column_usage                    | view | postgres
 information_schema | routine_privileges                      | view | postgres
 information_schema | routine_routine_usage                   | view | postgres
 information_schema | routine_sequence_usage                  | view | postgres
 information_schema | routine_table_usage                     | view | postgres
 information_schema | routines                                | view | postgres
 information_schema | schemata                                | view | postgres
 information_schema | sequences                               | view | postgres
 information_schema | table_constraints                       | view | postgres
 information_schema | table_privileges                        | view | postgres
 information_schema | tables                                  | view | postgres
 information_schema | transforms                              | view | postgres
 information_schema | triggered_update_columns                | view | postgres
 information_schema | triggers                                | view | postgres
 information_schema | udt_privileges                          | view | postgres
 information_schema | usage_privileges                        | view | postgres
 information_schema | user_defined_types                      | view | postgres
 information_schema | user_mapping_options                    | view | postgres
 information_schema | user_mappings                           | view | postgres
 information_schema | view_column_usage                       | view | postgres
 information_schema | view_routine_usage                      | view | postgres
 information_schema | view_table_usage                        | view | postgres
 information_schema | views                                   | view | postgres
(65 rows)
```

### 5. Запросы к системному каталогу

Чтобы увидеть запросы, которые выполняют команды psql, включим переменную ECHO_HIDDEN.

```
=> \set ECHO_HIDDEN on
```

```
=> \d+ pg_views
```

```
********* QUERY **********
SELECT c.oid,
  n.nspname,
  c.relname
FROM pg_catalog.pg_class c
     LEFT JOIN pg_catalog.pg_namespace n ON n.oid = c.relnamespace
WHERE c.relname OPERATOR(pg_catalog.~) '^(pg_views)$' COLLATE pg_catalog.default
  AND pg_catalog.pg_table_is_visible(c.oid)
ORDER BY 2, 3;
**************************

******** QUERY **********
SELECT c.relchecks, c.relkind, c.relhasindex, c.relhasrules, c.relhastriggers,
c.relrowsecurity, c.relforcerowsecurity, false AS relhasoids, c.relispartition,
pg_catalog.array_to_string(c.reloptions || array(select 'toast.' || x from
pg_catalog.unnest(tc.reloptions) x), ', ')
, c.reltablespace, CASE WHEN c.reloftype = 0 THEN '' ELSE
c.reloftype::pg_catalog.regtype::pg_catalog.text END, c.relpersistence, c.relreplident,
am.amname
FROM pg_catalog.pg_class c
 LEFT JOIN pg_catalog.pg_class tc ON (c.reltoastrelid = tc.oid)
LEFT JOIN pg_catalog.pg_am am ON (c.relam = am.oid)
WHERE c.oid = '12028';
**************************

********* QUERY **********
SELECT a.attname,
  pg_catalog.format_type(a.atttypid, a.atttypmod),
  (SELECT pg_catalog.pg_get_expr(d.adbin, d.adrelid, true)
   FROM pg_catalog.pg_attrdef d
   WHERE d.adrelid = a.attrelid AND d.adnum = a.attnum AND a.atthasdef),
  a.attnotnull,
  (SELECT c.collname FROM pg_catalog.pg_collation c, pg_catalog.pg_type t
   WHERE c.oid = a.attcollation AND t.oid = a.atttypid AND a.attcollation <>
t.typcollation) AS attcollation,
  a.attidentity,
  a.attgenerated,
  a.attstorage,
  pg_catalog.col_description(a.attrelid, a.attnum)
FROM pg_catalog.pg_attribute a
WHERE a.attrelid = '12028' AND a.attnum > 0 AND NOT a.attisdropped
ORDER BY a.attnum;
**************************

********* QUERY **********
SELECT pg_catalog.pg_get_viewdef('12028'::pg_catalog.oid, true);
**************************

********* QUERY **********
SELECT r.rulename, trim(trailing ';' from pg_catalog.pg_get_ruledef(r.oid, true))
FROM pg_catalog.pg_rewrite r
WHERE r.ev_class = '12028' AND r.rulename != '_RETURN' ORDER BY 1;
**************************

                        View "pg_catalog.pg_views"
   Column   | Type | Collation | Nullable | Default | Storage  | Description
------------+------+-----------+----------+---------+----------+-------------
 schemaname | name |           |          |         | plain    |
 viewname   | name |           |          |         | plain    |
 viewowner  | name |           |          |         | plain    |
 definition | text |           |          |         | extended |
View definition:
 SELECT n.nspname AS schemaname,
    c.relname AS viewname,
    pg_get_userbyid(c.relowner) AS viewowner,
    pg_get_viewdef(c.oid) AS definition
   FROM pg_class c
     LEFT JOIN pg_namespace n ON n.oid = c.relnamespace
  WHERE c.relkind = 'v'::"char";
```

Для формирования вывода потребовалось выполнить пять запросов.

```
=> \set ECHO_HIDDEN off
```