In [ ]:

```
df2.to_excel('temp.xlsx', index = False, sheet_name = 'data')
```

df.to_stata()

## 2.2 保存数据至数据库

df.to_sql(

```
    name : 将要存储数据的表名称
    con : SQLAlchemy引擎/DBAPI2连接引擎名称
    if_exists = 'fail' : 指定表已经存在时的处理方式
        fail : 不做任何处理（不插入新数据）
        replace : 删除原表并重建新表
        append : 在原表后插入新数据
    index = True : 是否导出索引
```

)

In [ ]:

```
'''
apptbl.to_sql(name="jt_histrec", con=eng,
              if_exists='append', index=False)
'''
```

## 2.3 实战：保存北京PM2.5数据为数据文件

要求：尝试将PM2.5数据保存为csv、EXCEL等格式，并使用各种不同的参数设置。

# 3 变量列的基本操作

## 3.1 对数据作简单浏览

In [ ]:

```
print(df2)
```

In [ ]:

```
# 数据框的基本信息
df2.info()
```

In [ ]:

```
# 浏览前几条记录
df2.head(10)
```

In [ ]:

```
# 浏览最后几条记录
df2.tail()
```

## 3.2 重命名变量列

直接修改columns属性

```
df.columns = 新的名称list
```

In [ ]:

```
# 给出变量名/列名
df2.columns
```

In [ ]:

```
df2.columns = ['名次2', '学校名称2', '总分', '类型',
               '所在省份', '所在城市', '办学方向', '主管部门']
df2
```

只修改指定列名

df.rename(

```
columns = 新旧名称字典 : {'旧名称': '新名称'}
inplace = False : 是否直接替换原数据框
```

)

In [ ]:

```
df2.rename(
    columns = {'名次2': '名次', '学校名称2':'学校名称'}, inplace = True
)
df2
```

## 3.3 筛选变量列

df.var

```
只适用于已存在的列
只能筛选单列，结果为Series
```

In [ ]:

```
df2.名次
```

df['var']

```
单列的筛选结果为Series，如果希望为df，需使用列表形式
```

In [ ]:

```
df2[  ['名次']   ]
```

df[['var1', 'var2']]

多列时，列名需要用列表形式提供（因此可使用列表中的切片操作）
多列的筛选结果为DF

In [ ]:

```
df2[['名次','总分']]
```

# 3.4 删除变量列

df.drop(

index / columns = 准备删除的行/列标签，多个时用列表形式提供
inplace = False : 是否直接更改原数据框

) # pandas0.21版本以上

In [ ]:

```
df2.drop(columns = ['名次', '所在城市'])
```

df.drop(

labels : 准备删除的行/列标签，多个时用列表形式提供
axis : 指定需要操作的行/列序号（0/1）/名称（'index'/'columns'）
level = None : 存在多重索引时，指定具体操作的索引序号/标签

)

In [ ]:

```
df2.drop(['名次', '所在城市'], axis = 1)
```

del df['column-name']

直接删除原数据框相应的一列，建议尽量少用

# 3.5 变量类型的转换

## 3.5.1 Pandas支持的数据类型

具体类型是Python，Numpy各种类型的混合，可以比下表分的更细

```
float
int
string
bool
datetime64[ns], datetime64[ns, tz], timedelta[ns]
category
object
```

df.dtypes : 查看各列的数据类型

In [ ]:

```
df2.dtypes
```

### 3.5.2 在不同数据类型间转换

df.astype(

```
dtype : 指定希望转换的数据类型
    可以使用numpy或者python中的数据类型: int/float/bool/str
copy = True : 是否生成新的副本，而不是替换原数据框
errors = 'raise' : 转换出错时是否抛出错误，'raise'/'ignore'
```

)

In [ ]:

```
df2.astype('str').dtypes
```

In [ ]:

```
df2.名次.astype('str')
```

In [ ]:

```
df2.astype('int', errors = 'ignore').dtypes
```

明确指定转换类型的函数:

```
pd.to_datetime()
pd.to_timedelta()
pd.to_numeric()
df.to_string()
可以配合df.apply来批量进行多列的转换
```

In [ ]:

```
pd.to_numeric(df2.总分)
```

In [ ]:

```
df2[['名次', '总分']].astype('str').apply(pd.to_numeric).dtypes
```

df.infer_objects()

基于数据特征进行自动转换
0.21版以后新增

In [ ]:

```
df = pd.DataFrame({"A": ["a", 1, 2, 3]})
df = df.iloc[1:]
df.dtypes
```

In [ ]:

```
df.infer_objects().dtypes
```

## 3.6 实战：对PM2.5数据做简单清理

要求:

在数据中删除对后续分析无用的Parameter、Duration、QC Name等变量列
尝试对Date (LST)、Value等变量进行重命名
尝试对数据做各种类型的转换

# 4 胖哒黑魔法：索引

索引的用途:

用于在分析、可视化、数据展示、数据操作中标记数据行
提供数据汇总、合并、筛选时的关键依据
提供数据重构时的关键依据

注意事项:

索引是不可直接修改的，只能增、删、替换
逻辑上索引不应当出现重复值，Pandas对这种情况不会报错，但显然有潜在风险

## 4.1 建立索引

### 4.1.1 新建数据框时建立索引

所有的数据框默认都已经拥有流水号格式的索引，因此这里的“建立”索引指的是自定义索引

Q：课程学习遇到不懂怎么办？

- ✧ 本课程提供额外福利：QQ 群供学员交流心得，群号：630030855，可直接扫下方的二维码进入。
- ✧ 老师有空时也会参与讨论，但请不要把你的工作问题直接让老师解决。
- ✧ 老师鼓励学员多思考、多动手，通过自己努力解决问题，这样能发现乐趣、有成就感、成长快。