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

Базы данных

Отчет по лабораторной работе №3
Генерация тестовых данных

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

Санкт-Петербург
2019

Содержание

1. Цель работы

Сформировать набор данных, позволяющий производить операции на реальных объемах данных.

2. Программа работы

1. Реализация в виде программы параметризуемого генератора, который позволит сформировать набор связанных данных в каждой таблице.
2. Частные требования к генератору, набору данных и результирующему набору данных:
   - количество записей в справочных таблицах должно соответствовать ограничениям предметной области
   - количество записей в таблицах, хранящих информацию об объектах или субъектах должно быть параметром генерации
   - значения для внешних ключей необходимо брать из связанных таблиц

3. Выполнение работы

В качестве языка программирования для параметризуемого создания генератора был выбран Python 3.6 и библиотека psycopg2 - самая популярная библиотека для работы с PostgreSQL.

В ходе выполнения работы была написана программа, реализующая генератор. С полным кодом программы можно ознакомиться по адресу:

http://gitlab.icc.spbstu.ru/grafa/wine_card/blob/master/lab3/src/fill.py

Ниже рассмотрим фрагмент программы, заполняющий таблицы components, drinks, а также вспомогательную таблицу для их связи.

Листинг 1. fill.py

```
28 def fill_components_and_drinks(components_size=100, drinks_size=100, max_components_per_drink=10):
29     with conn.cursor() as cursor:
30         conn.autocommit = True
31         for i in range(components_size):
32             rand_component = rand_string()
33             rand_alco = random.randint(0, 100)
34             components.append((rand_component, rand_alco))
35 
36         insert_components = sql.SQL('INSERT INTO components(title, alcohol) VALUES {}').format(
37             sql.SQL(',').join(map(sql.Literal, components))
38         )
39         cursor.execute(insert_components)
40 
41         for i in range(drinks_size):
42             rand_drink = rand_string()
43             rand_vol = random.randint(100, 1000)
44             rand_alco = random.randint(0, 100)
```

```
            drinks.append((rand_drink, rand_vol, rand_alco, drink_type[random.randint(0, len(drink_type
        insert_drinks = sql.SQL('INSERT INTO drinks(title, volume, alcohol, drink_type) VALUES {}').for
            sql.SQL(',').join(map(sql.Literal, drinks))
        )
        cursor.execute(insert_drinks)

        components_drinks = []  # [component_id, drink_id, quantity]
        for drink in drinks:
            cursor.execute("select drink_id from drinks where title = '{}'".format(drink[0]))
            rand_drink_id = cursor.fetchone()
            for i in range(0, max_components_per_drink):
                cursor.execute("select component_id from components where title = '{}';".format(
                    components[random.randint(0, len(components) - 1)][0]))
                rand_component_id = cursor.fetchone()
                quantity = random.randint(0, 10)
                components_drinks.append((rand_component_id, rand_drink_id, quantity))

        insert_components_drinks = sql.SQL(
            'INSERT INTO components_drinks(component_id, drink_id, quantity) VALUES {}').format(
            sql.SQL(',').join(map(sql.Literal, components_drinks))
        )
        cursor.execute(insert_components_drinks)
        print("Successfully filled components and drinks!")
```

В качестве параметров данной функции задается количество строк, генерируемых в таблицу components, drinks, а также максимальное число связей между напитком и его компонентами.

В первом цикле for создаются и сохраняются в массив components случайные строки размером 1-50 и числа 0-100, которые с помощью запроса insert заполнят таблицу. Эти числа соответствуют полям title и alcohol таблицы components.

Подобная процедура повторяется для массива и таблицы drinks.

Затем с помощью цикла for мы проходим массив drinks, берём из него title, по нему ищем соответствующий id напитка. После этого мы проходим по вложенному циклу и подобным образом берем из таблицы components id. Затем генерируем случайное число и вставляем 3 полученных число в массив components_drinks, и заполняем таблицу components_drinks с помощью запроса insert и полученного массива.

4. Результат работы

Запустим данную программу через терминал.

```
(base) grafa@KRAB:~/Desktop/wine_card/lab3/src$ python fill.py
Successfully filled components and drinks!
Successfully filled places!
Successfully filled food!
Successfully filled places_food!
Successfully filled places_drinks!
Successfully filled discounts!
Successfully filled supplies_drinks!
Successfully filled supplies_food!
```

Примеры генерируемых данных:

	component_id	title	1	alcohol
1	77	08S1yNQ8ZjZ3y96Tf9Ls1DpNyesytdCkPhB6eJ2JVT6YdbRyj6		36
2	35	1zlThZSFgmlkzfV6OS6		27
3	102	2c4tSL2VJxz5CY7zdLC0QNdB5b6cyE8vBjODJeriwvW5		50
4	62	3Chhrthaw7frp5HqXGM3BeEjG1mC4lwBPQzU7wvcfb4a5q		34
5	33	3tCddpYeTdp350fMCLPayRirWa6ppG		15
6	51	4dUY0fJw		14
7	70	4NI2EArILJggK0bKIOX67uNIhLc5WqkQPgd		82
8	65	5T3UytedA6MTDfimYe8EdD		85
9	67	6aZ9VaI0xfL7zCyfxwP7w04iA2hXJpGL7kV0gYTXuG7znVJ		43
10	15	7UCqX5R3uWcBZW1KmbkWtlVPybWG8w9qbNxpZmnP11		30
11	36	9GXbsaxHC0o3gmxkJJg1kvlCU2gpOZrVvaCcYRRUArb0ngx		46
12	44	9pNjd6BjIivmkC2B4v8Wky4IueAYYxvCuojVQaz9Sr4Jh		61
13	11	aPizd2RQG		32
14	90	AqS3eARtXv1Pu6MgzJZH5ehG		24
15	72	arVxdjii5QfPLACbX		79
16	100	axRqSZh1mwWzgFuT		57
17	38	BCCDxZJi8dVnV882EwoWxtlczOhQVKgWWjQ		56
18	26	BJzKHYd61nEoNRxH5lqEc1iVk0iGu		61
19	5	bW872YM		21
20	75	CdJvOIIzWwaMNlB9pBExkv		35
21	95	ci1VCDalOIbmsstIVYyS6AL6wCx5pHVj6EllfbFHgdi		4
22	45	CnacVZ0ptHAYg8ZQg83Y282Cqju45Uk9ui8vBZ4675		84
23	84	cnXWg6KcNZeRh		75
24	48	cRXqiLTrKncPVbNNFbkwNHMlpouLXE		19
25	104	CspqkRosZfyw2t0nuMdvvuRh6l		27
26	82	CTvpTq7IOHk6RUkxMotEKZyGXC18XDz		51
27	73	cxILngnfcmhNVpyWIwB3jnKlFB8mvz2BjQS3gAQGlDGsoE3Uhi		83
28	85	CZ41mspq4H4Uc8kX9wj		88
29	88	DIHjVKnnaWgNCXMUVSS3O0eNALoZKmVGJWyGVILJyq8rkh		78
30	16	DyZNK5uMKBgnPW0G		67

Рисунок 4.1. Содержимое таблицы components

	supplies_drinks_id 1	place_id	drink_id	amount	price_per_item	date
1	1010	37	65	16	118	2009-03-23 14:41:31.558981
2	1009	101	20	48	394	2004-03-04 06:21:03.558598
3	1008	19	60	30	677	2013-10-12 17:38:45.558069
4	1007	72	60	82	518	2018-04-14 22:30:06.557638
5	1006	39	40	31	758	2016-11-06 13:12:46.557197
6	1005	47	100	99	935	2017-04-24 10:53:00.556765
7	1004	26	70	50	589	2013-09-01 15:08:18.556392
8	1003	55	92	87	841	2002-08-10 18:27:01.556015
9	1002	12	65	80	76	2002-01-31 21:27:42.555646
10	1001	63	66	62	414	2007-04-21 20:03:33.555317
11	1000	92	51	70	723	2011-08-19 18:11:11.554767
12	999	19	72	90	660	2004-01-18 01:02:13.554202
13	998	26	74	7	606	2012-05-02 19:59:19.553738
14	997	17	88	34	825	2009-09-28 08:09:10.553271
15	996	69	95	54	44	2014-10-06 12:58:35.552788
16	995	51	16	89	11	2015-05-16 04:30:34.552446
17	994	45	34	65	94	2006-12-23 00:19:18.552101
18	993	44	43	90	902	2014-09-30 05:53:25.551754
19	992	59	69	89	643	2014-03-12 15:58:16.551411
20	991	85	94	84	673	2018-08-03 09:09:48.551064
21	990	61	42	83	529	2015-12-03 08:54:03.550714
22	989	94	42	82	882	2014-08-11 16:56:56.550359
23	988	104	41	69	258	2009-09-24 03:44:08.550013
24	987	66	10	79	86	2014-07-31 14:12:40.549664
25	986	80	19	64	56	2009-09-23 12:58:30.549307
26	985	96	26	58	290	2018-10-18 21:14:24.548962

Рисунок 4.2. Содержимое таблицы supplies_drinks

	discount_id	place_id	drink_type	amount	description
1	18	15	виски	0.7414360619367385	2KeaArmVOq6JhbN0QVa3iR
2	11	67	текила	0.7198427226334619	A2
3	12	70	ром	0.7290235045170657	Esug3keLhJYyEMeZrnr3ZpeDR2QwzjJ0J3o
4	13	71	пиво и сидр	0.182336489667546	HAuo3YxN3lSDmDmnjeWn6Lcy2CtZXidNx8TmI57
5	15	16	бренди и коньяк	0.6130395355830033	hIYkmTHhH1GUYYaCY
6	20	67	шоты	0.12565166052860477	L82r0bf
7	14	62	биттеры	0.1062119366335399	NuRWcIMMGHcAXB7WZwLj3
8	16	54	водка и настойки	0.9232124366668905	vN7PiTctwLUikGry
9	17	18	ром	0.959227174910468	Z8DaatrbaliOMJKdepEoOfkKiMWTeGQahI6Aaggp
10	19	92	ром	0.06219161989266875	zDGbC3oKnf5S8eTDqjJc7TbnQWD2

Рисунок 4.3. Содержимое таблицы discounts

5. Выводы

В ходе выполнения данной работы на языке программирования Python был написан параметризуемый генератор. В качестве параметра данного генератора можно указать количество записей в таблицах, как это и требовало задание.