实习三 非关系数据

2000013094 陈嘉 2000013095 黄粤 2000013121 马佳媛

实习简介

本次实习主要是关于在关系数据库内处理非关系型数据的手段。
本次实习包括两个练习：练习一为窗口函数，练习二为JSON。
两个练习中，主要包含三个方面的内容：时序数据的窗口查询、使用递归查询生成层次结构、JSON数据的基本操作

练习一 窗口函数

本部分中，我们会在tushare的数据集上随机实现5个WorldQuant Alpha 101因子函数

数据导入

我们使用stock_demo.ipynb提供的数据抓取代码来获取数据
注：这部分代码由于平台原因无法正确运行，下文会提供替代方案

In [ ]:

```
!pip install tushare
!pip install pandas
!pip install numpy
import tushare as ts
import pandas as pd
import numpy as np

## set this to your tushare token
ts.set_token('2aebd8c845bbfca4719c9ae9b91e5b05ac5bb83a6c4b63add331dabc')
pro = ts.pro_api()

data = pro.daily(ts_code='600019.SH', start_date='20110101', end_date='20230101')
## sort the data in trade_date
data.sort_values('trade_date', inplace = True,ascending= True)
## format the trade-date to "year-month-date"
data['trade_date'] = pd.to_datetime(data['trade_date'], format="%Y%m%d")
```

In [ ]:

```
print(data.head(10))
```

替代方案：我们在本地环境中通过上述代码获取了数据，将其导出为data.csv文件，通过文件导入数据

In [2]:

```
import pandas as pd
import numpy as np

data = pd.read_csv('./data.csv', index_col=0)
```

In [3]:

```
print(data.head(10))
```

```
        ts_code  trade_date  open  high   low  close  pre_close  change  \
2817  600019.SH  2011-01-04  6.44  6.56  6.31   6.55       6.39    0.16
2816  600019.SH  2011-01-05  6.49  6.50  6.42   6.45       6.55   -0.10
2815  600019.SH  2011-01-06  6.46  6.50  6.39   6.46       6.45    0.01
2814  600019.SH  2011-01-07  6.46  6.68  6.41   6.56       6.46    0.10
2813  600019.SH  2011-01-10  6.54  6.64  6.51   6.52       6.56   -0.04
2812  600019.SH  2011-01-11  6.52  6.72  6.51   6.70       6.52    0.18
2811  600019.SH  2011-01-12  6.72  6.90  6.67   6.83       6.70    0.13
2810  600019.SH  2011-01-13  6.87  6.90  6.73   6.77       6.83   -0.06
2809  600019.SH  2011-01-14  6.77  6.77  6.61   6.71       6.77   -0.06
2808  600019.SH  2011-01-17  6.64  6.74  6.48   6.53       6.71   -0.18

      pct_chg        vol      amount
2817     2.50  572249.83  369381.672
2816    -1.53  334574.19  215978.272
2815     0.16  248061.88  159829.914
2814     1.55  562598.63  368965.887
2813    -0.61  422610.09  277610.464
2812     2.76  608401.78  403478.981
2811     1.94  815284.93  554413.060
2810    -0.88  511347.57  348201.644
2809    -0.89  415524.64  277649.450
2808    -2.68  524684.69  346912.702
```

接下来，我们将连接数据库，在数据库内建表，并将数据导入数据库

In [4]:

```
%load_ext sql
import pymysql
pymysql.install_as_MySQLdb()
%sql mysql://stu2000013094:stu2000013094@162.105.146.37:43306
%sql use stu2000013094;
```

```
 * mysql://stu2000013094:***@162.105.146.37:43306
0 rows affected.
```

Out[4]:

```
[]
```

In [5]:

```
import sqlalchemy

conn = sqlalchemy.create_engine('mysql+pymysql://stu2000013094:stu2000013094@162.105.146.37:43306
data.to_sql('tushare', conn, if_exists='replace', index=False)
```

In [6]:

```
%%sql
SELECT * FROM information_schema.columns WHERE table_name='tushare'
```

```
 * mysql://stu2000013094:***@162.105.146.37:43306
11 rows affected.
```

Out[6]:

TABLE_CATALOG	TABLE_SCHEMA	TABLE_NAME	COLUMN_NAME	ORDINAL_POSITION	CO
def	stu2000013094	tushare	ts_code	1	
def	stu2000013094	tushare	trade_date	2	
def	stu2000013094	tushare	open	3	
def	stu2000013094	tushare	high	4	
def	stu2000013094	tushare	low	5	
def	stu2000013094	tushare	close	6	
def	stu2000013094	tushare	pre_close	7	
def	stu2000013094	tushare	change	8	
def	stu2000013094	tushare	pct_chg	9	
def	stu2000013094	tushare	vol	10	
def	stu2000013094	tushare	amount	11	

In [8]:

```
%%sql
SELECT * FROM tushare ORDER BY trade_date LIMIT 15;
```

```
 * mysql://stu2000013094:***@162.105.146.37:43306
15 rows affected.
```

Out[8]:

ts_code	trade_date	open	high	low	close	pre_close	change	pct_chg	vol	
600019.SH	2011-01-04	6.44	6.56	6.31	6.55	6.39	0.16	2.5	572249.83	369
600019.SH	2011-01-05	6.49	6.5	6.42	6.45	6.55	-0.1	-1.53	334574.19	215
600019.SH	2011-01-06	6.46	6.5	6.39	6.46	6.45	0.01	0.16	248061.88	159
600019.SH	2011-01-07	6.46	6.68	6.41	6.56	6.46	0.1	1.55	562598.63	368
600019.SH	2011-01-10	6.54	6.64	6.51	6.52	6.56	-0.04	-0.61	422610.09	277
600019.SH	2011-01-11	6.52	6.72	6.51	6.7	6.52	0.18	2.76	608401.78	403
600019.SH	2011-01-12	6.72	6.9	6.67	6.83	6.7	0.13	1.94	815284.93	55
600019.SH	2011-01-13	6.87	6.9	6.73	6.77	6.83	-0.06	-0.88	511347.57	348
600019.SH	2011-01-14	6.77	6.77	6.61	6.71	6.77	-0.06	-0.89	415524.64	27
600019.SH	2011-01-17	6.64	6.74	6.48	6.53	6.71	-0.18	-2.68	524684.69	346
600019.SH	2011-01-18	6.51	6.56	6.45	6.53	6.53	0.0	0.0	292423.86	19
600019.SH	2011-01-19	6.55	6.6	6.48	6.59	6.53	0.06	0.92	294649.29	193
600019.SH	2011-01-20	6.58	6.58	6.38	6.4	6.59	-0.19	-2.88	377018.74	24
600019.SH	2011-01-21	6.39	6.67	6.36	6.56	6.4	0.16	2.5	457733.66	300
600019.SH	2011-01-24	6.57	6.75	6.54	6.63	6.56	0.07	1.07	505606.24	336

因子选取

接下来，我们将通过sql实现5个因子函数

In [7]:

```
import random
random.seed(2023)
print(random.randint(1,20), random.randint(21,40), random.randint(41,60), random.randint(61,80),
```

```
13 35 53 71 100
```

我们将选择离这五个随机整数序号最近的不带rank和ts_rank函数的因子
因此最终将实现的因子函数为：12 32 53 54 101
出于篇幅原因，每个因子我们只获取前15个进行展示

Alpha#12

sign(delta(volume, 1)*(-1*delta(close, 1)))

In [9]:

```
%%sql
SELECT trade_date,
    IF((vol - lag(vol, 1) OVER (ROWS 1 PRECEDING)) *
        ((-1) * (close - lag(close, 1) OVER (ROWS 1 PRECEDING))) > 0, 1,
       IF((vol - lag(vol, 1) OVER (ROWS 1 PRECEDING)) *
            ((-1) * (close - lag(close, 1) OVER (ROWS 1 PRECEDING))) < 0, -1, 0))
    AS Alpha12
FROM tushare LIMIT 15;
```

```
 * mysql://stu2000013094:***@162.105.146.37:43306
15 rows affected.
```

Out[9]:

trade_date	alpha12
2011-01-04	0
2011-01-05	-1
2011-01-06	1
2011-01-07	-1
2011-01-10	-1
2011-01-11	-1
2011-01-12	-1
2011-01-13	-1
2011-01-14	-1
2011-01-17	1
2011-01-18	0
2011-01-19	-1
2011-01-20	1
2011-01-21	-1
2011-01-24	-1

Alpha#32

scale(((sum(close, 7) / 7) - close)) + (20 * scale(correlation(vwap, delay(close, 5),230)))

In [10]:

```
%%sql
SELECT t.trade_date, t1.value / (SUM(t1.value) OVER ()) + (20 * t2.cor / (SUM(t2.cor) OVER ()))
FROM tushare t,
    (SELECT trade_date, SUM(close) OVER (ROWS 7 PRECEDING) AS value FROM tushare) t1,
    (SELECT tmp2.trade_date AS trade_date,
         tmp2.cov / (tmp2.stddev1 * tmp2.stddev2) AS cor
    FROM (SELECT tmp1.trade_date AS trade_date,
                SUM(tmp1.value1 * tmp1.value2) OVER (ROWS 230 PRECEDING) / (COUNT(tmp1.value1) OVER
                SUM(POWER(tmp1.value1, 2)) OVER (ROWS 230 PRECEDING) / (COUNT(tmp1.value1) OVER (ROWS
                SUM(POWER(tmp1.value2, 2)) OVER (ROWS 230 PRECEDING) / (COUNT(tmp1.value2) OVER (ROWS
        FROM (SELECT vwap.trade_date AS trade_date,
                    vwap.vwap - AVG(vwap.vwap) OVER (ROWS 230 PRECEDING) AS value1,
                    delay.delay - AVG(delay.delay) OVER (ROWS 230 PRECEDING) AS value2
                FROM (SELECT trade_date, (amount / vol) AS vwap FROM tushare) vwap,
                    (SELECT trade_date, lag(close, 5) OVER (ROWS 5 PRECEDING) AS delay FROM tusha
                WHERE vwap.trade_date = delay.trade_date) tmp1
         ) tmp2
    ) t2
WHERE t.trade_date = t1.trade_date AND t.trade_date = t2.trade_date
LIMIT 15
```

```
 * mysql://stu2000013094:***@162.105.146.37:43306
15 rows affected.
```

Out[10]:

trade_date	Alpha32
2011-01-04	None
2011-01-05	None
2011-01-06	None
2011-01-07	None
2011-01-10	None
2011-01-11	None
2011-01-12	-0.23429226496869168
2011-01-13	-0.34283028043769137
2011-01-14	-0.1888063364867501
2011-01-17	-0.2437311391550155
2011-01-18	-0.11712115490498508
2011-01-19	-0.06265831924348492
2011-01-20	-0.08119484880838819
2011-01-21	-0.08886135765032241
2011-01-24	-0.10143360547027164

Alpha#53

-1 * delta((((close - low) - (high - close)) / (close - low)), 9)

In [13]:

```
%%sql
SELECT trade_date,
    (-1) * (((close - low) - (high - close) / (close - low)) -
        lag((close - low) - (high - close) / (close - low), 9) OVER (ROWS 9 PRECEDING)) AS Alpha
FROM tushare
LIMIT 15
```

```
 * mysql://stu2000013094:***@162.105.146.37:43306
15 rows affected.
```

Out[13]:

trade_date	Alpha53
2011-01-04	None
2011-01-05	None
2011-01-06	None
2011-01-07	None
2011-01-10	None
2011-01-11	None
2011-01-12	None
2011-01-13	None
2011-01-14	None
2011-01-17	4.348333333333349
2011-01-18	-1.3416666666666552
2011-01-19	-0.5205194805194793
2011-01-20	8.329999999999773
2011-01-21	-11.640000000000262
2011-01-24	1.3280701754386028

Alpha#54

(-1 * ((low - close) * (open^5))) / ((low - high) * (close^5))

In [16]:

```
%%sql
SELECT trade_date,
    (-1) * (low - close) * POWER(open, 5) / ((low - high) * POWER(close, 5)) AS Alpha54
FROM tushare
LIMIT 15
```

```
 * mysql://stu2000013094:***@162.105.146.37:43306
15 rows affected.
```

Out[16]:

trade_date	Alpha54
2011-01-04	-0.8820517579847148
2011-01-05	-0.38677302625619203
2011-01-06	-0.636363636363637
2011-01-07	-0.5144828329641238
2011-01-10	-0.07811013900086033
2011-01-11	-0.7895838765437467
2011-01-12	-0.6414089173150279
2011-01-13	-0.2531928351303541
2011-01-14	-0.6534475890329252
2011-01-17	-0.20906009664641262
2011-01-18	-0.7162033338900877
2011-01-19	-0.8891824167473067
2011-01-20	-0.11487607755211249
2011-01-21	-0.5657875617879689
2011-01-24	-0.4095268873404357

Alpha#101

(close - open) / ((high - low) + .001)

In [18]:

```
%%sql
SELECT trade_date,
    (close - open) / (high - low + 0.001) AS Alpha101
FROM tushare
LIMIT 15
```

```
 * mysql://stu2000013094:***@162.105.146.37:43306
15 rows affected.
```

Out[18]:

trade_date	Alpha101
2011-01-04	0.43824701195218896
2011-01-05	-0.49382716049382713
2011-01-06	0.0
2011-01-07	0.36900369003689965
2011-01-10	-0.15267175572519448
2011-01-11	0.8530805687203822
2011-01-12	0.4761904761904767
2011-01-13	-0.5847953216374303
2011-01-14	-0.37267080745341546
2011-01-17	-0.4214559386973162
2011-01-18	0.18018018018018525
2011-01-19	0.33057851239669667
2011-01-20	-0.8955223880596993
2011-01-21	0.5466237942122191
2011-01-24	0.28436018957345793

练习2：JSON

In [25]:

```
import json

employees = []

with open('1.txt', 'r') as f:
    for line in f:
        d = {}
        list = line.strip("\n").split(",")
        d['name'] =list[0]
        d['salary']=list[1]
        family=list[2]
        fam=family.split(";")
        d['subordinates']=fam
        earn={}
        earn[list[3]]=list[4]
        earn[list[5]]=list[6]
        earn[list[7]]=list[8]
        d['deductions']=earn
        add={}
        add['street']=list[9]
        add['city']=list[10]
        add['state']=list[11]
        add['zip']=list[12]
        d['address']=add
        print(d)

with open('employees.json', 'w') as f:
    jas = json.dumps(d)
    f.write(jas)
```

```
-------------------------------------------------------------------------
----
FileNotFoundError                         Traceback (most recent call last)
<ipython-input-25-a1f65695b9a2> in <module>
      3 employees = []
      4
----> 5 with open('1.txt', 'r') as f:
      6     for line in f:
      7         d = {}

FileNotFoundError: [Errno 2] No such file or directory: '1.txt'
```

In [1]:

```
%load_ext sql
```

In [2]:

```
import pymysql
pymysql.install_as_MySQLdb()
%sql mysql://stu2000013095:stu2000013095@162.105.146.37:43306
```

Out[2]:

```
'Connected: stu2000013095@None'
```

In [3]:

```
%sql use stu2000013095;
```

```
 * mysql://stu2000013095:***@162.105.146.37:43306
0 rows affected.
```

Out[3]:

```
[]
```

In [148]:

```
%sql show processlist;
```

```
 * mysql://stu2000013095:***@162.105.146.37:43306
2 rows affected.
```

Out[148]:

Id	User	Host	db	Command	Time	State	Info
7600	stu2000013095	10.129.223.234:51194	stu2000013095	Sleep	1		None
7969	stu2000013095	10.129.223.234:63055	stu2000013095	Query	0	init	show processlist

In [149]:

```
%sql kill 7598
```

```
 * mysql://stu2000013095:***@162.105.146.37:43306
(pymysql.err.OperationalError) (1094, 'Unknown thread id: 7598')
[SQL: kill 7598]
(Background on this error at: http://sqlalche.me/e/13/e3q8) (http://sqlalche.me/
e/13/e3q8))
```

In [150]:

```
%%sql
drop table if exists employees;
set @@foreign_key_checks=0;
create table employees (
    id int primary key auto_increment,
    info json );

set @@foreign_key_checks=1;
```

```
 * mysql://stu2000013095:***@162.105.146.37:43306
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
```

Out[150]:

```
[]
```

In [151]:

```
%%sql
delete from employees;

insert into employees(info) values (
    '{"name"\:"John Doe","salary"\:100000.0, "subordinates" \:["Mary Smith", "Todd Jones"],"deduc
insert into employees(info) values (
    '{"name"\:"Mary Smith","salary"\:80000.0, "subordinates" \:["Bill King"],"deductions"\:{"Fede
insert into employees(info) values (
    '{"name"\:"Todd Jones","salary"\:70000.0, "deductions"\:{"Federal Taxes"\:"0.15","State Taxes
insert into employees(info) values (
    '{"name"\:"Bill King","salary"\:60000.0, "deductions"\:{"Federal Taxes"\:"0.15","State Taxes
insert into employees(info) values (
    '{"name"\:"Boss Man","salary"\:200000.0, "subordinates" \:["John Doe", "Fred Finance"],"deduc
insert into employees(info) values (
    '{"name"\:"Fred Finance","salary"\:150000.0, "subordinates" \:["Stacy Accountant"],"deductior
insert into employees(info) values (
    '{"name"\:"Stacy Accountant","salary"\:60000.0, "subordinates" \:["Mary Smith", "Todd Jones"]
```

```
 * mysql://stu2000013095:***@162.105.146.37:43306
0 rows affected.
1 rows affected.
1 rows affected.
1 rows affected.
1 rows affected.
1 rows affected.
1 rows affected.
1 rows affected.
```

Out[151]:

```
[]
```

操作2：

基于上面的employees表，定义5个的查询，使用JSON_EXTRACT或者JSON_SEARCH完成。

In [152]:

```
%%sql
desc employees;
select * from employees;
```

```
 * mysql://stu2000013095:***@162.105.146.37:43306
2 rows affected.
7 rows affected.
```

Out[152]:

id	info
1	{"name": "John Doe", "salary": 100000.0, "address": {"zip": "60601", "city": "Chicago", "state": "IL", "street": "1 Michigan Ave."}, "deductions": {"Insurance": "0.1", "State_Taxes": "0.05", "Federal_Taxes": "0.2"}, "subordinates": ["Mary Smith", "Todd Jones"]}
2	{"name": "Mary Smith", "salary": 80000.0, "address": {"zip": "60600", "city": "Chicago", "state": "IL", "street": "100 Ontario St."}, "deductions": {"Insurance": "0.1", "State Taxes ": "0.05", "Federal Taxes": "0.2"}, "subordinates": ["Bill King"]}
3	{"name": "Todd Jones", "salary": 70000.0, "address": {"zip": "60700", "city": "Oak Park", "state": "IL", "street": "200 Chicago Ave."}, "deductions": {"Insurance": "0.1", "State Taxes ": "0.03", "Federal Taxes": "0.15"}}
4	{"name": "Bill King", "salary": 60000.0, "address": {"zip": "60100", "city": "Obscuria", "state": "IL", "street": "300 Obscure Dr."}, "deductions": {"Insurance": "0.1", "State Taxes ": "0.03", "Federal Taxes": "0.15"}}
5	{"name": "Boss Man", "salary": 200000.0, "address": {"zip": "60500", "city": "Chicago", "state": "IL", "street": "Pretentious Drive."}, "deductions": {"Insurance": "0.05", "State Taxes ": "0.07", "Federal Taxes": "0.3"}, "subordinates": ["John Doe", "Fred Finance"]}
6	{"name": "Fred Finance", "salary": 150000.0, "address": {"zip": "60500", "city": "Chicago", "state": "IL", "street": "Pretentious Drive."}, "deductions": {"Insurance": "0.05", "State Taxes ": "0.07", "Federal Taxes": "0.3"}, "subordinates": ["Stacy Accountant"]}
7	{"name": "Stacy Accountant", "salary": 60000.0, "address": {"zip": "60563", "city": "Naperville", "state": "IL", "street": "300 Main St."}, "deductions": {"Insurance": "0.1", "State Taxes ": "0.03", "Federal Taxes": "0.15"}, "subordinates": ["Mary Smith", "Todd Jones"]}

In [153]:

```
%%sql
# 查询给定员工对应的地址编号
select json_extract(info,'$.address.zip') from employees where json_extract(info,'$.name') = 'Jo
```

```
 * mysql://stu2000013095:***@162.105.146.37:43306
1 rows affected.
```

Out[153]:

json_extract(info,'$.address.zip')
"60601"

In [154]:

```
%%sql
# 查询给定员工的实际收入（扣除税收等）
select json_extract(info,'$.salary')*(1-json_extract(info,'$.deductions.Insurance')-json_extract
```

```
 * mysql://stu2000013095:***@162.105.146.37:43306
1 rows affected.
```

Out[154]:

json_extract(info,'$. salary') * (1 - json_{e}xtract(info, ')$**.deductions.Insurance')-json_extract(info,'**$. deductions. State_{T}axes') - json_{e}xtract(info, ')$**.deductions.Federal_Taxes'))**
64999.99999999999

In [155]:

```
%%sql
# 查询Chicago对应的路径
select json_search(info,'all','Chicago') from employees
```

```
 * mysql://stu2000013095:***@162.105.146.37:43306
7 rows affected.
```

Out[155]:

json_search(info,'all','Chicago')
"$.address.city"
"$.address.city"
None
None
"$.address.city"
"$.address.city"
None

In [156]:

```
%%sql
# 查询给定员工的上级员工及其工资
select json_extract(info, '$.subordinates') as subordinates,
       json_extract(info, '$.salary') as subordinate_salary
from employees,
     json_table(info, '$.subordinates[*]' columns (
         subordinate VARCHAR(100) path '$'
     )) as sub
where sub.subordinate = 'John Doe';
```

```
 * mysql://stu2000013095:***@162.105.146.37:43306
1 rows affected.
```

Out[156]:

subordinates	subordinate_salary
["John Doe", "Fred Finance"]	200000.0

In [157]:

```
%%sql
# 查询和某位员工来自同一个城市的员工姓名
select json_extract(info, '$.name') as employee_name, json_extract(info, '$.address.city') as ci
from employees
where json_extract(info, '$.address.city') = (
    select json_extract(info, '$.address.city')
    from employees
    where json_extract(info, '$.name') = 'John Doe'
)
and json_extract(info, '$.name') != 'John Doe';
```

```
 * mysql://stu2000013095:***@162.105.146.37:43306
3 rows affected.
```

Out[157]:

employee_name	city
"Mary Smith"	"Chicago"
"Boss Man"	"Chicago"
"Fred Finance"	"Chicago"

操作3：将上面的employees表展开成1NF的平面表，新的平面表的属性为 (name, salary, subordinates, Federal Taxes, State Taxes, Insurance, street, city, state, zip)

In [158]:

```
%%sql
drop table if exists employees_flat;
create table employees_flat (
  name VARCHAR(100),
  salary DECIMAL(10, 2),
  subordinates JSON,
  Federal_Taxes DECIMAL(10, 2),
  State_Taxes DECIMAL(10, 2),
  Insurance DECIMAL(10, 2),
  street VARCHAR(100),
  city VARCHAR(100),
  state VARCHAR(100),
  zip VARCHAR(10)
);
```

```
 * mysql://stu2000013095:***@162.105.146.37:43306
0 rows affected.
0 rows affected.
```

Out[158]:

```
[]
```

In [159]:

```
%%sql
insert into employees_flat (name, salary, subordinates, Federal_Taxes, State_Taxes, Insurance, st
select
  json_extract(info, '$.name'),
  json_extract(info, '$.salary'),
  json_extract(info, '$.subordinates'),
  json_extract(info, '$.deductions.Federal_Taxes'),
  json_extract(info, '$.deductions.State_Taxes'),
  json_extract(info, '$.deductions.Insurance'),
  json_extract(info, '$.address.street'),
  json_extract(info, '$.address.city'),
  json_extract(info, '$.address.state'),
  json_extract(info, '$.address.zip')
from employees;
```

```
 * mysql://stu2000013095:***@162.105.146.37:43306
7 rows affected.
```

Out[159]:

```
[]
```

In [160]:

```
%%sql
desc employees_flat;
select * from employees_flat;
```

```
 * mysql://stu2000013095:***@162.105.146.37:43306
10 rows affected.
7 rows affected.
```

Out[160]:

name	salary	subordinates	Federal_Taxes	State_Taxes	Insurance	street	
"John Doe"	100000.00	["Mary Smith", "Todd Jones"]	0.20	0.05	0.10	"1 Michigan Ave."	"
"Mary Smith"	80000.00	["Bill King"]	None	None	0.10	"100 Ontario St."	"
"Todd Jones"	70000.00	None	None	None	0.10	"200 Chicago Ave."	"C
"Bill King"	60000.00	None	None	None	0.10	"300 Obscure Dr."	"C
"Boss Man"	200000.00	["John Doe", "Fred Finance"]	None	None	0.05	"Pretentious Drive."	"
"Fred Finance"	150000.00	["Stacy Accountant"]	None	None	0.05	"Pretentious Drive."	"
"Stacy Accountant"	60000.00	["Mary Smith", "Todd Jones"]	None	None	0.10	"300 Main St."	"Na

操作4：对下面的Assembly表，使用递归查询，生成树形结构，再将它输出如下的JSON数据：[trike[wheel[spoke, tire[rim, tube]], frame[seat, pedal]]]

In [5]:

```
%%sql
drop table if exists assembly;
create table assembly(
id  int not null primary key auto_increment,
part char(20),
subpart char(20));
insert into assembly values
(1,'trike','wheel'),
(2,'trike','frame'),
(3,'frame','seat'),
(4,'frame','pedal'),
(5,'wheel','spoke'),
(6,'wheel','tire'),
(7,'tire','rim'),
(8,'tire','tube');
select * from assembly;
```

```
 * mysql://stu2000013095:***@162.105.146.37:43306
0 rows affected.
0 rows affected.
8 rows affected.
8 rows affected.
```

Out[5]:

id	part	subpart
1	trike	wheel
2	trike	frame
3	frame	seat
4	frame	pedal
5	wheel	spoke
6	wheel	tire
7	tire	rim
8	tire	tube

In [6]:

```
%%sql
desc assembly;
select * from assembly;
```

```
 * mysql://stu2000013095:***@162.105.146.37:43306
3 rows affected.
8 rows affected.
```

Out[6]:

id	part	subpart
1	trike	wheel
2	trike	frame
3	frame	seat
4	frame	pedal
5	wheel	spoke
6	wheel	tire
7	tire	rim
8	tire	tube

展示某个节点对应的子节点，其中“.”个数表明之间的节点个数

In [9]:

```
%%sql
with recursive assembly_tree AS (
  select part, subpart, 1 AS level, concat(part, '.', subpart) as path
  from assembly
  where part = 'trike'
  union all
  select a.part, a.subpart, at.level + 1, concat(at.path, '.', a.subpart)
  from assembly a
  join assembly_tree at on a.part = at.subpart
)
select concat(
  repeat('.', at.level - 1),
  substring_index(at.path, '.', -1)
) as tree_structure
from assembly_tree at
order by at.path;
```

```
 * mysql://stu2000013095:***@162.105.146.37:43306
8 rows affected.
```

Out[9]:

tree_structure
frame
.pedal
.seat
wheel
.spoke
.tire
..rim
..tube

In [10]:

```
%%sql
with recursive assembly_tree AS (
  select part, subpart, 1 AS level, concat(part, ’.’, subpart) as path
  from assembly
  where part = ’trike’
  union all
  select a.part, a.subpart, at.level + 1, concat(at.path, ’.’, a.subpart)
  from assembly a
  join assembly_tree at on a.part = at.subpart
)
select * from assembly_tree
```

```
 * mysql://stu2000013095:***@162.105.146.37:43306
8 rows affected.
```

Out[10]:

part	subpart	level	path
trike	wheel	1	trike.wheel
trike	frame	1	trike.frame
frame	seat	2	trike.frame.seat
frame	pedal	2	trike.frame.pedal
wheel	spoke	2	trike.wheel.spoke
wheel	tire	2	trike.wheel.tire
tire	rim	3	trike.wheel.tire.rim
tire	tube	3	trike.wheel.tire.tube

In [8]:

```
%%sql
with recursive assembly_cte (part, subpart, level) as (
  select part, subpart, 1
  from assembly
  where part = 'trike'
  union all
  select a.part, a.subpart, cte.level + 1
  from assembly a
  inner join assembly_cte cte on a.part = cte.subpart
)
select
    json_object(
        main.part,
       (
            select json_arrayagg(
                json_object(
                    sub.subpart,
                    (
                        select json_arrayagg(
                            json_object(
                                subsub.subpart, (
                                    select json_arrayagg(subsubsub.subpart)
                                    from assembly_cte subsubsub
                                    where subsubsub.part=subsub.subpart
                                )
                            )

                        )
                        from assembly_cte subsub
                        where subsub.part=sub.subpart
                    )
                ))
            from assembly_cte sub
            where sub.part = main.part
        )
    )
as json_output
from assembly_cte main WHERE main.level = 1
order by main.level
limit 1
```

```
 * mysql://stu2000013095:***@162.105.146.37:43306
1 rows affected.
```

Out[8]:

json_output
{"trike": [{"wheel": [{"spoke": null}, {"tire": ["rim", "tube"]}]}, {"frame": [{"seat": null}, {"pedal": null}]}]}

In [ ]: