

## 7.2 用Pandas模块读写常见格式文件

- Python的模块函数分三个层次：
  - 一.内置函数
    - 不用import语句引入，它里面的函数可直接调用。
  - 二.标准模块函数
    - 用import语句引入后再调用，但不必安装。如math库。
  - 三.第三方模块函数
    - 需先安装，再用import语句引入模块后才能调用里面的函数，如Pandas模块。

# 第三方库安装

- ◎ 打开网页：<https://pypi.org/>
- ◎ 输入模块名，就可查到模块的详细说明



# pip命令

- c:\>pip install pandas 安装pandas模块

```
pip <command> [options]
```

Commands:

install	Install packages.
download	Download packages.
uninstall	Uninstall packages.
freeze	Output installed packages in requirements format.
list	List installed packages.
show	Show information about installed packages.
config	Manage local and global configuration.
search	Search PyPI for packages.
wheel	Build wheels from your requirements.
hash	Compute hashes of package archives.
completion	A helper command used for command completion.
help	Show help for commands.

# Pandas模块

- ◎ Pandas是python的一个数据分析包，
- ◎ Pandas最初被作为金融数据分析工具而开发出来，Pandas为时间序列分析提供了很好的支持。
- ◎ Pandas是基于NumPy 的一种工具。
- ◎ Pandas 纳入了大量函数和一些标准的数据模型，提供了高效操作大型数据集所需的工具，提供了大量能使我们快速便捷地处理数据的函数和方法，让Python成为强大而高效的数据分析环境。

# Plotly模块

- Plotly是一个基于JavaScript的动态绘图模块。Plotly的绘图效果与我们在网页上看到的动态交互式绘图结果是一样的，其默认的绘图结果是一个HTML网页文件，通过浏览器就可以查看。
- Plotly有着强大又丰富的绘图库，支持各种类型的绘图方案。
- Plotly是基于JavaScript的绘图库，所以其绘图结果可以与web应用无缝集成。
- Plotly最初是一款商业化的绘图软件，自plotly.js开源之后，我们可以使用本地的离线模式进行绘图，不依赖于官方的服务器，使得绘图速度更快，而效果与在线绘图一样。
- 离线模式： `from plotly.offline import plot`

# DataFrame数据类型

DataFrame是Pandas库的一种数据类型。DataFrame是一个行和列都具有标签的表格，它与Excel电子表格并无不同。

DataFrame使用非常方便，当你在处理二维表格数据时，都应该使用它们。DataFrame可由元组、列表、字典或另一个DataFrame构造出来

# 用列表产生DataFrame变量data

- from plotly.offline import plot
- from plotly import figure\_factory as FF
- import pandas as pd
- ```
data=pd.DataFrame([["2050921018","詹延峰","计算数学",65,85,76], ["2050921036","李 小鹏","金融学类",86,95,85], ["2050921039","裴凡法","经济学类",86,95,65], ["2040912116","茅舒瑶","社会保障",90,95,100], ["2050912017","陈 见影","化学 工程",62,75,92], ["2050912064","梅 钦钦","材料科学",87,95,80], ["2050109153","王影平","大 气科学",86,89,72], ["2050151003","韩平医","化学 工程",82,99,60]], columns= ("学号","姓名","专业","笔试","平时","实验"))
```
- table = FF.create\_table(data) #用plotly产生输出表格
- plot(table, show\_link=False)

| 学号         | 姓名  | 专业   | 笔试 | 平时 | 实验  |
|------------|-----|------|----|----|-----|
| 2050921018 | 詹延峰 | 计算数学 | 65 | 85 | 76  |
| 2050921036 | 李小鹏 | 金融学类 | 86 | 95 | 85  |
| 2050921039 | 裴凡法 | 经济学类 | 86 | 95 | 65  |
| 2040912116 | 茅舒瑶 | 社会保障 | 90 | 95 | 100 |
| 2050912017 | 陈见影 | 化学工程 | 62 | 75 | 92  |
| 2050912064 | 梅钦钦 | 材料科学 | 87 | 95 | 80  |
| 2050109153 | 王影平 | 大气科学 | 86 | 89 | 72  |
| 2050151003 | 韩平医 | 化学工程 | 82 | 99 | 60  |

# 用DataFrame计算总评分

- from plotly.offline import plot
- from plotly import figure\_factory as FF
- import pandas as pd
- data=pd.DataFrame([["2050921018","詹延峰","计算数学",65,85,76], ["2050921036","李 小鹏","金融学类",86,95,85], ["2050921039","裴凡法","经济学类",86,95,65], ["2040912116","茅舒瑶","社会保障",90,95,100], ["2050912017","陈 见影","化学 工程",62,75,92], ["2050912064","梅 钦钦","材料科学",87,95,80], ["2050109153","王影平","大 气科学",86,89,72], ["2050151003","韩平医","化学 工程",82,99,60]], columns= ("学号","姓名","专业","笔试","平时","实验"))
- #data变量中增加一列
- data["总评成绩"]=data["笔试"]\*0.5+data["平时"]\*0.25+data["实验"]\*0.25
- table = FF.create\_table(data)
- plot(table, show\_link=False)

| 学号         | 姓名  | 专业   | 笔试 | 平时 | 实验  | 总评成绩  |
|------------|-----|------|----|----|-----|-------|
| 2050921018 | 詹延峰 | 计算数学 | 65 | 85 | 76  | 72.75 |
| 2050921036 | 李小鹏 | 金融学类 | 86 | 95 | 85  | 88.0  |
| 2050921039 | 裴凡法 | 经济学类 | 86 | 95 | 65  | 83.0  |
| 2040912116 | 茅舒瑶 | 社会保障 | 90 | 95 | 100 | 93.75 |
| 2050912017 | 陈见影 | 化学工程 | 62 | 75 | 92  | 72.75 |
| 2050912064 | 梅钦钦 | 材料科学 | 87 | 95 | 80  | 87.25 |
| 2050109153 | 王影平 | 大气科学 | 86 | 89 | 72  | 83.25 |
| 2050151003 | 韩平医 | 化学工程 | 82 | 99 | 60  | 80.75 |

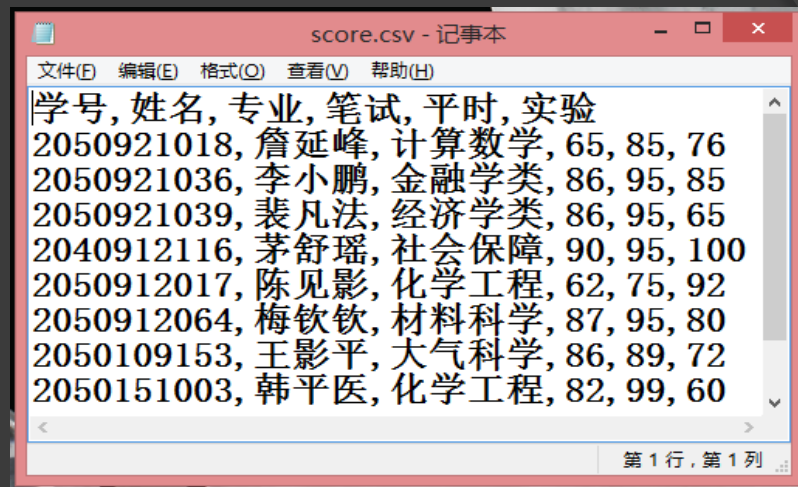


# 用pandas读写各种类型文件

|              |          |            |
|--------------|----------|------------|
| ⦿ read_csv   | to_csv   | 读写csv 文件   |
| ⦿ read_excel | to_excel | 读写excel 文件 |
| ⦿ read_json  | to_json  | 读写json 文件  |

# 读取CSV文件

- from plotly.offline import plot
- from plotly import figure\_factory as FF
- import pandas as pd



| 学号         | 姓名  | 专业   | 笔试 | 平时 | 实验  |
|------------|-----|------|----|----|-----|
| 2050921018 | 詹延峰 | 计算数学 | 65 | 85 | 76  |
| 2050921036 | 李小鹏 | 金融学类 | 86 | 95 | 85  |
| 2050921039 | 裴凡法 | 经济学类 | 86 | 95 | 65  |
| 2040912116 | 茅舒瑶 | 社会保障 | 90 | 95 | 100 |
| 2050912017 | 陈见影 | 化学工程 | 62 | 75 | 92  |
| 2050912064 | 梅钦钦 | 材料科学 | 87 | 95 | 80  |
| 2050109153 | 王影平 | 大气科学 | 86 | 89 | 72  |
| 2050151003 | 韩平医 | 化学工程 | 82 | 99 | 60  |

- data = pd.read\_csv("score.csv",encoding="GBK")
- table = FF.create\_table(data) #产生表格
- plot(table, show\_link=False)
- “GB2312”、“GBK”和“CP936”都是用两个字节表示中文的编码。“GB2312”是国标码，“GBK”是“GB2312”的扩展，
- “CP936”是在“GB2312”基础上开发的汉字编码。

# 写网页文件

## ◎产生网页文件'score.html'

- ◎ `from plotly.offline import plot`
- ◎ `from plotly import figure_factory as FF`
- ◎ `import pandas as pd`
  
- ◎ `data = pd.read_csv("score.csv",encoding="GBK")`
- ◎ `table = FF.create_table(data) #产生表格`
- ◎ `plot(table, filename='score.html',show_link=False)`

# 读写Excel 文件

- 要装xlrd模块
- 读score.xlsx文件
- 写入scoregp.xlsx文件
- import pandas as pd
- data = pd.read\_excel("score.xlsx")
- data["总评"]=data["笔试"]\*0.5+data["平时"]\*0.25+data["实验"]\*0.25
- data.to\_excel("scoregp.xlsx",index=0)

|   | A          | B   | C    | D  | E  | F   |
|---|------------|-----|------|----|----|-----|
| 1 | 学号         | 姓名  | 专业   | 笔试 | 平时 | 实验  |
| 2 | 2050921018 | 詹延峰 | 计算数学 | 65 | 85 | 76  |
| 3 | 2050921036 | 李小鹏 | 金融学类 | 86 | 95 | 85  |
| 4 | 2050921039 | 裴凡法 | 经济学类 | 86 | 95 | 65  |
| 5 | 2040912116 | 茅舒瑶 | 社会保障 | 90 | 95 | 100 |
| 6 | 2050912017 | 陈见影 | 化学工程 | 62 | 75 | 92  |
| 7 | 2050912064 | 梅钦钦 | 材料科学 | 87 | 95 | 80  |
| 8 | 2050109153 | 王影平 | 大气科学 | 86 | 89 | 72  |
| 9 | 2050151003 | 韩平医 | 化学工程 | 82 | 99 | 60  |

# JSON 文件读写

- ◎ JSON文件主要有两种结构：
- ◎ “键/值”对的集合：不同的语言中，它被理解为对象，纪录，结构，字典，哈希表，有键列表，或者关联数组。Python中对应字典类型。
- ◎ 值的有序列表：在大部分语言中，它被理解为数组。Python中对应列表类型。

# to\_json和read\_json

- 读取Excel 文件"score.xlsx", 用to\_json函数产生JSON 文件data.json, 最后用read\_json读JSON 文件"data.json"
- import pandas as pd
- data = pd.read\_excel("score.xlsx")
- # force\_ascii=False 有中文时不要用ASCII编码
- data.to\_json("data.json",force\_ascii=False)
- jsontdata=pd.read\_json("data.json")
- print(jsontdata)

# 数据库文件读写

- example.db 是sqlite数据库 sqlalchemy模块
- from sqlalchemy import create\_engine
- import pandas as pd
- from plotly import figure\_factory as FF
- from plotly.offline import plot
- engine = create\_engine('sqlite:///example.db')
- data=pd.read\_sql('select 图书书目表.id,类别名,书名,作者 \
- from 图书书目表,类别表 \
- where 类别表.id=图书书目表.类别',engine)
- table = FF.create\_table(data) #用plotly产生输出表格
- plot(table, show\_link=False)

| 类别 | id | 书名       | 作者      | 价格      |
|----|----|----------|---------|---------|
| 过滤 | 过滤 | 过滤       | 过滤      | 过滤      |
| G  | 1  | 高中物理典型…  | 余建丽     | ¥ 19.80 |
| G  | 2  | 高中语文基础…  | 李洪达     | ¥ 14.90 |
| G  | 3  | 英语学习技法…  | 夏惠敏     | ¥ 19.80 |
| I  | 4  | 忠诚卫士     | 贺梦凡、邓原  | ¥ 22.80 |
| I  | 5  | 破晓       | 冀夫      | ¥ 19.00 |
| I  | 6  | 皖南事变     | 黎汝清     | ¥ 32.00 |
| R  | 7  | 健康要素     | 江国生     | ¥ 16.00 |
| R  | 8  | 病理学      | 朗志峰     | ¥ 30.00 |
| R  | 9  | 家庭用药手册   | 张晓友、董淑华 | ¥ 40.00 |
| TP | 10 | 数据库应用技…  | 黄志球、李清  | ¥ 27.00 |
| TP | 11 | 数据库系统及…  | 崔巍      | ¥ 28.40 |
| R  | 12 | 健康快乐100岁 | 洪昭光     | ¥ 8.00  |
| I  | 13 | 围城       | 钱钟书     | ¥ 15.20 |

| id | 类别名     |
|----|---------|
| 过滤 | 过滤      |
| G  | 教辅      |
| I  | 文学      |
| R  | 医药健康    |
| TP | 计算机技术与… |

# 程序运行结果

| id | 类别名      | 书名           | 作者      |
|----|----------|--------------|---------|
| 1  | 教辅       | 高中物理典型错误诊疗大全 | 余建丽     |
| 2  | 教辅       | 高中语文基础知识大全   | 李洪达     |
| 3  | 教辅       | 英语学习技法大全     | 夏惠敏     |
| 4  | 文学       | 忠诚卫士         | 贺梦凡、邓原  |
| 5  | 文学       | 破晓           | 冀夫      |
| 6  | 文学       | 皖南事变         | 黎汝清     |
| 7  | 医药健康     | 健康要素         | 江国生     |
| 8  | 医药健康     | 病理学          | 朗志峰     |
| 9  | 医药健康     | 家庭用药手册       | 张晓友、董淑华 |
| 10 | 计算机技术与科学 | 数据库应用技术基础    | 黄志球、李清  |
| 11 | 计算机技术与科学 | 数据库系统及应用     | 崔巍      |
| 12 | 医药健康     | 健康快乐100岁     | 洪昭光     |
| 13 | 文学       | 围城           | 钱钟书     |



# 访问DataFrame元素

- import pandas as pd
- data=pd.DataFrame([["China",2000,3700,70.0],
  - ["China",2001,3980,70.3],
  - ["China",2002,4320,70.7]],
  - columns=("country","year","ppp-gdp","life-exp"))
- print(data)
- print()
- print(data.iloc[0,2]) #第一行第三列，取单个元素
- print()
- print(data.iloc[1:3,2:4]) #第二，三行，第三，四列，取多个元素
- print()
- print([name for name in data]) #由列名组成的列表

- country year ppp-gdp life-exp
- 0 China 2000 3700 70.0
- 1 China 2001 3980 70.3
- 2 China 2002 4320 70.7
- 3700
- ppp-gdp life-exp
- 1 3980 70.3
- 2 4320 70.7
- ['country', 'year', 'ppp-gdp', 'life-exp']

# 用Pandas作数据整理

- import pandas as pd
- data=pd.read\_excel("gdp.xlsx")
- columnname=[name for name in data] #由列名组成的列表
- #创建空的dataframe
- result=pd.DataFrame(columns=("country","year","ppp-gdp"))
- for i in range(len(columnname)-1):
  - # result.loc[i]表示第i行
  - result.loc[i]=[data.iloc[0,0],columnname[i+1],data.iloc[0,i+1]]
- print(result)
- pd.to\_excel("ppp-gdp.xlsx")

|   | A       | B    | C    | D    | E    | F    | G    | H    |
|---|---------|------|------|------|------|------|------|------|
| 1 | country | 2000 | 2001 | 2002 | 2003 | 2004 | 2005 | 2006 |
| 2 | China   | 3700 | 3980 | 4320 | 4720 | 5170 | 5720 | 6410 |

| J       | K    | L       |
|---------|------|---------|
| country | year | ppp-gdp |
| China   | 2000 | 3700    |
| China   | 2001 | 3980    |
| China   | 2002 | 4320    |
| China   | 2003 | 4720    |
| China   | 2004 | 5170    |
| China   | 2005 | 5720    |
| China   | 2006 | 6410    |

# 表格拼接

- import pandas as pd
- data=pd.read\_excel("ppp-gdp.xlsx")
- print(data)
- data1=pd.read\_excel("life-exp.xlsx")
- print(data1)
- data2=data1[["year","life-exp"]] #取两列
- data3=pd.merge(data,data2,on="year",how='left') #合并re
- print(data3)

|   | country | year | ppp-gdp |
|---|---------|------|---------|
| 0 | China   | 2000 | 3700    |
| 1 | China   | 2001 | 3980    |
| 2 | China   | 2002 | 4320    |
| 3 | China   | 2003 | 4720    |
| 4 | China   | 2004 | 5170    |
| 5 | China   | 2005 | 5720    |
| 6 | China   | 2006 | 6410    |

|   | country | year | ppp-gdp | life-exp |
|---|---------|------|---------|----------|
| 0 | China   | 2000 | 3700    | 70.0     |
| 1 | China   | 2001 | 3980    | 70.3     |
| 2 | China   | 2002 | 4320    | 70.7     |
| 3 | China   | 2003 | 4720    | NaN      |
| 4 | China   | 2004 | 5170    | 71.9     |
| 5 | China   | 2005 | 5720    | 72.4     |
| 6 | China   | 2006 | 6410    | 73.0     |

|   | country | year | life-exp |
|---|---------|------|----------|
| 0 | China   | 2006 | 73.0     |
| 1 | China   | 2001 | 70.3     |
| 2 | China   | 2002 | 70.7     |
| 3 | China   | 2004 | 71.9     |
| 4 | China   | 2005 | 72.4     |
| 5 | China   | 2000 | 70.0     |

# Excel电子表格合并



income



life-gdp



life

# Excel电子表格合并程序

- import pandas as pd
- startyear=int(input())
- #人均期望寿命
- dataset=pd.read\_excel("life.xlsx")
- a=pd.DataFrame(columns=("country","life-exp","year"))
- for i in range(startyear,2019):
  - b=dataset[['country',i]].copy()
  - b['year']=i
  - b.columns=['country','life-exp','year']
  - c=a.append(b)
  - a=c
- #人均收入，以PPP计算
- dataset=pd.read\_excel("income.xlsx")
- x=pd.DataFrame(columns=("country","income","year"))
- for i in range(startyear,2019):
  - y=dataset[['country',i]].copy()
  - y['year']=i
  - y.columns=['country','income','year']
  - z=x.append(y)
  - x=z
- data=pd.merge(a,x)
- data.to\_excel(f"{startyear}到2018人均GDP和人均寿命.xlsx")