Университет ИТМО

Факультет ФПИ и КТ P33131

Отчет по лабраторной работе №1

«Распределенные системы хранения данных»

Вариант 816

Студент:

Чжоу Хунсян

Гр.P33131

Преподаватель:

Оглавление

Задание

Используя сведения из системных каталогов получить информацию о первичных и внешних ключах схемы: Номер по порядку, Имя ограничения целостности, Тип, Имя столбца, Имя таблицы. Кроме того, для внешних ключей указать Имя таблицы и Имя столбца на которые ссылаются эти ключи. Тип ограничения: R - внешний ключ, P - первичный ключ,

Имя ограничения	Тип	Имя столбца	Имя таблицы	Имя таблицы	Имя столбц
ПЛАН_PK	P	ИД	Н_ПЛАНЫ		
ПЛАН_КАФ_FK	R	ОТД_ИД_ЗАКРЕПЛЕН_ЗА	Н_ПЛАНЫ	Н_ОТДЕЛЫ	ИД
ПЛАН_НАПС_FK	R	НАПС_ИД	Н_ПЛАНЫ	Н_НАПРАВЛЕН	ИД
ПЛАН_ПЛАН_FK	R	ПЛАН_ИД	Н_ПЛАНЫ	Н_ПЛАНЫ	ИД
ПЛАН_ПЛАН_ОСНОВ_НА_FK	R	ПЛАН_ИД_ОСНОВ_НА	Н_ПЛАНЫ	Н_ПЛАНЫ	ИД
ПЛАН_ТЛП_FK	R	ТПЛ_ИД	Н_ПЛАНЫ	Н_ТИПЫ_ПЛАН	ИД
ПЛАН_ФАК_FK	R	ОТД_ИД	Н_ПЛАНЫ	Н_ОТДЕЛЫ	ИД
ПЛАН_ФО_FK	R	ФО_ИД	Н_ПЛАНЫ	Н_ФОРМЫ_ОБУ	ИД
УЧЕН_PK	P	ИД	Н_УЧЕНИКИ		
УЧЕН_ОБУЧ_FK	R	ВИД_ОБУЧ_ИД	Н_УЧЕНИКИ	Н_ОБУЧЕНИЯ	ИД_ОБУЧ_ИД
УЧЕН_ОБУЧ_FK	R	ЧЛВК_ИД	Н_УЧЕНИКИ	Н_ОБУЧЕНИЯ	ЧЛВК_ИД
УЧЕН_ПЛАН_FK	R	ПЛАН_ИД	Н_УЧЕНИКИ	Н_ПЛАНЫ	ИД
УЧЕН_ПЛАН_ГРУППА_FK	R	ГРУППА	Н_УЧЕНИКИ	Н_ГРУППЫ_ПЛ	ГРУППА
УЧЕН_ПЛАН_ГРУППА_FK	R	ПЛАН_ИД	Н_УЧЕНИКИ	Н_ГРУППЫ_ПЛ	
ПЛАН_ИД		...			

Решение

```
CREATE OR REPLACE PROCEDURE get_pf_constraint_info(schema TEXT)
    LANGUAGE plpgsql
AS
$$
DECLARE
    constraint_record RECORD;
BEGIN
    RAISE INFO '% % % % % %',
        format('%-40s', 'Имя ограничения'),
        format('%-3s', 'Тип'),
        format('%-20s', 'Имя таблицы'),
        format('%-30s', 'Имя столбцов'),
        format('%-20s', 'Имя внешней таблицы'),
        format('%-30s', 'Имя внешних столбцов');
    RAISE INFO '% % % % % %',
        repeat('-', 40),
        repeat('-', 3),
        repeat('-', 20),
        repeat('-', 30),
        repeat('-', 20),
        repeat('-', 30);
    FOR constraint_record IN
        SELECT
            conname AS constraint_name,
            contype AS constraint_type,
            conrelid::regclass::text AS table_name,
            array_agg(DISTINCT a.attname) AS column_names,
            f.relname AS foreign_table,
            array_agg(DISTINCT af.attname) AS foreign_column_names
```

```
        FROM
            pg_constraint c
                JOIN pg_namespace namespace ON c.connamespace = namespace.oid
                JOIN pg_class t ON c.conrelid = t.oid
                JOIN pg_attribute a ON a.attrelid = t.oid AND a.attnum = ANY(c.conkey)
                LEFT JOIN pg_class f ON c.confrelid = f.oid
                LEFT JOIN pg_attribute af ON af.attrelid = f.oid AND af.attnum = ANY(c.confkey)
        WHERE namespace.nspname = schema AND contype IN ('p', 'f')
        GROUP BY conname, contype, table_name, foreign_table
        LOOP
            IF constraint_record.constraint_type = 'p' THEN
                RAISE INFO '% % % %',
                    format('%-40s', constraint_record.constraint_name),
                    format('%-3s', 'P'),
                    format('%-20s', constraint_record.table_name),
                    format('%-30s', array_to_string(constraint_record.column_names, ', '));
            ELSIF constraint_record.constraint_type = 'f' THEN
                RAISE INFO '% % % % % %',
                    format('%-40s', constraint_record.constraint_name),
                    format('%-3s', 'R'),
                    format('%-20s', constraint_record.table_name),
                    format('%-30s', array_to_string(constraint_record.column_names, ', ')),
                    format('%-20s', constraint_record.foreign_table),
                    format('%-30s', array_to_string(constraint_record.foreign_column_names, ', '));
            END IF;
        END LOOP;
END
$$;

call get_pf_constraint_info('s336184');
```

Структура таблицы

```
-- Create TYPE ENUM
CREATE TYPE PROBLEM_TYPE AS ENUM ('UI', 'BUGS', 'SCRIPT');
CREATE TYPE HOUSE_TYPE AS ENUM ('APARTMENTS', 'VILLAS', 'HIGH-END','ORDINARY');
CREATE TYPE DEVICE_TYPE AS ENUM ('AIR_CONDITION','LIGHT', 'HUMIDIFIER', 'BATHTUB', 'OUTLET','CURTAINS', 'FAN',
'CAMERA', 'WATER_HEATER');
CREATE TYPE SENSOR_TYPE AS ENUM ('TEMPERATURE', 'HUMIDITY', 'SMOKE');
CREATE TYPE ACTION_TYPE AS ENUM ('CLOSE','OPEN','SWITCH_OFF','SWITCH_ON','ADJUST_VALUE','TURN_ON','TURN_OFF');
CREATE TYPE SCRIPT_TYPE AS ENUM ('CONDITIONAL','SCHEDULE');
CREATE TYPE COUNTRY AS ENUM('US','UK','RUSSIAN','CHINA','FRANCE');
CREATE TYPE CITY AS ENUM('Shanghai', 'Beijing', 'Shenzhen', 'Guangzhou', 'Chengdu','Paris', 'Marseille', 'Lyon',
'Toulouse','Cambridge', 'Edinburgh',  'London', 'Liverpool','New York', 'Los Angeles', 'Chicago', 'Boston');
CREATE TYPE ROOM_TYPE AS ENUM('KITCHEN','BEDROOM','BATHROOM','LIVING');
CREATE TYPE GENDER AS ENUM ('MALE','FEMALE');

-- Create Table
CREATE TABLE IF NOT EXISTS
family(     id SERIAL PRIMARY KEY
NOT NULL,     name VARCHAR(64) NOT
NULL,     info TEXT NOT NULL
);

CREATE TABLE IF NOT EXISTS
"user"(     id SERIAL PRIMARY KEY
NOT NULL,     password VARCHAR(256)
NOT NULL,     gender GENDER NOT
NULL,     username VARCHAR(64) NOT
NULL,     age INT NOT NULL CHECK
(age > 0)
);

CREATE TABLE IF NOT EXISTS
supporter(     id SERIAL PRIMARY KEY NOT
NULL,     password VARCHAR(256) NOT NULL,
username VARCHAR(64) NOT NULL,
is_free BOOLEAN NOT NULL DEFAULT TRUE
);

CREATE TABLE IF NOT EXISTS issue(     id SERIAL
PRIMARY KEY NOT NULL,     user_id INT NOT NULL
REFERENCES "user"(id),     supporter_id INT NOT
NULL REFERENCES supporter(id),     is_finished
BOOLEAN NOT NULL DEFAULT FALSE,     description
TEXT NOT NULL,     issue_type PROBLEM_TYPE NOT
NULL,     create_time DATE DEFAULT CURRENT_DATE
);

CREATE TABLE IF NOT EXISTS
address(     id SERIAL NOT NULL
PRIMARY KEY,     country COUNTRY
```

```
NOT NULL,     city CITY NOT NULL,
street VARCHAR(128) NOT NULL
);

CREATE TABLE IF NOT EXISTS house(     id SERIAL
PRIMARY KEY NOT NULL,     address_id INT REFERENCES
address(id) NOT NULL,     house_type HOUSE_TYPE NOT
NULL
);

CREATE TABLE IF NOT EXISTS room(     id SERIAL
PRIMARY KEY NOT NULL,     house_id INT NOT NULL
REFERENCES house(id),     area_size FLOAT NOT
NULL,     height FLOAT NOT NULL,     room_type
ROOM_TYPE NOT NULL,     is_filled BOOLEAN NOT
NULL DEFAULT FALSE
);

CREATE TABLE IF NOT EXISTS device(     id
SERIAL PRIMARY KEY NOT NULL,     room_id INT
REFERENCES room(id) NOT NULL,     manufacture
VARCHAR(256) NOT NULL,     available BOOLEAN
NOT NULL DEFAULT TRUE,     device_type
DEVICE_TYPE NOT NULL
);

CREATE TABLE IF NOT EXISTS sensor(     id
SERIAL PRIMARY KEY NOT NULL,     room_id INT
REFERENCES room(id) NOT NULL,     manufacture
VARCHAR(256) NOT NULL,     available BOOLEAN
NOT NULL DEFAULT TRUE,     sensor_type
SENSOR_TYPE NOT NULL
);

CREATE TABLE IF NOT EXISTS
device_action(     id SERIAL PRIMARY KEY NOT
NULL,     device_type DEVICE_TYPE NOT NULL,
action_type ACTION_TYPE NOT NULL,
description TEXT
);

CREATE TABLE IF NOT EXISTS script(     id
SERIAL PRIMARY KEY NOT NULL,     creator INT
REFERENCES "user"(id) NOT NULL,     script_type
SCRIPT_TYPE NOT NULL
);

CREATE TABLE IF NOT EXISTS schedule_script(     id
SERIAL PRIMARY KEY NOT NULL,     script_id INT
REFERENCES script(id) NOT NULL,     action_time TIME
NOT NULL,     repeat_on_monday BOOLEAN NOT NULL DEFAULT
FALSE,     repeat_on_tuesday BOOLEAN NOT NULL DEFAULT
FALSE,     repeat_on_wednesday BOOLEAN NOT NULL DEFAULT
FALSE,     repeat_on_thursday BOOLEAN NOT NULL DEFAULT
FALSE,     repeat_on_friday BOOLEAN NOT NULL DEFAULT
FALSE,     repeat_on_saturday BOOLEAN NOT NULL DEFAULT
FALSE,     repeat_on_sunday BOOLEAN NOT NULL DEFAULT
FALSE
);

CREATE TABLE IF NOT EXISTS
condition_script(     id SERIAL PRIMARY KEY NOT
NULL,     script_id INT REFERENCES script(id) NOT
NULL,     condition_text TEXT NOT NULL
);

CREATE TABLE IF NOT EXISTS contact(     user_id INT
REFERENCES "user"(id) NOT NULL UNIQUE,     email
VARCHAR(128) NOT NULL UNIQUE,     phone_num
VARCHAR(64) NOT NULL UNIQUE
);

CREATE TABLE IF NOT EXISTS
list_action_script(     id SERIAL NOT NULL
PRIMARY KEY,     script_id INT REFERENCES
script(id),     action_id INT REFERENCES
device_action(id) );

CREATE TABLE IF NOT EXISTS
list_script_user(     id SERIAL NOT NULL
PRIMARY KEY,     script_id INT REFERENCES
script(id),     user_id INT REFERENCES
"user"(id),
    UNIQUE (script_id, user_id)
);

CREATE TABLE IF NOT EXISTS
list_user_house(     id SERIAL NOT NULL
PRIMARY KEY,     user_id INT REFERENCES
```

```
"user"(id),     house_id INT REFERENCES
house(id),
    UNIQUE (user_id, house_id)
);

CREATE TABLE IF NOT EXISTS
list_user_family(     id SERIAL NOT NULL
PRIMARY KEY,     user_id INT REFERENCES
"user"(id),     family_id INT REFERENCES
house(id),
    UNIQUE (user_id, family_id) );
```

Скрипт:

```
psql -h pg -d studs -c call
get_pf_constraint_info('s336184');
```

Пример

```
studs.s336184> CALL get_pf_constraint_info('s336184') Имя ограничения                          Тип Имя столбца

Имя таблицы          Имя таблицы          Имя столбца ---------------------------------------- --- ---------------

-------------------- --------------------

--------------- address_pkey                              P   id              address condition_script_pkey

P   id           condition_script condition_script_script_id_fkey       R   script_id     condition_script

script              id contact_user_id_fkey                   R   user_id        contact             user

id device_action_pkey                                 P    id                   device_action  device_pkey

P    id                 device device_room_id_fkey                         R    room_id           device

room                     id family_pkey                                    P    id                 family

house_address_id_fkey                    R   address_id      house               address              id

house_pkey                               P   id             house issue_pkey                              P

id            issue issue_supporter_id_fkey               R   supporter_id    issue              supporter

id issue_user_id_fkey                       R   user_id        issue               user                id

list_action_script_action_id_fkey        R   action_id       list_action_script   device_action        id

list_action_script_pkey              P   id           list_action_script list_action_script_script_id_fkey

R   script_id       list_action_script   script              id list_script_user_pkey                    P

id            list_script_user list_script_user_script_id_fkey         R   script_id      list_script_user

script                 id list_script_user_user_id_fkey             R   user_id          list_script_user

user                    id list_user_family_family_id_fkey           R   family_id        list_user_family

house                  id list_user_family_pkey                     P   id               list_user_family

list_user_family_user_id_fkey            R   user_id         list_user_family     user                id

list_user_house_house_id_fkey            R   house_id        list_user_house      house               id

list_user_house_pkey                     P   id             list_user_house list_user_house_user_id_fkey

R   user_id         list_user_house      user                id room_house_id_fkey                       R

house_id        room                 house                id room_pkey                                 P

id                room schedule_script_pkey                         P   id               schedule_script

schedule_script_script_id_fkey           R   script_id       schedule_script      script              id

script_creator_fkey                      R   creator         script               user                id

script_pkey                              P   id             script sensor_pkey                           P

id             sensor sensor_room_id_fkey                    R   room_id        sensor              room

id supporter_pkey                           P   id             supporter
```

Вывод

В ходе выполнения работы изучал как использовать системный каталог чтобы читать информацию таблиц в БД.