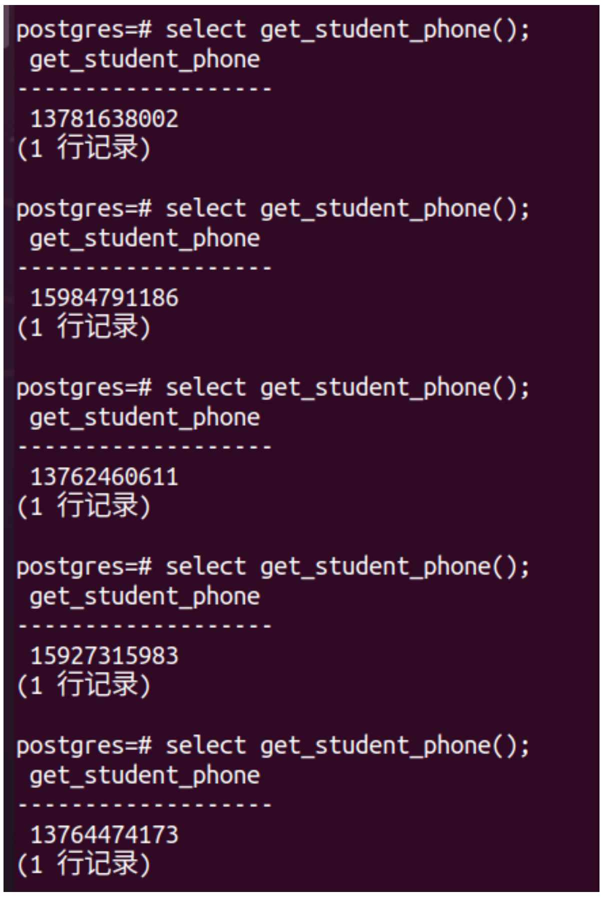
1. 编写一个函数，函数名为 get_student_phone，无接收参数，返回一个随机的手机号，长度 11 位，手机号以'159'或 '137'开头，要求任意满足该要求的手机号能等概率生成。

SQL 语句：

```
create or replace function get_student_phone() returns character varying
as $$
declare
result varchar(50);
num integer;
begin
select array_to_string(array(select chr((48 + round(random() * 9)) :: integer) from generate_series(1,8)), '') into result;
num = random() * 1;
if num = 1 then
result = '137' || result;
else
result = '159' || result;
end if;
return result;
end;
$$
language plpgsql;
```

百分之五十的概率生成 137 或者 159 开头的电话号码，后面 8 位电话号码随机等概率生成

查询结果：



```
postgres=# select get_student_phone();
 get_student_phone
-------------------
 13781638002
(1 行记录)

postgres=# select get_student_phone();
 get_student_phone
-------------------
 15984791186
(1 行记录)

postgres=# select get_student_phone();
 get_student_phone
-------------------
 13762460611
(1 行记录)

postgres=# select get_student_phone();
 get_student_phone
-------------------
 15927315983
(1 行记录)

postgres=# select get_student_phone();
 get_student_phone
-------------------
 13764474173
(1 行记录)
```

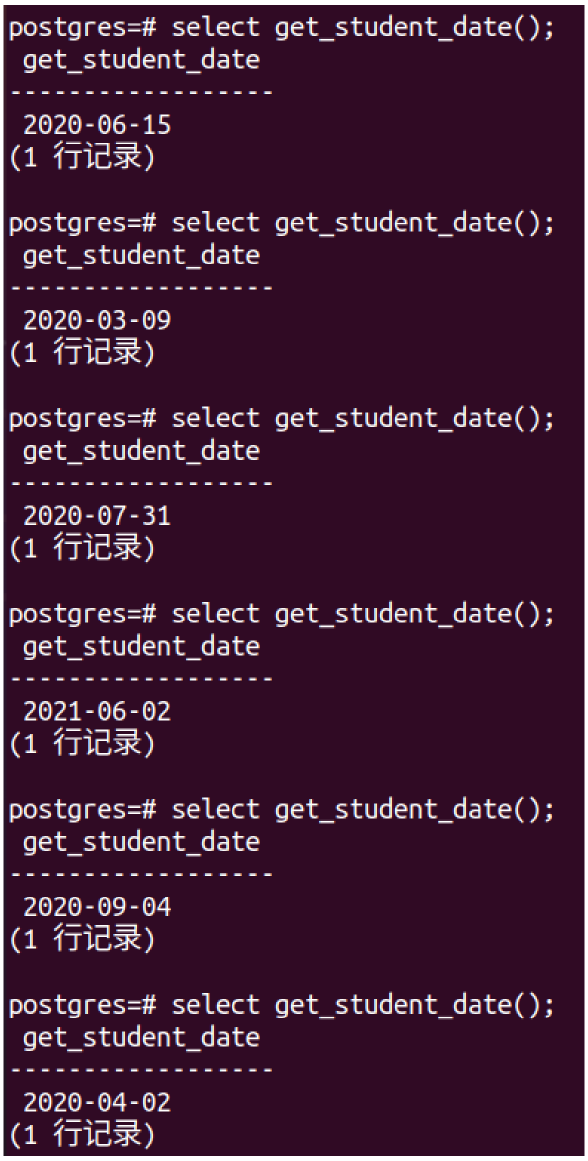
2. 编写一个函数，函数名为 get_student_date，无接收参数，返回一个随机的日期，日期格式为'YYYY-MMDD'。要求返回的日期区间为[2020-01-01, 2021-12-31], 其中, 要求生成 2020 年份概率为 60%，生成 2021 年份概率为 40%，此外，月和日则是等概率返回。

SQL 语句：

```
create or replace function get_student_date() returns character varying
as $$
declare
result varchar(100);
month integer;
year integer;
day integer;
num integer;
begin
num = random() * 9;
if num >= 6 then
year = 2021;
result = '2021-';
else
year = 2020;
result = '2020-';
end if;

month = random() * 12 + 1;
if month < 10 then
result = result || '0' || cast(month as varchar) || '-';
else
result = result || cast(month as varchar) || '-';
end if;

if month = 1 or month = 3 or month = 5 or month = 7 or month = 8 or month = 10 or month
= 12 then
day = random() * 31 + 1;
end if;
if month = 4 or month = 6 or month = 9 or month = 11 then
day = random() * 30 + 1;
end if;
if month = 2 and year = 2020 then
day = random() * 29 + 1;
end if;
if month = 2 and year = 2021 then
day = random() * 28 + 1;
end if;

if day < 10 then
```

```
result = result || '0' || cast(day as varchar) ;
else
result = result || cast(day as varchar);
end if;

return result;
end;
$$
language plpgsql;
```

查询结果:



```
postgres=# select get_student_date();
 get_student_date
------------------
 2020-06-15
(1 行记录)

postgres=# select get_student_date();
 get_student_date
------------------
 2020-03-09
(1 行记录)

postgres=# select get_student_date();
 get_student_date
------------------
 2020-07-31
(1 行记录)

postgres=# select get_student_date();
 get_student_date
------------------
 2021-06-02
(1 行记录)

postgres=# select get_student_date();
 get_student_date
------------------
 2020-09-04
(1 行记录)

postgres=# select get_student_date();
 get_student_date
------------------
 2020-04-02
(1 行记录)
```

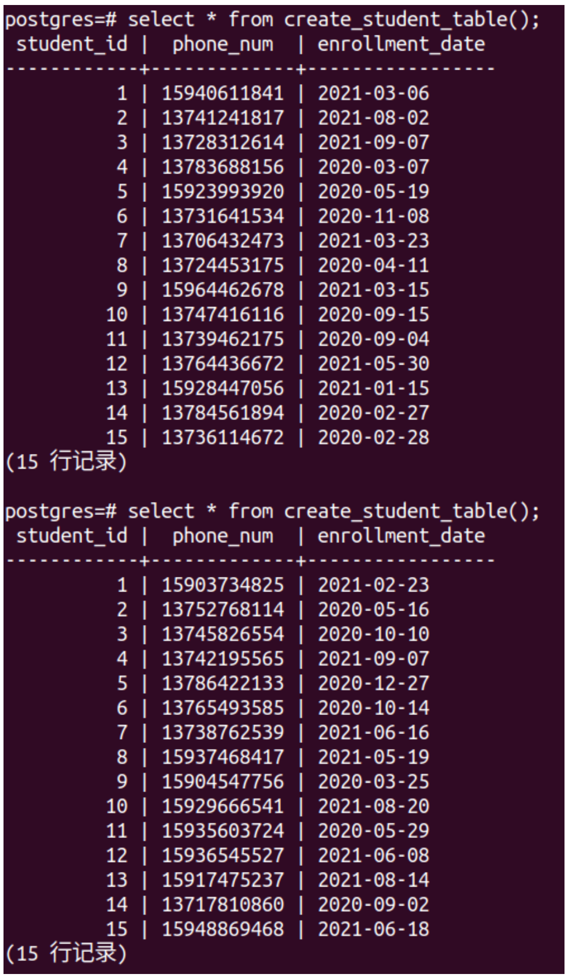
3. 编写一个函数，函数名为 create_student_table，无接收参数。在该函数中，新建一个数据表 student，该数据表拥有 3 个字段，分别是 student_id, phone_num, enrollment_date，其中 student_id 为自增的序列，从 1 开始自增，且为主键；然后，往该数据表新增 15 条记录，这 15 条记录中，phone_num 和 enrollment_date 分别使用上述自己编写的第一个和第二个函数生成。最后返回该表。该函数理应可以连续调用多次，每次生成并返回的表都不一样。

SQL 语句：

```
create or replace function create_student_table()
returns table(student_id integer, phone_num varchar, enrollment_date varchar)
as $$
declare
num integer := 1;
begin
drop table if exists student;
create table student(student_id integer, phone_num varchar, enrollment_date varchar);
while num <=   15 loop
insert into student(student_id, phone_num, enrollment_date)
values(num,get_student_phone(),get_student_date());
num = num + 1;
end loop;
return query select * from student;
end;
$$
language plpgsql;
```

创建一个表格 student，通过调用 get_student_phone(),get_student_date()不断地往表格中插入数据

查询结果：



```
postgres=# select * from create_student_table();
 student_id |  phone_num  | enrollment_date
------------+-------------+-----------------
          1 | 15940611841 | 2021-03-06
          2 | 13741241817 | 2021-08-02
          3 | 13728312614 | 2021-09-07
          4 | 13783688156 | 2020-03-07
          5 | 15923993920 | 2020-05-19
          6 | 13731641534 | 2020-11-08
          7 | 13706432473 | 2021-03-23
          8 | 13724453175 | 2020-04-11
          9 | 15964462678 | 2021-03-15
         10 | 13747416116 | 2020-09-15
         11 | 13739462175 | 2020-09-04
         12 | 13764436672 | 2021-05-30
         13 | 15928447056 | 2021-01-15
         14 | 13784561894 | 2020-02-27
         15 | 13736114672 | 2020-02-28
(15 行记录)

postgres=# select * from create_student_table();
 student_id |  phone_num  | enrollment_date
------------+-------------+-----------------
          1 | 15903734825 | 2021-02-23
          2 | 13752768114 | 2020-05-16
          3 | 13745826554 | 2020-10-10
          4 | 13742195565 | 2021-09-07
          5 | 13786422133 | 2020-12-27
          6 | 13765493585 | 2020-10-14
          7 | 13738762539 | 2021-06-16
          8 | 15937468417 | 2021-05-19
          9 | 15904547756 | 2020-03-25
         10 | 15929666541 | 2021-08-20
         11 | 15935603724 | 2020-05-29
         12 | 15936545527 | 2021-06-08
         13 | 15917475237 | 2021-08-14
         14 | 13717810860 | 2020-09-02
         15 | 15948869468 | 2021-06-18
(15 行记录)
```

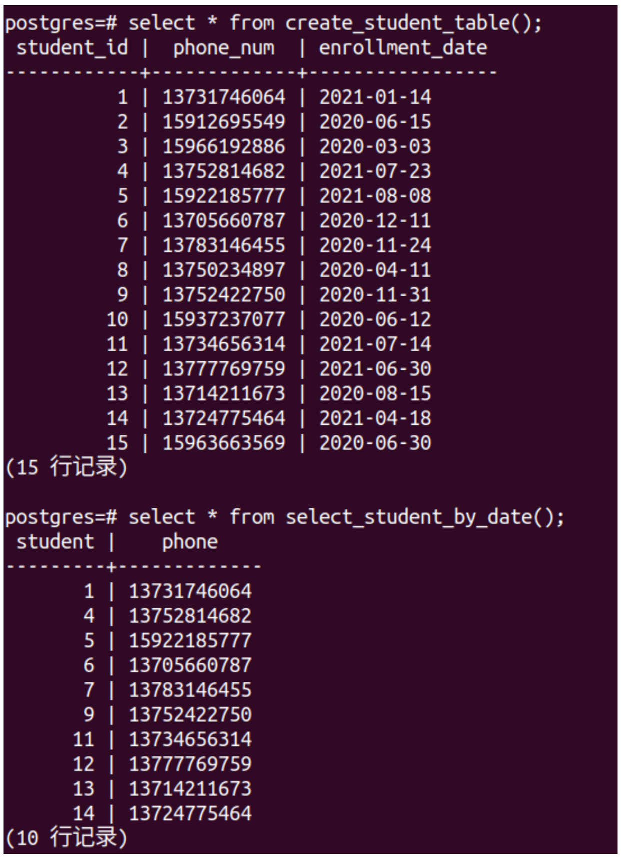
4. 查询：使用 student 表，找出所有 enrollment_date 在 2020 年 7 月 1 日（包括这一天）之后的学生，并输出其 phone_num。

SQL 语句：

```
create or replace function select_student_by_date()
returns table(student integer,phone varchar)
as $$
begin
return query select student_id, phone_num from student where enrollment_date >= '2020-07-01';
end;
$$
language plpgsql;
```

通过比较字符串大小，即可获得日期大于等于 2020-07-01 的学生

查询结果：



```
postgres=# select * from create_student_table();
 student_id |  phone_num  | enrollment_date
------------+-------------+-----------------
          1 | 13731746064 | 2021-01-14
          2 | 15912695549 | 2020-06-15
          3 | 15966192886 | 2020-03-03
          4 | 13752814682 | 2021-07-23
          5 | 15922185777 | 2021-08-08
          6 | 13705660787 | 2020-12-11
          7 | 13783146455 | 2020-11-24
          8 | 13750234897 | 2020-04-11
          9 | 13752422750 | 2020-11-31
         10 | 15937237077 | 2020-06-12
         11 | 13734656314 | 2021-07-14
         12 | 13777769759 | 2021-06-30
         13 | 13714211673 | 2020-08-15
         14 | 13724775464 | 2021-04-18
         15 | 15963663569 | 2020-06-30
(15 行记录)

postgres=# select * from select_student_by_date();
 student |    phone
---------+-------------
       1 | 13731746064
       4 | 13752814682
       5 | 15922185777
       6 | 13705660787
       7 | 13783146455
       9 | 13752422750
      11 | 13734656314
      12 | 13777769759
      13 | 13714211673
      14 | 13724775464
(10 行记录)
```