pd.read_csv(
header=作列名的行号,
names=[作为结果的列名列表],
index_col=[指定列名列表作为索引],
sep=',',
skiprows=[跳过的行列表],
na_values=[哪些值被视为NaN，可使用字典为每列分别指定],
usecols=[读取的列列表],
skipfooter=忽略的尾部行数,
nrows=读取的行数,
parse_dates=[1, 2, 3]单独解析1、2、3列/[[1, 3]]把1、3列结合起来解析,
chunksize=分块读取的大小)

参数

参数	说明
path	表示文件系统位置、URL、文件型对象的字符串
sep或delimiter	用于对行中各字段进行拆分的字符序列或正则表达式
header	用作列名的行号。默认为0（第一行），如果没有header行就应该设置为None
index_col	用作行索引的列编号或列名。可以是单个名称/数字或由多个名称/数字组成的列表（层次化索引）
names	用于结果的列名列表，结合header=None
skiprows	需要忽略的行数（从文件开始处算起），或需要跳过的行号列表（从0开始）
na_values	一组用于替换NA的值
comment	用于将注释信息从行尾拆分出去的字符（一个或多个）
parse_dates	尝试将数据解析为日期，默认为False。如果为True，则尝试解析所有列。此外，还可以指定需要解析的一组列号或列名。如果列表的元素为列表或元组，就会将多个列组合到一起再进行日期解析工作（例如，日期/时间分别位于两个列中）
keep_date_col	如果连接多列解析日期，则保持参与连接的列。默认为False。
converters	由列号/列名跟函数之间的映射关系组成的字典。例如，{'foo': f}会对foo列的所有值应用函数f
dayfirst	当解析有歧义的日期时，将其看做国际格式（例如，7/6/2012 → June 7, 2012）。默认为False
date_parser	用于解析日期的函数
nrows	需要读取的行数（从文件开始处算起）
iterator	返回一个TextParser以便逐块读取文件
chunksize	文件块的大小（用于迭代）
skip_footer	需要忽略的行数（从文件末尾处算起）

表6-2：read_csv/read_table函数的参数（续）

参数	说明
verbose	打印各种解析器输出信息，比如“非数值列中缺失值的数量”等
encoding	用于unicode的文本编码格式。例如，“utf-8”表示用UTF-8编码的文本
squeeze	如果数据经解析后仅含一列，则返回Series
thousands	千分位分隔符，如“,”或“.”

- 06章、文件
  - 6.1 文本格式读写
    - 6.1.1 分块读取
      - 调整显示 → pd.options.display.max_rows = 值
      - chunksize=
        - 返回TextFileReader对象，可遍历
        - chunker = pd.read_csv(chunksize=1000)
for piece in chunker:
.....
    - 6.1.2 写入数据
      - df.to_csv(文件名,
sep=,
na_rep=指定缺失值写入时用什么替代,
index=False， header=False,
columns=[列列表，按顺序只写入这些列的数据])
    - 6.1.3 使用分隔格式
      - 使用内建csv模块

```
import csv
with open('file') as f:
    lines = list(csv.reader(f))
    header, values = lines[0], lines[1:]
    data_dict = {h: v for h, v in zip(header, zip(*values))}
```

    - 6.1.4 JSON数据
      - json转python → json.loads()
      - python转json → json.dumps()
      - pd.read_json() → 默认选项假设JSON数组中的每个对象是表格中的一行
      - s/df.to_json()
    - 6.1.5 HTML数据
      - pd.read_html() → 自动将HTML文件中的<table>标签内的的表格解析为DataFrame对象
  - 6.2 二进制格式
    - df.to_pickle()
    - pd.read_pickle()
    - 6.2.1 HDF5格式
    - 6.2.2 Excel文件
      - 读取 → pd.read_excel(文件, sheet_name=)
      - 写入 → writer = pd.ExcelWriter(文件)
df.to_excel(writer, sheet_name)
  - 6.3 与web交互 → requests库
  - 6.4 与数据库交互
    - cur.executemany(sql, df.values.tolist())
    - 查询表结构 → columns = [x[0] for x in cursor.description]