Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

Базы данных

Отчет по лабораторной работе №2
Язык SQL-DDL

Работу выполнил:
Графов Д.И.
Группа: 33531/2
Преподаватель:
Мяснов А.В.

Санкт-Петербург
2019

Содержание

1. Цель работы

Познакомиться с основами проектирования схемы БД, языком описания сущностей и ограничений БД SQL-DDL.

2. Программа работы

1. Самостоятельное изучение SQL-DDL.
2. Создание скрипта БД в соответствии с согласованной схемой. Должны присутствовать первичные и внешние ключи, ограничения на диапазоны значений. Демонстрация скрипта преподавателю.
3. Создание скрипта, заполняющего все таблицы БД данными.
4. Выполнение SQL-запросов, изменяющих схему созданной БД по заданию преподавателя. Демонстрация их работы преподавателю.

3. Теоретическая информация

3.1. Язык SQL

Язык SQL (Structured Query Language) - язык структурированных запросов. Он позволяет формировать весьма сложные запросы к базам данных. В SQL определены два подмножества языка:

- SQL-DDL (Data Definition Language) - язык определения структур и ограничений целостности баз данных. Сюда относятся команды создания и удаления баз данных; создания, изменения и удаления таблиц; управления пользователями и т.д.
- SQL-DML (Data Manipulation Language) - язык манипулирования данными: добавление, изменение, удаление и извлечение данных, управления транзакциями

3.2. Операции ON DELETE и ON UPDATE

Выражения ON DELETE и ON UPDATE внешних ключей используются для указания действий, которые будут выполняться при удалении строк родительской таблицы (ON DELETE) или изменении родительского ключа (ON UPDATE). Один и тот же внешний ключ может иметь различные действия, указанные для ON DELETE и ON UPDATE.

В базе данных PostgreSQL действия ON DELETE и ON UPDATE, ассоциированные с внешним ключом, могут быть следующими: NO ACTION, RESTRICT, SET NULL, SET DEFAULT или CASCADE. Если действие не указывается специально, оно по умолчанию является NO ACTION. Рассмотрим действия, используемые в данной лабораторной работе.

- **NO ACTION**: опция «NO ACTION» означает, что когда родительский ключ изменяется или удаляется из базы данных, никаких специальных действий не производится.

- **CASCADE**: действие «CASCADE» распространяет операции удаления и изменения родительского ключа на зависящие от него дочерние ключи. Для действия ON DELETE CASCADE это выражается в том, что каждая строка дочерней таблицы, которая ассоциирована с удаляемой родительской строкой, также будет удалена. Для действия ON UPDATE CASCADE это выражается в том, что значения, сохранённые в зависящем дочернем ключе, будут заменены на новые значения родительского ключа.

3.3. Типы данных

Символьные типы данных:

- CHAR(n) - символьные строки фиксированной длины. Длина строки определяется параметром n. CHAR без параметра соответствует CHAR(1). Для хранения таких данных всегда отводится n байт вне зависимости от реальной длины строки.
- VARCHAR(n) - символьная строка переменной длины. Для хранения данных этого типа отводится число байт, соответствующее реальной длине строки.

Целые типы данных:

- SMALLINT - короткое целое (2 байта)
- INTEGER - обычное целое (4 байта)
- BIGINT - длинное целое (8 байт)

Вещественные типы данных:

- REAL - числа с плавающей точкой (4 байта).
- DOUBLE PRESCISION - числа с плавающей точкой (8 байт).
- NUMERIC(p,n) - тип данных аналогичный FLOAT с числом значащих цифр p и точностью n.

Дата и время - используются для хранения даты, времени и их комбинаций:

- DATE - тип данных для хранения даты.
- TIME - тип данных для хранения времени.
- TIMESTAMP - тип данных для хранения моментов времени (год + месяц + день + часы + минуты + секунды + доли секунд).

Двоичные типы данных - позволяют хранить данные любого объема в двоичном коде (оцифрованные изображения, исполняемые файлы и т.д.):

- BYTEA

4. Выполнение работы

4.1. Инициализация

Листинг 1: init.sql

```sql
DROP SCHEMA IF EXISTS public CASCADE;
CREATE SCHEMA public;

CREATE TYPE drink_type AS ENUM (
  'пиво и сидр',
  'водка и настойки',
  'ром',
  'текила',
  'виски',
  'биттеры',
  'вино',
  'бренди и коньяк',
  'шоты',
  'коктейли',
  'безалкогольные'
  );

CREATE TABLE components
(
  component_id serial                NOT NULL,
  title        character varying(50) NOT NULL UNIQUE,
  alcohol      double precision,
  primary key (component_id)
);

CREATE TABLE drinks
(
  drink_id      serial                NOT NULL,
  title         character varying(50) NOT NULL UNIQUE,
  rating        double precision,
  volume        double precision,
  alcohol       double precision,
  average_price double precision,
  drink_type    drink_type,
  primary key (drink_id)
);

CREATE TABLE components_drinks
(
  component_id int NOT NULL references components (component_id) on delete cascade,
  drink_id     int NOT NULL references drinks (drink_id) on delete cascade,
  quantity     double precision
);

```

```sql
CREATE TABLE places
(
  place_id     serial                NOT NULL,
  title        character varying(50) NOT NULL,
  address      text,
  rating       double precision,
  average_bill double precision,
  primary key (place_id)
);

CREATE TABLE places_drinks
(
  place_id integer NOT NULL references places (place_id) on delete cascade,
  drink_id integer NOT NULL references drinks (drink_id) on delete cascade
);

CREATE TABLE discounts
(
  discount_id serial           NOT NULL,
  place_id    integer          NOT NULL references places (place_id) on delete cascade,
  drink_type  drink_type       NOT NULL,
  amount      double precision NOT NULL,
  description text,
  weekday     integer,
  time_start  time,
  time_end    time,
  primary key (discount_id)
);

CREATE TABLE food
(
  food_id       serial                NOT NULL,
  title         character varying(50) NOT NULL UNIQUE,
  rating        double precision,
  volume        double precision,
  average_price double precision,
  primary key (food_id)
);

CREATE TABLE places_food
(
  place_id integer NOT NULL references places (place_id) on delete cascade,
  food_id integer NOT NULL references food (food_id) on delete cascade
);
```

4.2. Заполнение

Листинг 2: fill.sql

```sql
--FILL
```

```
insert into components(title, alcohol)
values ('водка', 40),
       ('гренадин', 0),
       ('соус табаско', 0);

insert into drinks(title, volume, alcohol, drink_type)
values ('боярский', 50, 40 / 2, 'шоты');

insert into components_drinks(component_id, drink_id, quantity)
values ((select component_id from components where title = 'водка'),
        (select drink_id from drinks where title = 'боярский'), 1),
       ((select component_id from components where title = 'гренадин'),
        (select drink_id from drinks where title = 'боярский'), 1),
       ((select component_id from components where title = 'соус табаско'),
        (select drink_id from drinks where title = 'боярский'), 1);


insert into components(title, alcohol)
values ('томатный сок', 0);

insert into drinks(title, volume, alcohol, drink_type)
values ('кровавая мэри', 50, 40 / 2, 'шоты');

insert into components_drinks(component_id, drink_id)
values ((select component_id from components where title = 'водка'),
        (select drink_id from drinks where title = 'кровавая мэри')),
       ((select component_id from components where title = 'томатный сок'),
        (select drink_id from drinks where title = 'кровавая мэри')),
       ((select component_id from components where title = 'соус табаско'),
        (select drink_id from drinks where title = 'кровавая мэри'));

insert into places(title, address)
values ('Контакт бар', 'пр. Просвещения, 25'),
       ('Контакт бар', 'ул. Садовая, 35'),
       ('Контакт бар', 'пр. Владимирский, 17'),
       ('Контакт бар', 'пр. Коломяжский, 15, к.2'),
       ('Контакт бар', 'пл. Стачек, 7, лит. А'),
       ('Контакт бар', 'ул. Гаккелевская, 34'),
       ('Контакт бар', 'ул. Марата, 7'),
       ('Контакт бар', 'Средний пр-т. ВО, 28'),
       ('Контакт бар', 'пр.Чернышевского 11/57'),
       ('Контакт бар', 'ул. Бухарестская, 74'),
       ('СПБ бар', 'Каменноостровский пр., 37');

insert into food(title)
values ('стрипсы'),
       ('начос');

insert into places_drinks(place_id, drink_id)
select distinct place_id, drink_id
```

```sql
from places,
     drinks
where drinks.title = 'боярский'
  and places.title = 'Контакт бар';

insert into places_food(place_id, food_id)
select distinct place_id, food_id
from places,
     food
where food.title = 'стрипсы'
  and places.title = 'Контакт бар';

insert into places_food(place_id, food_id)
select distinct place_id, food_id
from places,
     food
where food.title = 'начос'
  and places.title = 'Контакт бар';

insert into discounts(place_id, drink_type, amount, description)
select distinct place_id, 'пиво и сидр'::drink_type, 0.15, 'Цены для друзей на пиво!'
from places
where places.title = 'Контакт бар';
```

4.3. Изменение

Формулировка требуемых изменений:

1. Добавить цены на позиции меню в каждом конкретном баре.
2. Добавить поставки блюд и ингредиентов для коктейлей в бары.

Листинг 3: change.sql

```sql
--ISSUE
alter table places_drinks
  add column price double precision;

alter table places_food
  add column price double precision;

alter table places_drinks
  add column amount int;

alter table places_food
  add column amount int;

create table supplies_food
(
  supplies_food_id serial not null,
```

```
  place_id           int references places (place_id) on delete cascade,
  food_id            int references food (food_id) on delete cascade,
  amount             int,
  price_per_item     double precision,
  date               timestamp,
  primary key (supplies_food_id)
);

create table supplies_drinks
(
  supplies_drinks_id serial not null,
  place_id           int references places (place_id) on delete cascade,
  drink_id           int references drinks (drink_id) on delete cascade,
  amount             int,
  price_per_item     double precision,
  date               timestamp,
  primary key (supplies_drinks_id)
);

insert into supplies_food(place_id, food_id)
select distinct place_id, food_id
from places,
     food
where food.title = 'начос'
  and places.title = 'Контакт бар';

insert into supplies_drinks(place_id, drink_id)
select distinct place_id, drink_id
from places,
     drinks
where drinks.title = 'боярский'
  and places.title = 'Контакт бар';
```

5. Результат работы

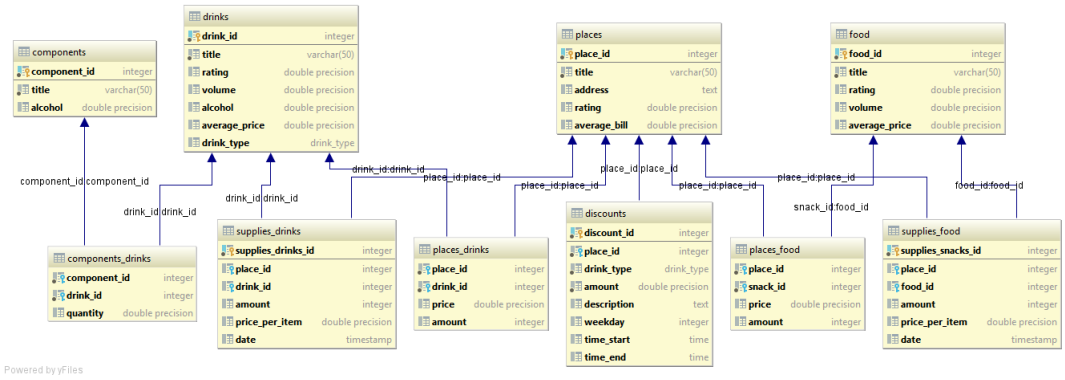

Рисунок 5.1. Структура базы данных

	drink_id	title	rating	volume	alcohol	average_price	drink_type
1	1	боярский	<null>	50	20	<null>	шоты
2	2	кровавая мэри	<null>	50	20	<null>	шоты

Рисунок 5.2. Содержимое таблицы drinks

	place_id	title	address	rating	average_bill
1	1	Контакт бар	пр. Просвещения, 25	<null>	<null>
2	2	Контакт бар	ул. Садовая, 35	<null>	<null>
3	3	Контакт бар	пр. Владимирский, 17	<null>	<null>
4	4	Контакт бар	пр. Коломяжский, 15, к.2	<null>	<null>
5	5	Контакт бар	пл. Стачек, 7, лит. А	<null>	<null>
6	6	Контакт бар	ул. Гаккелевская, 34	<null>	<null>
7	7	Контакт бар	ул. Марата, 7	<null>	<null>
8	8	Контакт бар	Средний пр-т. ВО, 28	<null>	<null>
9	9	Контакт бар	пр.Чернышевского 11/57	<null>	<null>
10	10	Контакт бар	ул. Бухарестская, 74	<null>	<null>
11	11	СПБ бар	Каменноостровский пр., 37	<null>	<null>

Рисунок 5.3. Содержимое таблицы places

	place_id	drink_id	price	amount
1	1	1	<null>	<null>
2	2	1	<null>	<null>
3	3	1	<null>	<null>
4	4	1	<null>	<null>
5	5	1	<null>	<null>
6	6	1	<null>	<null>
7	7	1	<null>	<null>
8	8	1	<null>	<null>
9	9	1	<null>	<null>
10	10	1	<null>	<null>

Рисунок 5.4. Содержимое таблицы places_drinks

6. Выводы

В ходе выполнения данной работы были написаны 3 скрипта на языке PostgreSQL: создающий таблицы; наполняющий таблицы данными; изменяющий таблицы и добавляющий новые. Таким образом, было осуществлено моё знакомство с основами проектирования схемы БД, языком описания БД SQL-DDL.