

第三次数据库作业

19335286 郑有为

1. 编写一个函数，函数名为 `get_student_phone`，无接收参数，返回一个随机的手机号，长度11位，手机号以'159'或'137'开头，要求任意满足该要求的手机号能等概率生成

函数代码：

```
1  /* DROP 清除之前创建的函数 */
2  DROP FUNCTION get_student_phone();
3
4  /* get_student_phone 函数 */
5  CREATE FUNCTION get_student_phone() RETURNS bigint AS $$
6  DECLARE
7      head1 bigint := 13700000000;
8      head2 bigint := 15900000000;
9      s_phone bigint := 0;
10     ten int := 1;
11 BEGIN
12     IF RANDOM() > 0.5 THEN
13         s_phone := head1;
14     ELSE
15         s_phone := head2;
16     END IF;
17
18     FOR i IN 1..8 LOOP
19         s_phone := s_phone + ten * FLOOR(RANDOM() * 10);
20         ten := ten * 10;
21     END LOOP;
22
23     RETURN s_phone;
24 END;
25 $$ LANGUAGE plpgsql;
26
27 /* SELECT调用函数 */
28 select get_student_phone();
```

注解：

- 使用一个长整型 `bigint` 来记录电话号码；
- 使用整数加法来实现固定最高位为 137 或者 159：
 - 若取得随机数位于 $[0.5, 1)$ ，则 + 13700000000；
 - 若取得随机数位于 $[0, 0.5)$ ，则 + 15900000000；
- 对于剩下的 8 位数字，每次使用随机数获得 0~9 内的一个数字，然后依次加到号码的后八位上。

返回结果：三组不同的输出结果

```
max@ubuntu: ~  
postgres=# select get_student_phone();  
get_student_phone  
-----  
15987261488  
(1 行记录)  
  
postgres=# select get_student_phone();  
get_student_phone  
-----  
13725118551  
(1 行记录)  
  
postgres=# select get_student_phone();  
get_student_phone  
-----  
15988872287  
(1 行记录)  
  
postgres=#
```

2. 编写一个函数，函数名为 `get_student_date`，无接收参数，返回一个随机的日期，日期格式为 'YYYY-MM-DD'。要求返回的日期区间为 [2020-01-01, 2021-12-31]，其中，要求生成2020年份概率为60%，生成2021年份概率为40%，此外，月和日则是等概率返回。

函数代码：

```
1  /* DROP 清除之前创建的函数 */  
2  DROP FUNCTION get_student_date();  
3  
4  /* get_student_date 函数 */  
5  CREATE FUNCTION get_student_date() RETURNS date AS $$  
6  DECLARE  
7      s_date date;  
8      s_day int = 0;  
9  BEGIN  
10     IF RANDOM() < 0.6 THEN  
11         s_date := make_date(2021, 1, 1);  
12         s_day := FLOOR(RANDOM() * 364);  
13         s_date := s_date + s_day;  
14     ELSE  
15         s_date := make_date(2020, 1, 1);  
16         s_day := FLOOR(RANDOM() * 365);  
17         s_date := s_date + s_day;  
18     END IF;  
19     RETURN s_date;  
20 END;  
21 $$ LANGUAGE plpgsql;  
22  
23 /* SELECT调用函数 */  
24 select get_student_date();
```

注解：

- 使用 `date` 类型保存日期，对年份随机数：
 - 若取得随机数位于 [0.0, 0.6) (即60%概率)，则生成2020年；
 - 若取得随机数位于 [0.6, 1) (即40%概率)，则生成2021年；
- 接下来使用日期与整形的运算实现随机获得月份和日：

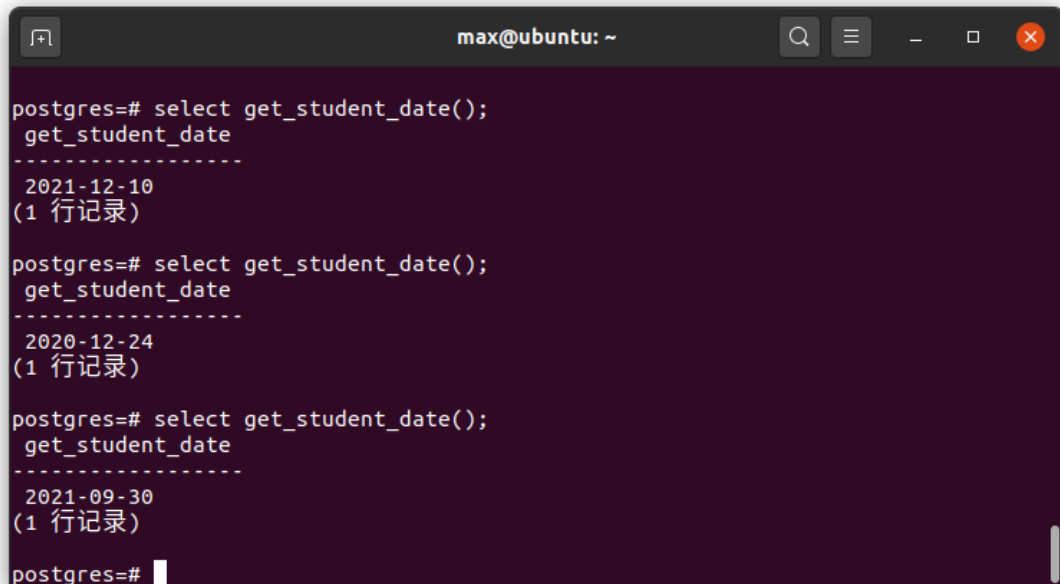
这样可以避开对不同月份的天数的判定，例如：

```
date '2001-09-28' + integer '7'
```

```
date '2001-10-05'
```

- 2020年共有 366 天，故随机取 [0,366) 之间的一个整数作为天数，2020年1月1号加上这个天数即可获得随机日期；
- 2021年共有 365 天，故随机取 [0,365) 之间的一个整数作为天数，2021年1月1号加上这个天数即可获得随机日期。

返回结果：三组不同的输出结果



```
max@ubuntu: ~  
postgres=# select get_student_date();  
get_student_date  
-----  
2021-12-10  
(1 行记录)  
  
postgres=# select get_student_date();  
get_student_date  
-----  
2020-12-24  
(1 行记录)  
  
postgres=# select get_student_date();  
get_student_date  
-----  
2021-09-30  
(1 行记录)  
  
postgres=#
```

3. 编写一个函数，函数名为 `create_student_table`，无接收参数。在该函数中，新建一个数据表 `student`，该数据表拥有3个字段，分别是 `student_id`, `phone_num`, `enrollment_date`，其中 `student_id` 为自增的序列，从1开始自增，且为主键；然后，往该数据表新增 15条记录，这 15条记录中，`phone_num` 和 `enrollment_date` 分别使用上述自己编写的第一个和第二个函数生成。最后返回该表。该函数理应可以连续调用多次，每次生成并返回的表都不一样。

函数代码：

```
1  /* DROP 清除之前创建的函数 */  
2  DROP FUNCTION create_student_table();  
3  
4  /* create_student_table 函数 */  
5  CREATE FUNCTION create_student_table(OUT student_id int, OUT phone_num  
6  bigint, OUT enrollment_date date) RETURNS SETOF record AS $$  
7  DECLARE  
8      student_table record;  
9  BEGIN  
10     FOR i IN 1..15 LOOP  
11         student_id := i;  
12         phone_num := get_student_phone();  
13         enrollment_date = get_student_date();  
14         RETURN NEXT;  
15     END LOOP;  
16 END;  
17 $$ LANGUAGE plpgsql;  
18  
19 /* SELECT * FROM 查看返回的表 */  
select * from create_student_table();
```

注解:

- 使用 `SETOF record` (记录的集合) 作为返回值, 并指定返回的记录各元素的类型和名称:
`OUT student_id int, OUT phone_num bigint, OUT enrollment_date date`
- 使用循环生成 15 组记录, 每次都调用 `get_student_phone()` 和 `get_student_date()`, 并使用 `RETURN NEXT` 将每一次结果添加到输出集合中。

返回结果: 两组不同的输出结果

```
max@ubuntu: ~  
postgres=#  
postgres=# select * from create_student_table();  
 student_id | phone_num | enrollment_date  
-----+-----+-----  
      1 | 13751927709 | 2020-07-30  
      2 | 13784658953 | 2021-03-04  
      3 | 15945109325 | 2021-08-01  
      4 | 13703546575 | 2021-11-08  
      5 | 13718085723 | 2021-01-11  
      6 | 15911687041 | 2021-02-06  
      7 | 13713277697 | 2021-06-24  
      8 | 15976825701 | 2021-11-17  
      9 | 15964916256 | 2021-05-07  
     10 | 13743413320 | 2020-01-27  
     11 | 13762095043 | 2021-09-04  
     12 | 15914087754 | 2021-11-16  
     13 | 13739734195 | 2021-04-19  
     14 | 13786880155 | 2020-10-13  
     15 | 13701058060 | 2021-09-28  
(15 行记录)  
  
postgres=# select * from create_student_table();  
 student_id | phone_num | enrollment_date  
-----+-----+-----  
      1 | 13711849292 | 2021-06-07  
      2 | 13716916195 | 2021-09-13  
      3 | 15933649797 | 2020-12-18  
      4 | 13770414641 | 2021-05-03  
      5 | 15967988183 | 2020-07-23  
      6 | 13772932880 | 2021-06-03  
      7 | 15962070382 | 2020-02-05  
      8 | 15906571913 | 2020-07-03  
      9 | 15908356999 | 2021-07-04  
     10 | 15958400134 | 2020-05-19  
     11 | 13780464643 | 2021-06-23  
     12 | 13749978843 | 2021-09-13  
     13 | 13741382999 | 2020-12-19  
     14 | 13702073407 | 2021-08-31  
     15 | 15923494328 | 2021-11-09  
(15 行记录)  
  
postgres=#
```

4. 使用 `student` 表, 找出所有 `enrollment_date` 在 2020 年 7 月 1 日 (包括这一天) 之后的学生, 并输出其 `phone_num`。

查询代码:

```
1 | SELECT *  
2 | FROM create_student_table()  
3 | WHERE enrollment_date > date '2020-07-01';
```

注解:

- 使用 `date` 加 符合规范的日期字符串即可生成一个日期变量
- 对 `date` 可以直接使用比较运算符, 靠后的日期大于靠前的日期

查询结果:

```
max@ubuntu: ~  
postgres=# SELECT *  
postgres=# FROM create_student_table()  
postgres=# WHERE enrollment_date > date '2020-07-01';  
 student_id | phone_num | enrollment_date  
-----+-----+-----  
          3 | 13753215368 | 2021-05-05  
          4 | 15913870561 | 2021-11-08  
          5 | 13788150406 | 2021-06-27  
          6 | 13740049794 | 2021-06-11  
          7 | 13724343020 | 2021-11-20  
          8 | 15979935245 | 2021-05-27  
          9 | 15973609593 | 2021-03-31  
         10 | 13797024870 | 2021-03-24  
         11 | 13772321642 | 2021-10-05  
         13 | 15912403797 | 2021-05-11  
         14 | 13710064742 | 2021-11-30  
(11 行记录)  
  
postgres=# SELECT *  
postgres=# FROM create_student_table()  
postgres=# WHERE enrollment_date > date '2020-07-01';  
 student_id | phone_num | enrollment_date  
-----+-----+-----  
          1 | 15959876662 | 2020-08-02  
          2 | 15949754640 | 2021-03-25  
          4 | 15905287947 | 2021-01-18  
          5 | 15981525207 | 2021-08-16  
          6 | 15963280167 | 2021-06-26  
          7 | 13742992435 | 2021-12-02  
          8 | 13770926250 | 2021-08-15  
          9 | 13786424885 | 2020-08-29  
         10 | 13768600792 | 2021-11-12  
         11 | 15958290805 | 2021-06-04  
         12 | 15900255880 | 2020-10-03  
         14 | 13715113845 | 2021-10-04  
         15 | 15993951570 | 2020-11-26  
(13 行记录)  
  
postgres=#
```