Университет ИТМО
Факультет программной инженерии и компьютерной техники

Распределённые системы хранения данных. Лабораторная работа №1.

Группа: P33131
Студент: Смирнов Виктор Игоревич
Преподаватель: Афанасьев Дмитрий Борисович
Вариант: 776

2024

Ключевые слова

База данных, PostgreSQL, системный каталог.

Содержание

1 Цель работы

Научиться проектировать базы данных, составлять инфологические и даталогические модели данных, реализовывать их в БД PostgreSQL, научиться выполнять запросы.

2 Текст задания

Используя сведения из системных каталогов получить информацию о любой таблице: Номер по порядку, Имя столбца, Атрибуты (в атрибуты столбца включить тип данных, ограничение типа CHECK).

Пример вывода:

```
Таблица: Н_ЛЮДИ
No.  Имя столбца         Атрибуты
--- ------------ ----------------------------------------------------
1   ИД           Type    : NUMBER(9) NOT NULL
                 Comment : 'Уникальный номер человека'
2   ФАМИЛИЯ      Type    : VARCHAR2(25) NOT NULL
                 Comment : 'Фамилия человека'
3   ИМЯ          Type    : VARCHAR2(2000) NOT NULL
                 Comment : 'Имя человека'
4   ОТЧЕСТВО     Type    : VARCHAR2(20)
                 Comment : 'Отчество человека'
5   ДАТА_РОЖДЕНИЯ Type    : DATE NOT NULL
                 Comment : 'Дата рождения человека'
6   ПОЛ          Type    : CHAR(1) NOT NULL
                 Constr  : "AVCON_378561_ПОЛ_000" CHECK (ПОЛ IN ('М', 'Ж'))
                 Constr  : "AVCON_388176_ПОЛ_000" CHECK (ПОЛ IN ('М', 'Ж'))
                 Comment : 'Пол человека'
7   ИНОСТРАН     Type    : VARCHAR2(3) NOT NULL
8   КТО_СОЗДАЛ   Type    : VARCHAR2(40) NOT NULL
9   КОГДА_СОЗДАЛ Type    : DATE NOT NULL
10  КТО_ИЗМЕНИЛ  Type    : VARCHAR2(40) NOT NULL
11  КОГДА_ИЗМЕНИ Type    : DATE NOT NULL
12  ДАТА_СМЕРТИ  Type    : DATE
                 Comment : 'Дата смерти человека'
13  ПИН          Type    : VARCHAR2(20)
14  ИНН          Type    : VARCHAR2(20)
```

3 Создание локального окружения для тестирования

Для локального взаимодействия я решил поднять PostgreSQL в Docker контейнере.

```yaml
version: '3.8'
services:
  database:
    container_name: database
    image: postgres
    restart: always
    ports:
      - 5432:5432
    volumes:
      - .:/workspace
    environment:
      POSTGRES_USER: postgres
      POSTGRES_PASSWORD: postgres
      POSTGRES_DB: postgres
    networks:
      - common
networks:
  common:
    name: common
    driver: bridge
```

Далее был написан SQL скрипт, создающий таблицу, аналогичную той, что в примере.

```sql
drop table person;
create table person (
  id numeric(9, 2) primary key,
  last_name varchar(25) not null,
  first_name varchar(2000) not null,
  patronymic varchar(20),
  birth_date date not null,
  gender char(1) not null,
  foreigner varchar(3) not null,
  created_who varchar(40) not null,
  created_when date not null,
  edited_who varchar(40) not null,
  edited_when date not null,
  death_date date,
  pin varchar(20),
  inn varchar(20),

  check (gender in ('M', 'F')),
  check (gender in ('M', 'F'))
);

comment on column person.id is 'The unique number of the person';
comment on column person.id is 'The unique number of the person';
comment on column person.last_name is 'Last name of the person';
comment on column person.first_name is 'The name of the person';
comment on column person.patronymic is 'The patronymic of the person';
comment on column person.birth_date is 'Date of birth of a person';
comment on column person.death_date is 'Date of death of a person';
```

4 Реализация скрипта

```sql
create or replace function table_column(
  table_schema text,
  table_name text
) returns table (
  number integer,
  name text,
  type text,
  is_nullable boolean,
  comment text,
  constraint_name text,
  check_clause text
) as $$
  select
```

```
    attnum as number,
    information_schema.columns.column_name as name,
    (
      data_type || (
        case
          when character_maximum_length is not null then
            ' (' || cast(character_maximum_length as text) || ')'
          else
            ''
        end
      ) || (
        (
          case
            when (
              numeric_precision is not null or
              numeric_scale is not null
            ) then
              ' ('
            else
              ''
          end
        ) || (
          case
            when numeric_precision is not null then
              cast(numeric_precision as text)
            else
              ''
          end
        ) || (
          case
            when numeric_scale is not null then
              ', ' || cast(numeric_scale as text)
            else
              ''
          end
        ) || (
          case
            when (
              numeric_precision is not null or
              numeric_scale is not null
            ) then
              ')'
            else
              ''
          end
        )
      )
    ) as type,
    (
      information_schema.columns.is_nullable = 'YES'
    ) as is_nullable,
    pg_description.description as comment,
    information_schema.check_constraints.constraint_name as constraint_name,
    information_schema.check_constraints.check_clause as check_clause
  from information_schema.columns
  left join information_schema.constraint_column_usage on (
    constraint_column_usage.table_schema = table_column.table_schema and
    constraint_column_usage.table_name = table_column.table_name and
    constraint_column_usage.column_name = information_schema.columns.column_name
  )
  left join information_schema.check_constraints on (
    check_constraints.constraint_schema = constraint_column_usage.constraint_schema and
    check_constraints.constraint_name = constraint_column_usage.constraint_name
  )
  left join pg_attribute on (
    pg_attribute.attrelid = (
      SELECT oid
      FROM pg_class
      WHERE (
        pg_class.relname = table_column.table_name and
        pg_class.relnamespace = (
          select oid
          from pg_namespace
```

```
          where pg_namespace.nspname = table_column.table_schema
        )
      )
    ) and
    pg_attribute.attname = information_schema.columns.column_name
  )
  left join pg_description on (
    pg_description.objoid = pg_attribute.attrelid and
    pg_description.objsubid = pg_attribute.attnum
  )
  where (
    information_schema.columns.table_schema = table_column.table_schema and
    information_schema.columns.table_name = table_column.table_name
  );
$$ language sql;

create or replace function table_column_pretty(
  table_schema text,
  table_name text
) returns table (
  number integer,
  name text,
  attributes text
) as $$
  select
    number,
    name,
    type || (
      case
        when constr is not null then
          e'\n' || constr
        else
          ''
      end
    ) || (
      case
        when comment is not null then
          e'\n' || comment
        else
          ''
      end
    ) as attributes
  from (
    select
      number,
      name,
      'Type: ' || cast(type as text) || ' ' || (
        case
          when not is_nullable then
            'NOT NULL'
          else
            ''
        end
      ) as type,
      string_agg(
        ('Constr: ' || constraint_name || ' ' || check_clause), e'\n'
      ) as constr,
      ('Comment: ' || comment) as comment
    from table_column(table_schema, table_name)
    group by number, name, type, comment, is_nullable
  ) as qqq
  order by number;
$$ language sql;

create or replace function print_table_info(
  table_schema text,
  table_name text
) returns void as $$
declare
  col record;
  col_contr record;

  C1W integer;
```

```
  C2W integer;
  C31W integer;
  C32W integer;
  REM integer;
begin
  C1W := 2;
  C2W := 12;
  C31W := 8;
  C32W := 64 + 8;
  REM := 11;

  if not exists (
    select *
    from table_column(table_schema, table_name)
    limit 1
  ) then
    raise exception '%', ('Table "' || table_schema || '.' || table_name || '" was not found!');
  end if;

  raise info
    '%',
    rpad(
      '|--- Table "' || table_schema || '.' || table_name || '" Information ',
      C1W + C2W + C31W + C32W + REM,
      '-'
    ) || '|';
  raise info
    '| % | % | % |',
    rpad('N', C1W, ' '),
    rpad('Name', C2W, ' '),
    rpad('Attributes', C31W + C32W + 2, ' ');
  raise info '%', rpad('|', C1W + C2W + C31W + C32W + REM, '-') || '|';
  for col in
    select distinct number, name, type, comment
    from table_column(table_schema, table_name)
    order by number
  loop
    raise info
      '| % | % | % |',
      rpad(cast(col.number as text), C1W, ' '),
      rpad(col.name, C2W, ' '),
      (rpad('Type', C31W, ' ') || ': ' || rpad(col.type, C32W, ' '));
    for col_contr in
      select
        constraint_name as name,
        check_clause as clause
      from table_column(table_schema, table_name)
      where (
        table_column.number = col.number and
        constraint_name is not null
      )
    loop
      raise info
        '| % | % | % |',
        rpad('', C1W, ' '),
        rpad('', C2W, ' '),
        (
          rpad('Constr', C31W, ' ') || ': ' || rpad(
            (col_contr.name || ' ' || col_contr.clause), C32W, ' '
          )
        );
    end loop;
    if col.comment is not null then
      raise info
        '| % | % | % |',
        rpad('', C1W, ' '),
        rpad('', C2W, ' '),
        (rpad('Comment', C31W, ' ') || ': ' || rpad(col.comment, C32W, ' '));
    end if;
    raise info '%', rpad('|', C1W + C2W + C31W + C32W + REM, '-') || '|';
  end loop;
end;
```

```
$$ language plpgsql;

select print_table_info('public', 'person');
```

5 Вывод

```
psql:get_table_info.sql:226: INFO:  |--- Table "public.person" Information ----------------------------------------------------------------|
psql:get_table_info.sql:226: INFO:  | N  | Name          | Attributes                                                                  |
psql:get_table_info.sql:226: INFO:  |----------------------------------------------------------------------------------------------------------|
psql:get_table_info.sql:226: INFO:  | 1  | id            | Type    : numeric (9, 2)                                                    |
psql:get_table_info.sql:226: INFO:  |    |               | Comment : The unique number of the person                                   |
psql:get_table_info.sql:226: INFO:  |----------------------------------------------------------------------------------------------------------|
psql:get_table_info.sql:226: INFO:  | 2  | last_name     | Type    : character varying (25)                                            |
psql:get_table_info.sql:226: INFO:  |    |               | Comment : Last name of the person                                           |
psql:get_table_info.sql:226: INFO:  |----------------------------------------------------------------------------------------------------------|
psql:get_table_info.sql:226: INFO:  | 3  | first_name    | Type    : character varying (2000)                                          |
psql:get_table_info.sql:226: INFO:  |    |               | Comment : The name of the person                                            |
psql:get_table_info.sql:226: INFO:  |----------------------------------------------------------------------------------------------------------|
psql:get_table_info.sql:226: INFO:  | 4  | patronymic    | Type    : character varying (20)                                            |
psql:get_table_info.sql:226: INFO:  |    |               | Comment : The patronymic of the person                                      |
psql:get_table_info.sql:226: INFO:  |----------------------------------------------------------------------------------------------------------|
psql:get_table_info.sql:226: INFO:  | 5  | birth_date    | Type    : date                                                              |
psql:get_table_info.sql:226: INFO:  |    |               | Comment : Date of birth of a person                                         |
psql:get_table_info.sql:226: INFO:  |----------------------------------------------------------------------------------------------------------|
psql:get_table_info.sql:226: INFO:  | 6  | gender        | Type    : character (1)                                                     |
psql:get_table_info.sql:226: INFO:  |    |               | Constr  : person_gender_check ((gender = ANY (ARRAY['M'::bpchar, 'F'::bpchar])))  |
psql:get_table_info.sql:226: INFO:  |    |               | Constr  : person_gender_check1 ((gender = ANY (ARRAY['M'::bpchar, 'F'::bpchar]))) |
psql:get_table_info.sql:226: INFO:  |----------------------------------------------------------------------------------------------------------|
psql:get_table_info.sql:226: INFO:  | 7  | foreigner     | Type    : character varying (3)                                             |
psql:get_table_info.sql:226: INFO:  |----------------------------------------------------------------------------------------------------------|
psql:get_table_info.sql:226: INFO:  | 8  | created_who   | Type    : character varying (40)                                            |
psql:get_table_info.sql:226: INFO:  |----------------------------------------------------------------------------------------------------------|
psql:get_table_info.sql:226: INFO:  | 9  | created_when  | Type    : date                                                              |
psql:get_table_info.sql:226: INFO:  |----------------------------------------------------------------------------------------------------------|
```

```
psql:get_table_info.sql:226: INFO:  | 10 | edited_who   | Type    : character varying
    (40)                                                 |
psql:get_table_info.sql:226: INFO:
    |-----------------------------------------------------------------------------------------------------

psql:get_table_info.sql:226: INFO:  | 11 | edited_when  | Type    : date
                                                          |
psql:get_table_info.sql:226: INFO:
    |-----------------------------------------------------------------------------------------------------

psql:get_table_info.sql:226: INFO:  | 12 | death_date   | Type    : date
                                                          |
psql:get_table_info.sql:226: INFO:  |    |              | Comment : Date of death of a
    person                                               |
psql:get_table_info.sql:226: INFO:
    |-----------------------------------------------------------------------------------------------------

psql:get_table_info.sql:226: INFO:  | 13 | pin          | Type    : character varying
    (20)                                                 |
psql:get_table_info.sql:226: INFO:
    |-----------------------------------------------------------------------------------------------------

psql:get_table_info.sql:226: INFO:  | 14 | inn          | Type    : character varying
    (20)                                                 |
psql:get_table_info.sql:226: INFO:
    |-----------------------------------------------------------------------------------------------------
```

6 Вывод

Данная лабораторная работа помогла мне изучить системный каталог PostgreSQL.

Список литературы