Федеральное государственное автономное образовательное учреждение высшего образования

Университет ИТМО

Дисциплина: Распределённые системы хранения данных

Лабораторная работа 1

Вариант 328

Выполнил:
Кривоносов Егор Дмитриевич

Группа: P33111

Преподаватель:
Николаев Владимир Вячеславович

2022 г.

Санкт-Петербург

Задание

Лабораторные работы

Введите вариант: 328

Используя сведения из представлений словаря данных получить информацию о любой таблице: Номер по порядку, Имя столбца, Атрибуты (в атрибуты столбца включить тип данных, комментарий и индекс).

```
Таблица: Н_ЛЮДИ

 No. Имя столбца Атрибуты
 --- ----------- ------------------------------------------------------
1    ИД          Type    :  NUMBER (9)
                 Commen  :  "Уникальный номер человека"
                 Index   :  "ЧЛВК_PK"

2    ФАМИЛИЯ     Type    :  VARCHAR2 (25)
                 Commen  :  "Фамилия человека"
                 Index   :  "ФАМ_ЛЮД"

3    ИНОСТРАН    Type    :  VARCHAR2 (3)
                 Commen  :  ""
                 Index   :  "ЧЛВК_ИНОСТРАН"

             ...
```

Программу оформить в виде процедуры.

Выполнение

Код функции (основная)

```
CREATE OR REPLACE FUNCTION table_columns_info(t text, schema text)
RETURNS VOID AS
$$
    DECLARE
        new_tab CURSOR FOR (
            SELECT tab.relname, attr.attnum, attr.attname, typ.typname, des.description, idx.indexrelid::regclass as idxname FROM pg_class tab
                JOIN pg_namespace space on tab.relnamespace = space.oid
                JOIN pg_attribute attr on attr.attrelid = tab.oid
                JOIN pg_type typ on attr.atttypid = typ.oid
                LEFT JOIN pg_description des on des.objoid = tab.oid and des.objsubid = attr.attnum
                LEFT JOIN pg_index idx on tab.oid = idx.indrelid and attr.attnum = any(idx.indkey)
            WHERE tab.relname = t and attnum > 0 and space.nspname = schema
            ORDER BY attnum
            );
        table_count int;
    BEGIN
        SELECT COUNT(DISTINCT nspname) INTO table_count FROM pg_class tab JOIN pg_namespace space on tab.relnamespace = space.oid WHERE relname = t and space.nspname = schema;

        IF table_count < 1 THEN
            RAISE EXCEPTION 'Таблица "%" не найдена в схеме "%"!', t, schema;
        ELSE
        RAISE NOTICE ' ';
            RAISE NOTICE 'Таблица: %', t;
            RAISE NOTICE ' ';
            RAISE NOTICE 'No.  Имя столбца      Атрибуты';
            RAISE NOTICE '---  --------------   ------------------------------------------------';

            FOR col in new_tab
            LOOP
                RAISE NOTICE '% % Type    :  %', RPAD(col.attnum::text, 5, ' '), RPAD(col.attname, 16, ' '), col.typname;
                RAISE NOTICE '% Commen  :  "%"', RPAD(' ', 22, ' '), CASE WHEN col.description is null THEN '' ELSE col.description END;
                RAISE NOTICE '% Index   :  "%"', RPAD(' ', 22, ' '), CASE WHEN col.idxname is null THEN '' ELSE col.idxname::text END;
            RAISE NOTICE ' ';
            END LOOP;
    END IF;
    END
$$ LANGUAGE plpgsql;
```

Код функции (вспомогательная)

```
CREATE OR REPLACE FUNCTION schemas_table(t text)
RETURNS VOID AS
$$
    DECLARE
        schema_tab CURSOR FOR (
            SELECT tab.relname, space.nspname FROM pg_class tab
                JOIN pg_namespace space on tab.relnamespace = space.oid
            WHERE tab.relname = t
            ORDER BY space.nspname
            );
        table_count int;
        schema text;
    BEGIN
        SELECT COUNT(DISTINCT nspname) INTO table_count FROM pg_class tab JOIN pg_namespace space on tab.relnamespace = space.oid WHERE relname = t;

        IF table_count < 1 THEN
            RAISE EXCEPTION 'Таблица "%" не найдена!', t;
        ELSE
        RAISE NOTICE ' ';
            RAISE NOTICE 'Выберите схему, с которой вы хотите получить данные: ';

            FOR col in schema_tab
            LOOP
                RAISE NOTICE '%', col.nspname;
            END LOOP;
            RAISE NOTICE ' ';
        END IF;
    END
$$ LANGUAGE plpgsql;
```

Код скрипта SQL

```
\echo 'Введите название таблицы: '
\prompt 'Введите название таблицы: ' name_table

\set name_tab '\'' :name_table '\''

SELECT schemas_table(:name_tab::text);

\echo 'Введите название схемы: '
\prompt 'Введите название схемы: ' schema_name_cur

\set name_shem '\'' :schema_name_cur '\''

SELECT table_columns_info(:name_tab::text, :name_shem::text);
```

Код скрипта Bash (Чтобы удалить NOTICE:)

```
#!/bin/bash

psql -h pg -d studs -f ~/Download/RSHD_LAB1/script.sql 2>&1 | sed 's|.*NOTICE:  ||g'
```

Проверка работаспособности функции

```
redgry@helios:~$ bash ~/Download/RSHD_LAB1/script.sh
Введите название таблицы:
world

Выберите схему, с которой вы хотите получить данные:
s261747
s265077
s265936
s283904
s285597
s285851

 schemas_table
---------------

(1 row)

Введите название схемы:
s285597

Таблица: world

No.  Имя столбца      Атрибуты
---  --------------   -------------------------------------------------
1     id_world         Type    :  int4
                       Commen  :  ""
                       Index   :  "s285597.world_pkey"

2     freedom          Type    :  bool
                       Commen  :  ""
                       Index   :  ""

3     danger           Type    :  int4
                       Commen  :  ""
                       Index   :  ""

4     id_shell         Type    :  int4
                       Commen  :  ""
                       Index   :  ""

 table_columns_info
--------------------

(1 row)

redgry@helios:~$
```

Вывод

В ходе выполнения работа с помощью системных каталогов PostgreSQL можно исследовать структуру базы данных: узнать какие существуют схемы, таблицы, стоблцы и т.п. Системные каталоги представляют с собой таблицы, поэтому с ними удобно работать и получать нужную информацию. Также как и обычные таблицы их можно модифицировать (добавить столбцы, изменить или добавить строки).
Также мною была изучена новая вещь как Курсор, которая позволяет получать не сразу полность запрос, а определенное количество строк. Это требуется например, чтобы не заполнять всю память большими запросами.

Процедуры и функции инкапсулируют логику работы с БД в удобной форме. Функции могут использоваться сторонними приложениями, выполняющими операции с БД. Самое интересное было узнать о мета-командах. Они позволяют сделать интерактивную логику (особенно на более новой версии postgresql).