CDA 0329期

阶段交流II

第七组

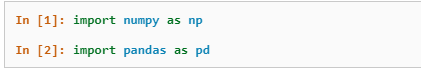
刘姝辰 韦青山 张文博

# Pandas简介

如果把Python比作是我们生活中经常使用的通信软件微信的话, 那么pandas可以看作是微信中的一个小程序. 微信中的小程序有各种各样的分类, 比如运动类, 购物类等. Pandas就是Python中一款用于专业数据分析的小程序. 在Python中称作包. Pandas是基于NumPy的专业数据分析工具，可以灵活高效的处理各种数据集，它提供了两种类型的数据结构，分别是DataFrame和Series，我们可以把DataFrame理解为Excel里面的一张表，而Series就是表中的某一列. Pandas和Excel、SQL相比，只是调用和处理数据的方式变了，核心都是对数据进行一系列的处理，在正式处理之前，更重要的是谋定而后动，明确分析的意义，理清分析思路之后再处理和分析数据，往往事半功倍。

# 创建、读入、存储数据

首先，在进行数据分析之前， 我们需要导入相关的包。

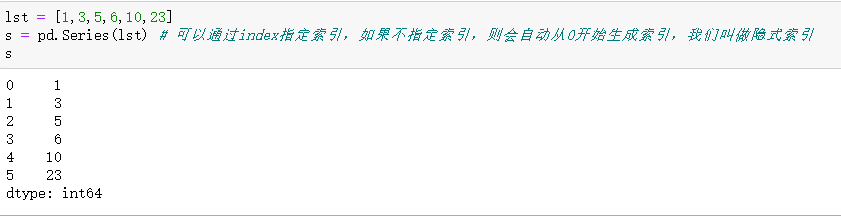


创建数据

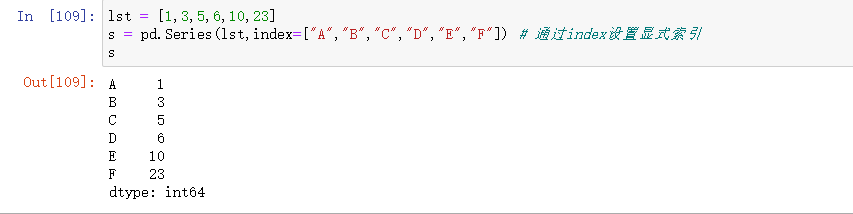
Pandas中的两个核心数据结构，一个是DataFrame，一个是Series。

## Series的创建方法

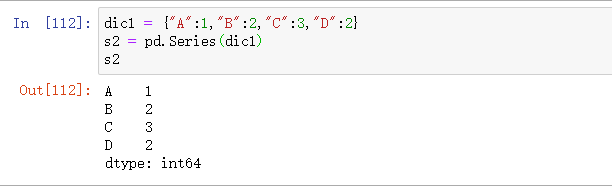
1. 列表方法
2. 隐式索引



1. 显示索引

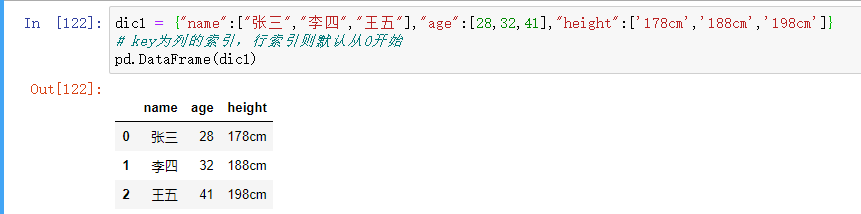


1. 字典创建

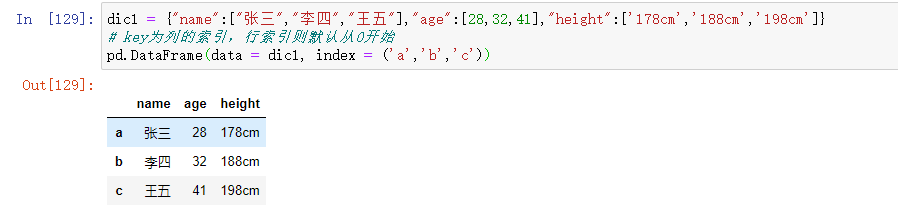


## DataFrame创建方法

1.使用pd.DataFrame构造函数来生成DataFrame, 最常用的方式是字典+列表。



也可指定index

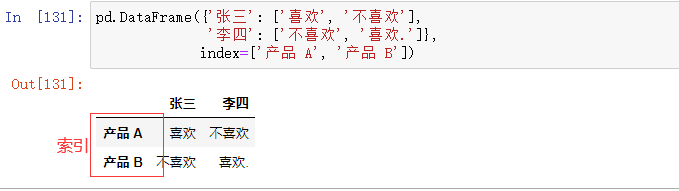


**Tips：**

1.DataFrame可以理解为Excel里面的一张表，有列名。

2.DataFrame包含一系列单独的条目，每个条目都有一定的值。 每个条目对应一行（或记录）和一列。DataFrame中使用的行标签列表称为索引， 可以在构造函数中使用索引参数为其赋值。

e.g.,



**“张三,产品A” 条目下对应的是”喜欢”, 表示张三喜欢产品A. 且DataFrame不仅限于使用数字类型数据, 也可以使用字符串。**

3.如果DataFrame是表，则Series是列表。Series本质上是DataFrame的单个列。但是，Series没有列名，它只有一个总名称。

4. Series和DataFrame密切相关。DataFrame实际上是多列Series“粘合在一起”的数据结构。

## ****导入数据文件****

通常情况下，做数据分析并不需要我们手动创建Dataframe和Series，而是通过导入外部文件，常用的文件类型有CSV和Excel。

* + - 1. 导入CSV文件

Pandas为我们提供了read\_csv()函数来读取csv格式的数据并将csv文件的内容转换成Dataframe.

e.g. data\_name = pd.read\_csv(‘file\_name.csv)

* + - 1. 导入Excel文件

Data\_name = read\_excel(‘file\_name.xls’)

## 导出数据

1. 导出csv文件

df.to\_csv(‘file\_name’)

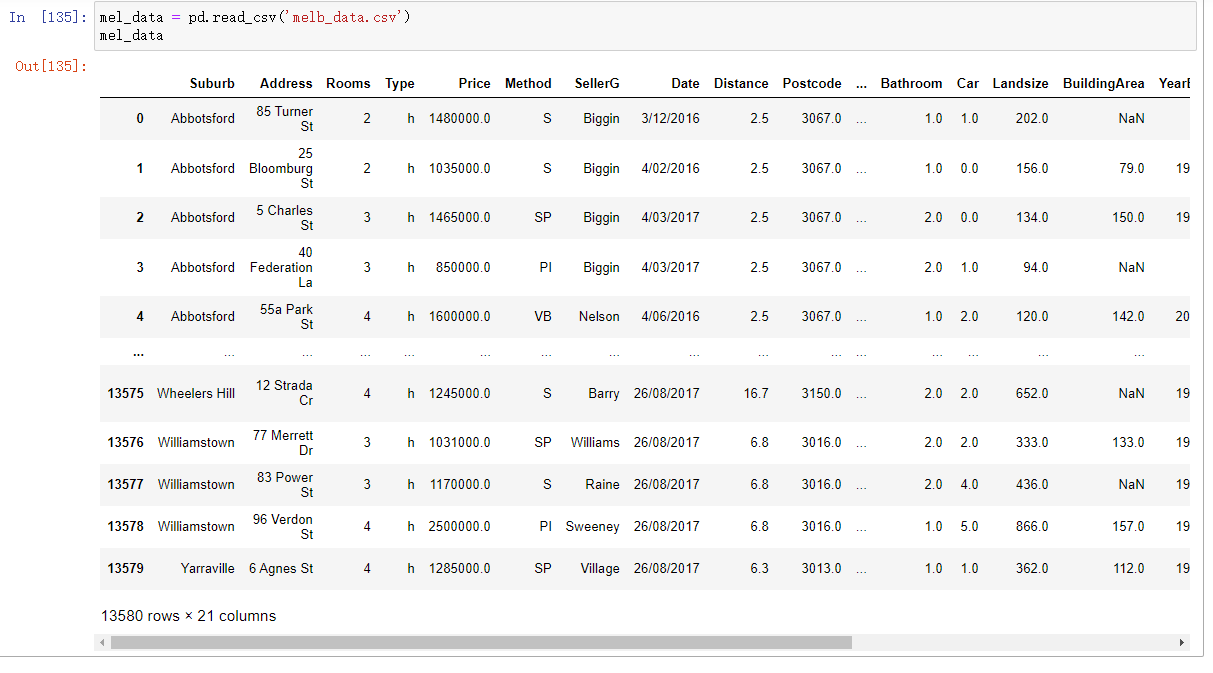
1. 导出Excel文件

df.to\_excel(‘file\_name’)

# 查看数据基本信息

通常我们拿到一份数据,需要对数据内容有一个基本的概念和对数据框架的了解. 下面我们将通过一个案例, 对pandas当中常用的一些查看数据内容的方法进行讲解.

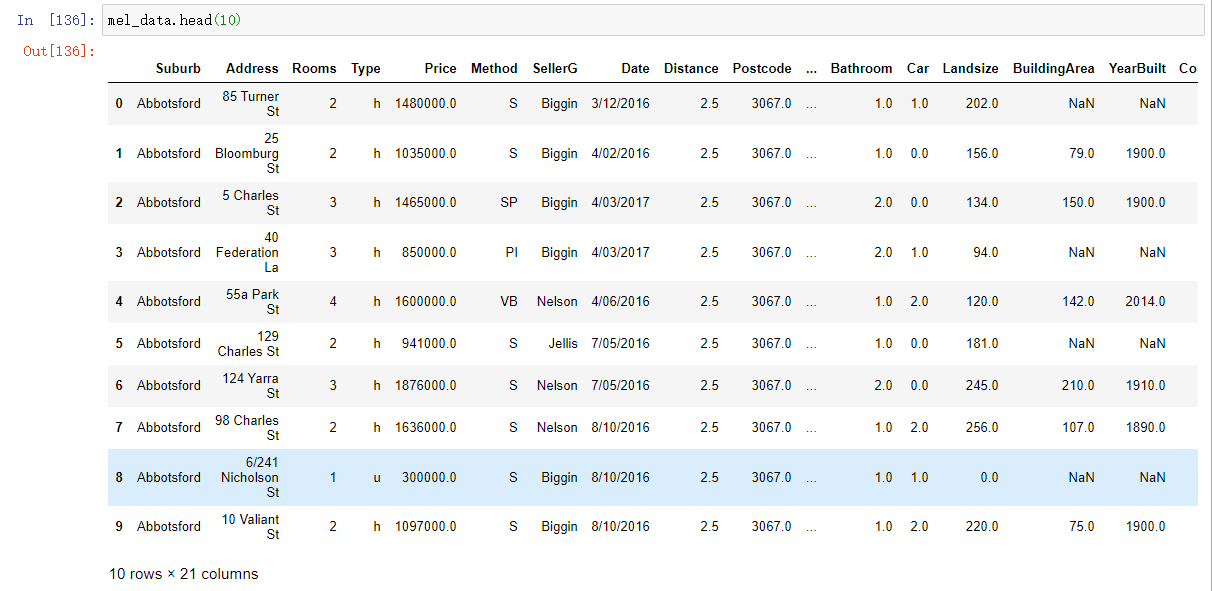
在这个案例里，我们将查看澳大利亚墨尔本的房价数据。



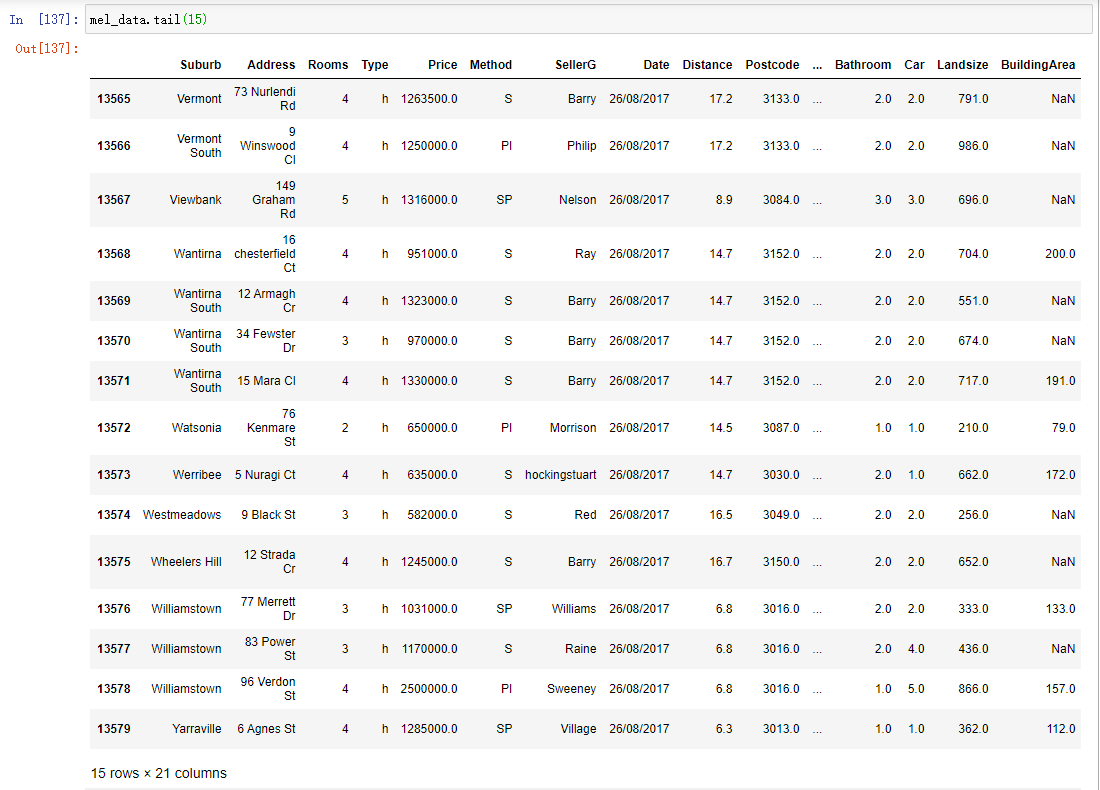
1. **查看数据，掐头看尾**

很多时候我们想要对数据内容做一个总览，用df.head()和df.tail()函数可以直接查看数据的前5行和最后5行. 这两个参数内可以传入一个数值来控制查看的行数，例如df.head(10)和df.tail(15), 表示查看前10行数据和最后15行数据.

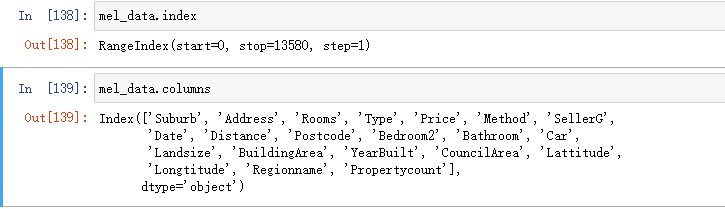
查看数据的前10行:



查看数据的后15行:

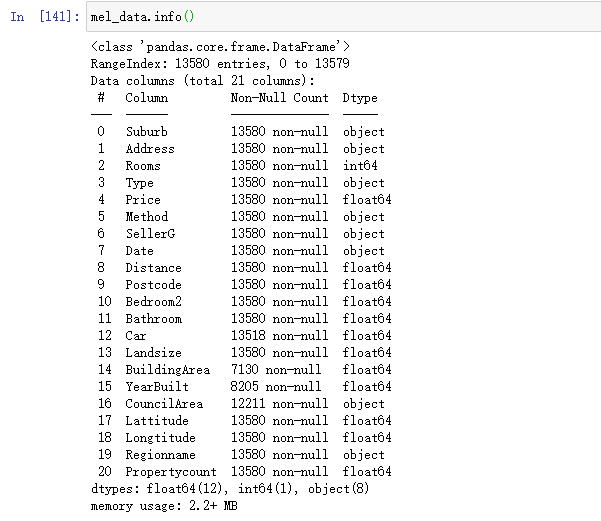


1. **查看index和columns**



1. 格式查看

df.info()函数可以帮助我们查看每一列数据的数据类型,及缺失情况.

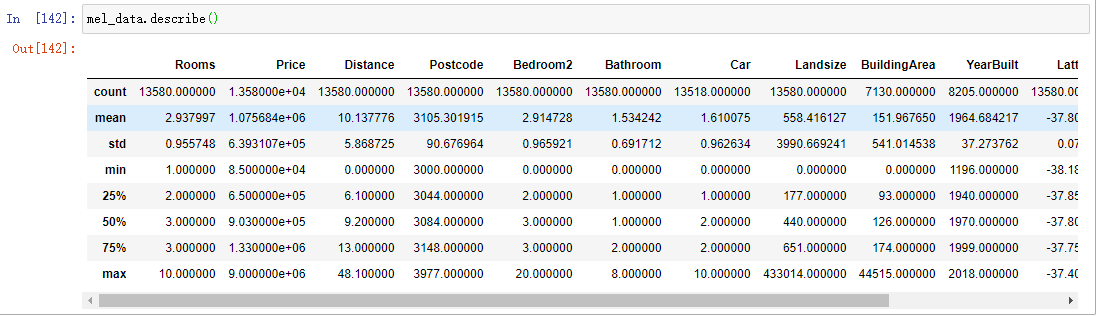


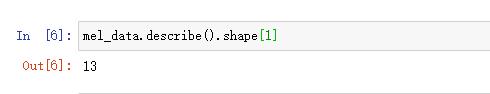
通过df.info()函数, 我们可以知道数据集的行列数，数据集的大小，每一列的数据类型，以及有多少条非空数据。

1. **统计信息概览**

通过df.describe()函数,可以快速计算数值型数据的统计指标, 主要包括:计数 count, 平均值 mean, 标准差 std, 最小值 min, 最大值 max, 25%分位数, 50%分位数和75%分位数.

注:df.describe()函数只针对数值型数据的列进行计算。非数值型数据,也就是object类不会返回统计指标值.





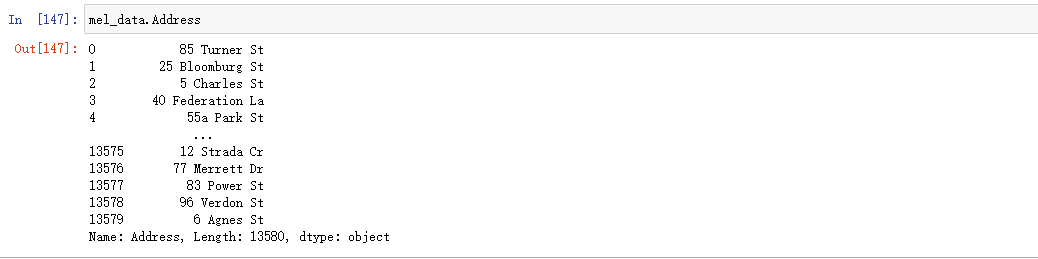
\*:df.shape[0]返回行数, df.shape[1]返回列数

# Pandas索引

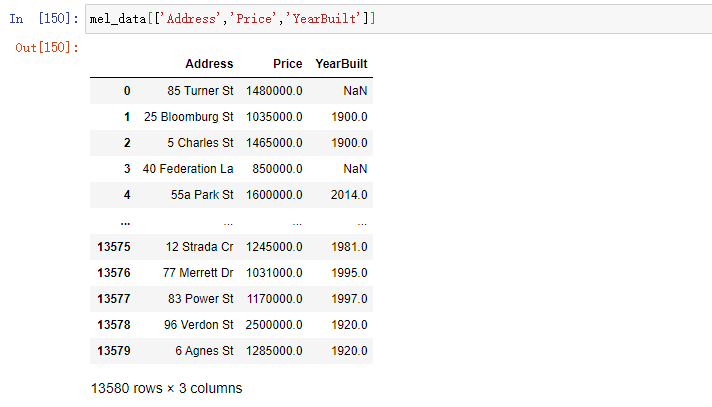
1. 原生索引方法

a. 在Python中，我们可以通过将其作为属性访问来访问对象的属性。 例如，book对象可能有title属性，我们可以通过调用book.title来获取相应信息。 pandas DataFrame中的列以相同的方式工作。

e.g. 访问房价数据的地址属性



b. 在 Python中有一个dict对象，我们可以使用索引（[]）运算符访问它的值。 同样，我们可以对pandas DataFrame列执行相同的操作。可以传入一个列表[ ],进行多列的索引.



这是从pandas DataFrame中选择特定Series的两种方法。它们虽然在语法上有差别, 但达到的目的是一样的. 不过, 索引[ ]运算符在处理的过程中可以保留字符串所在列的列名的优点.

1. Pandas索引方式

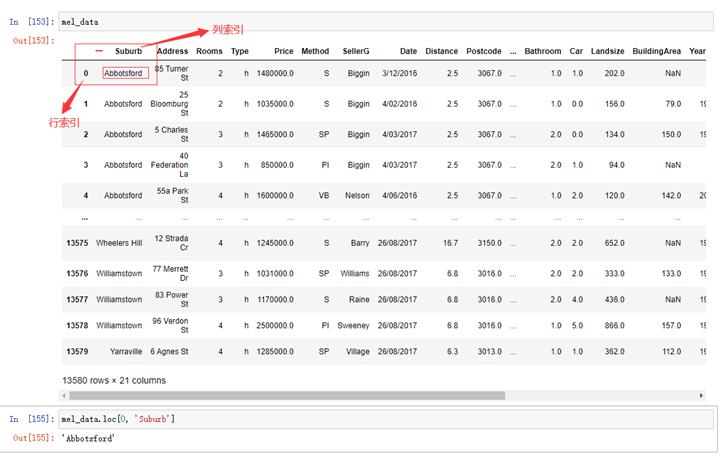
索引和切片对应字典和列表的两种方式,也是分为两种思路, 分别是**显式索引(loc)**和**隐式索引(iloc).**

**注:** loc和iloc都是行优先，列其次。

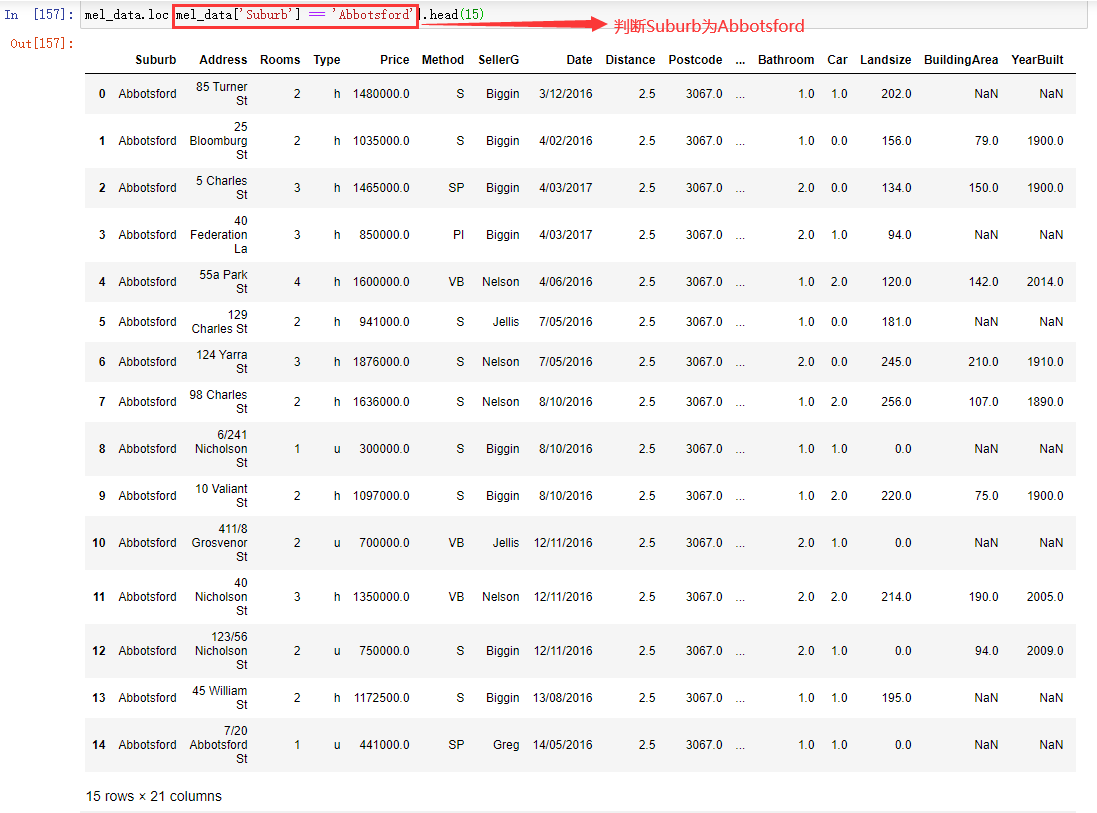
**a). 显示索引(loc):** 基于名称（标签）的索引

语法 df.loc[索引行,索引列]

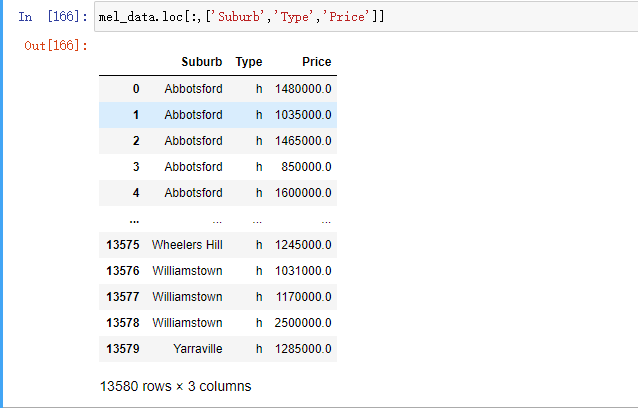
e.g. 获得房价数据中的第一个条目



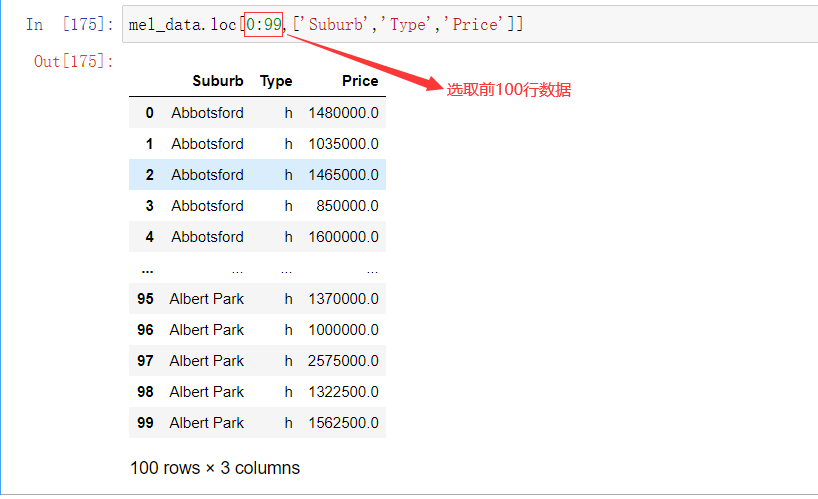
e.g. 选取Suburb为Abbotsford的前15行



e.g. 选取Suburb, Type和Price的所有行



e.g. 选取Suburb, Type和Price的前一百行



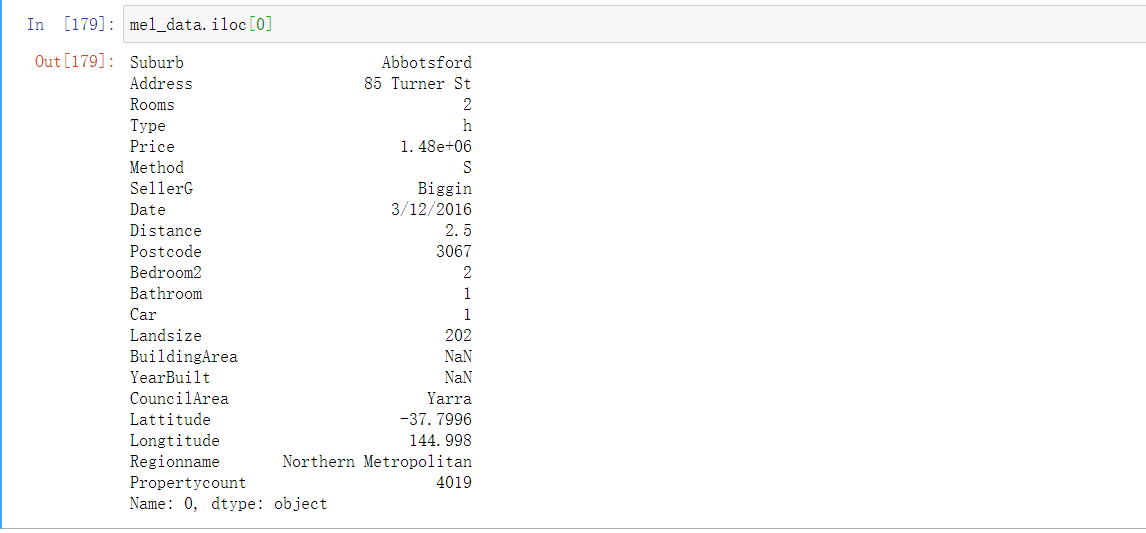
注: 当DataFrame索引是简单的数字列表时，df.loc [0：100]将返回101个条目, 要使用loc获取100个元素，你需要降低一个请求df.loc [0：99]。

* 1. **隐式索引:** 基于位置（整数）的索引

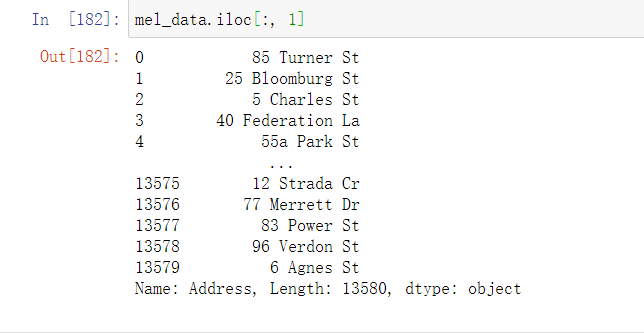
语法 df.iloc[索引行,索引列] (iloc是indexloc的缩写)

索引从0开始, 左闭右开

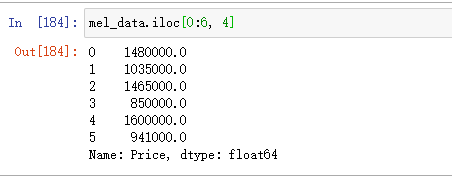
e.g. 选取第1行的数据



e.g. 获取第2列的数据



e.g. 获取前5行房屋的价格

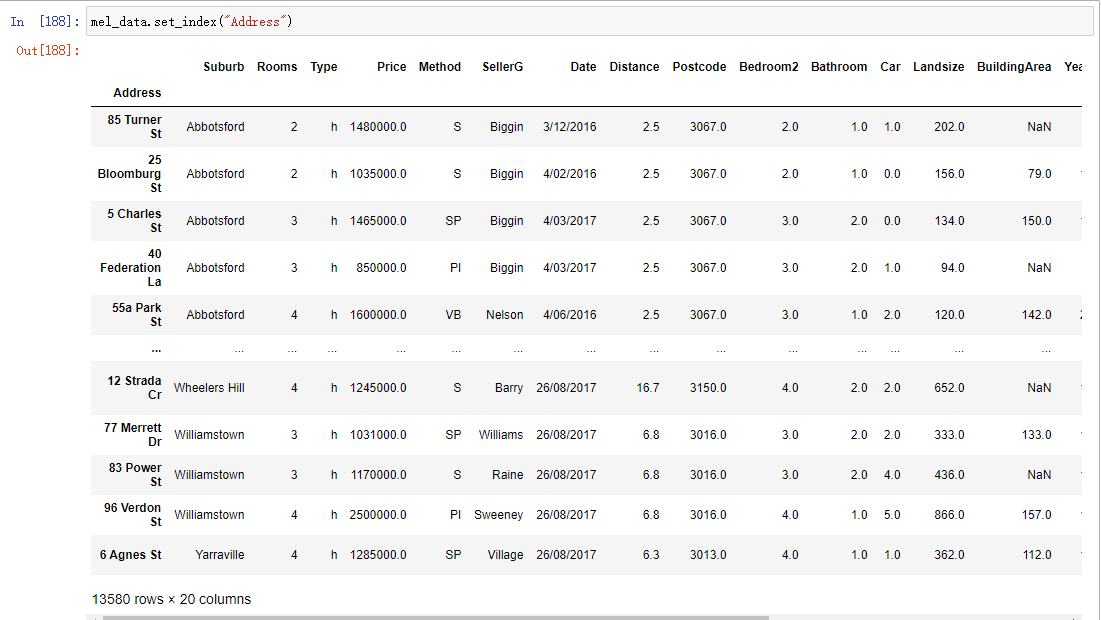


**注:** ‘**:**’运算符，来自原生Python，意思是“所有的”。 但是，当与其他选择器结合使用时，它可用于指示一系列值。

1. **操纵索引**

**当我们导入数据以后, 发现数据的行索引并不直观或者并不是合适的索引, 我们可以通过set\_index()方法来设定一个我们认为更适合数据的索引.**

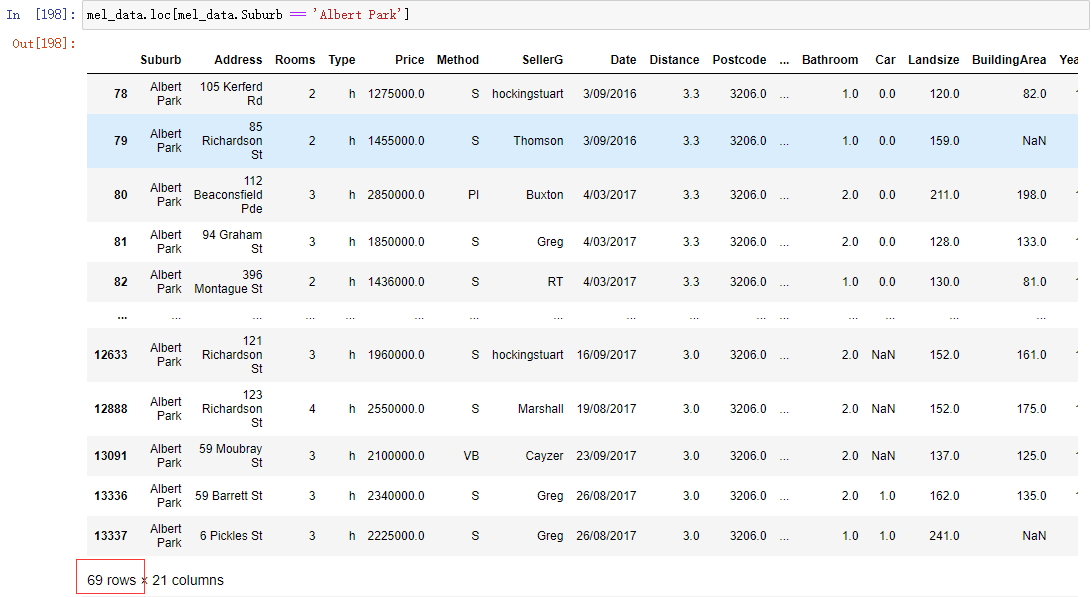
**e.g.将数据中的Address设定为索引**



1. **条件选择**

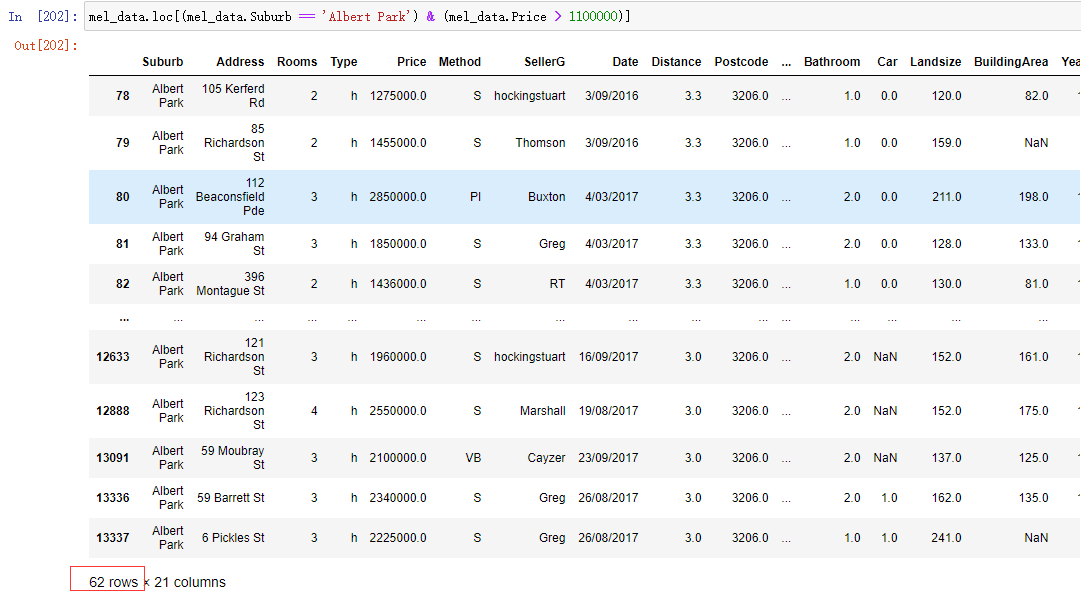
**在数据分析当中, 为了使数据当中的数字变得有意义,我们经常需要根据需求提取数据.**

**e.g. 查看Surburb中, 在Albert Park的房子**



**返回的DataFrame有69行, 数据本身有13580行, 由此可以看出, 来自Albert Park地区的房子占总数的0.51%.**

**e.g. 查看Albert Park地区房价高于1100000的房子**

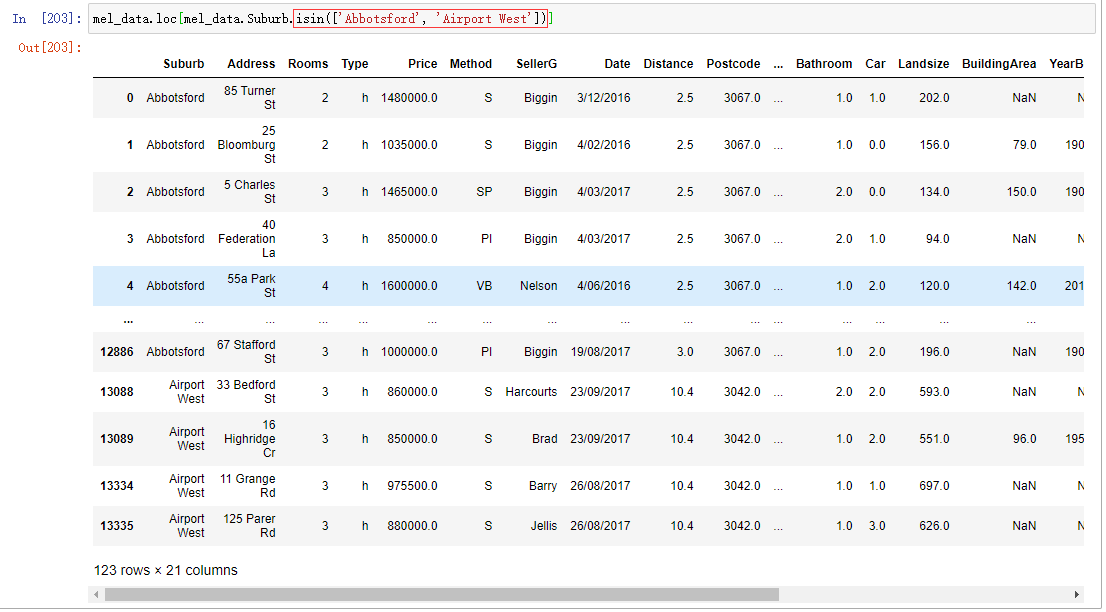


**返回的DataFrame有62行, 说明Albert Park地区房价高于1100000有62处.**

**Pandas本身也附带了一些条件选择器,比如 isin 和 isnull.**

1. **isin是你选择的值包括在列表中的数据.**

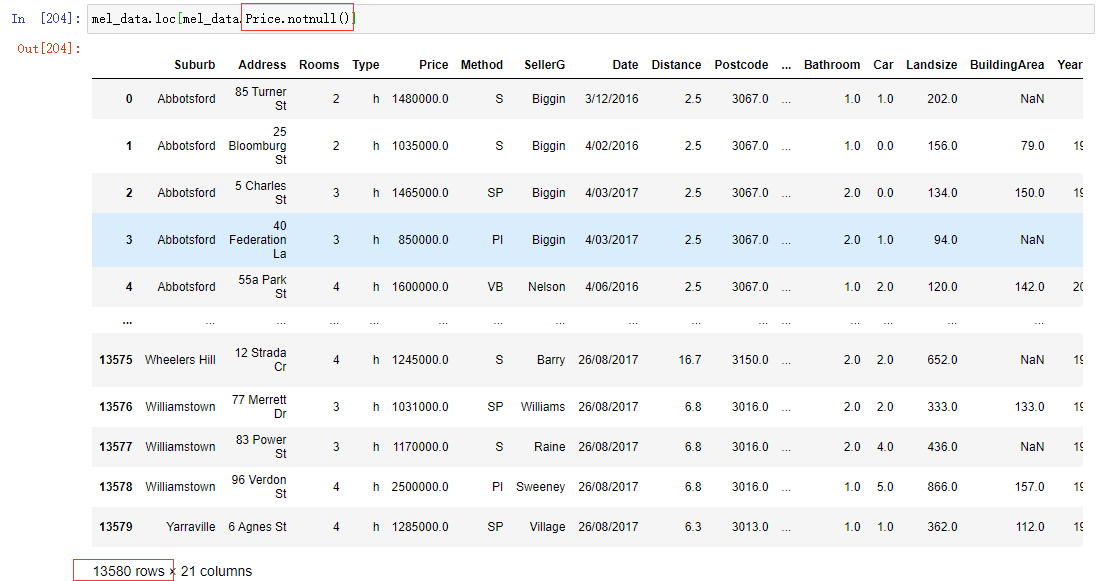
**e.g.** 选择来自Abbotsford, Airport West的房屋



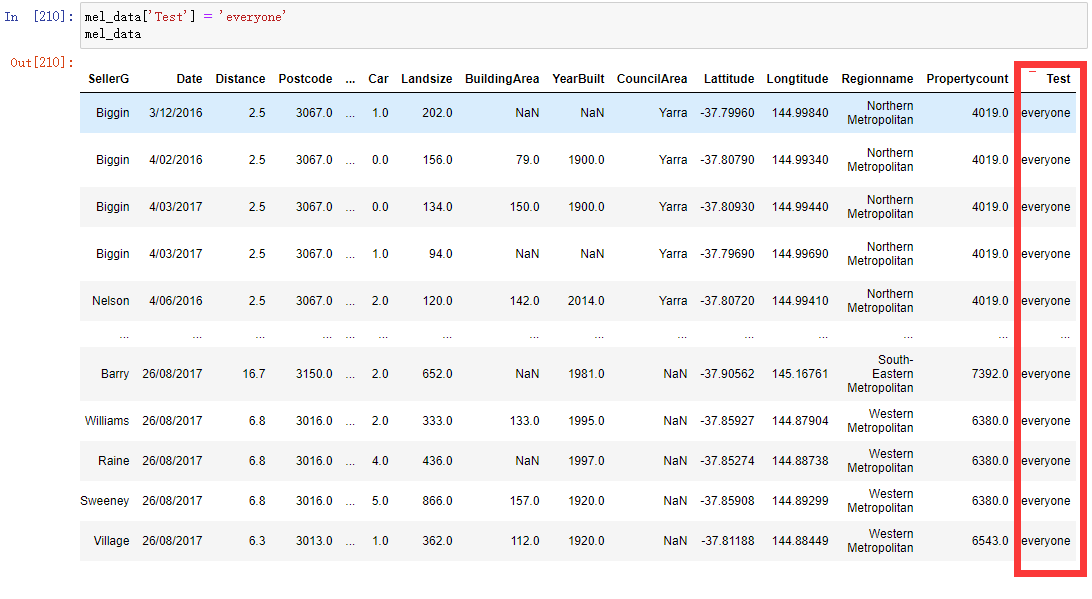
**返回的DataFrame有123行, 说明在Abbotsford和 Airport West的房屋一共有123处.**

1. isnull（和它的伴侣notnull）。 这些方法可以突出显示非空（NaN）的值。

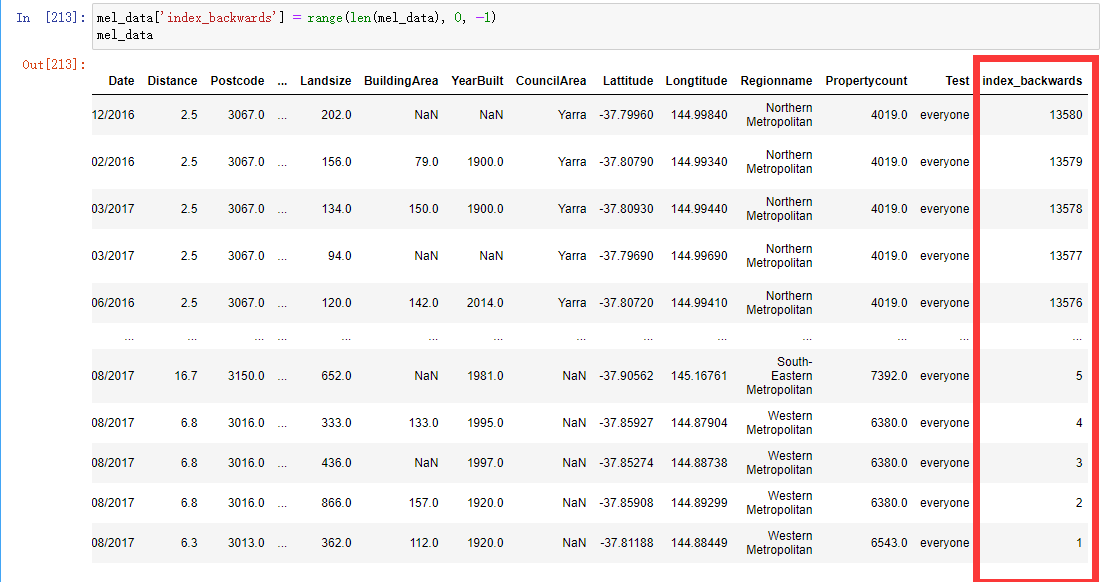
e.g. 过滤掉数据集中缺少价格标签的房屋

**返回的DataFrame有13580行, 说明Price中没有Null值.**

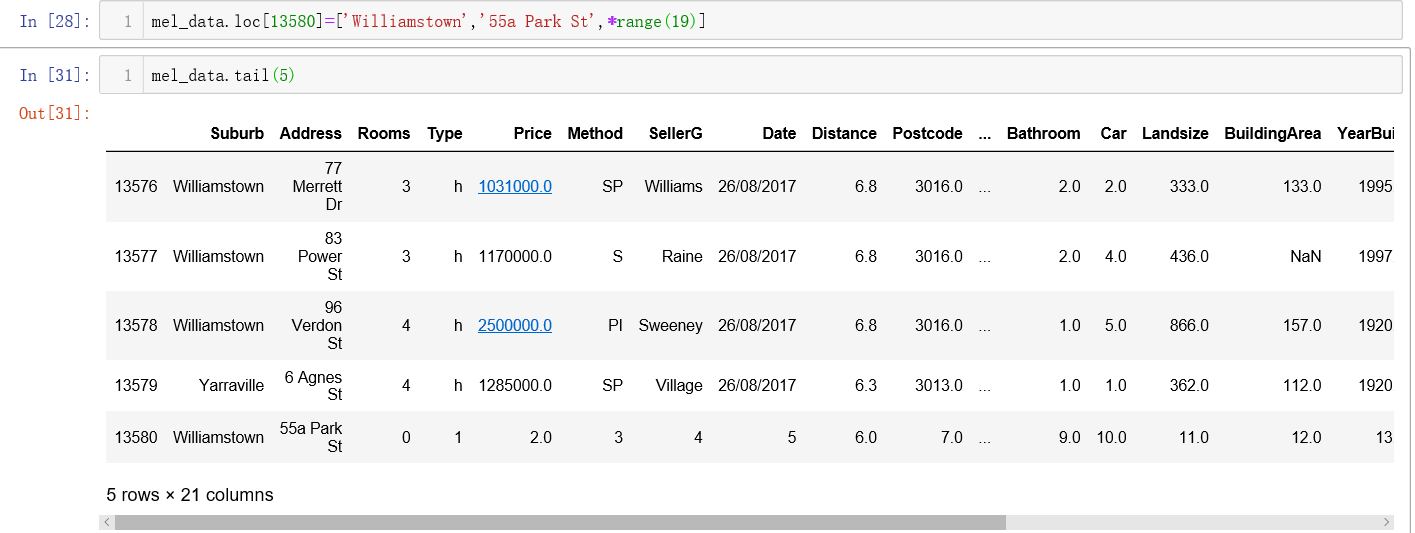
1. **赋值操作**
   1. **指定常量**



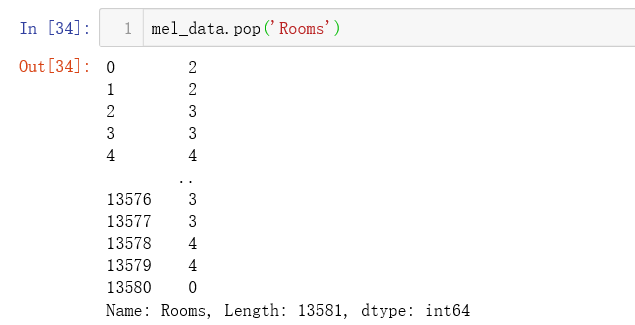
* 1. 使用可迭代的值



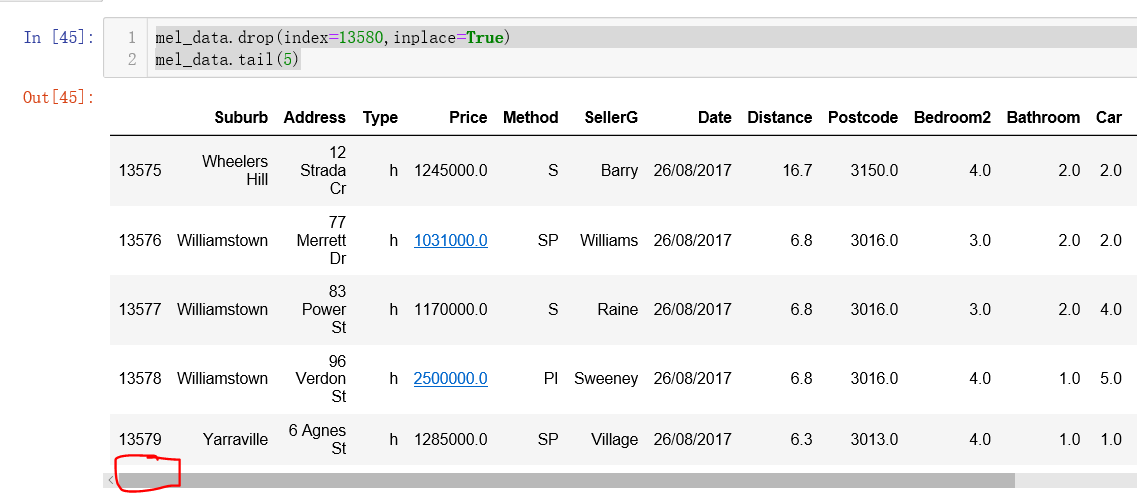
* 1. **增加某行必须用iloc**

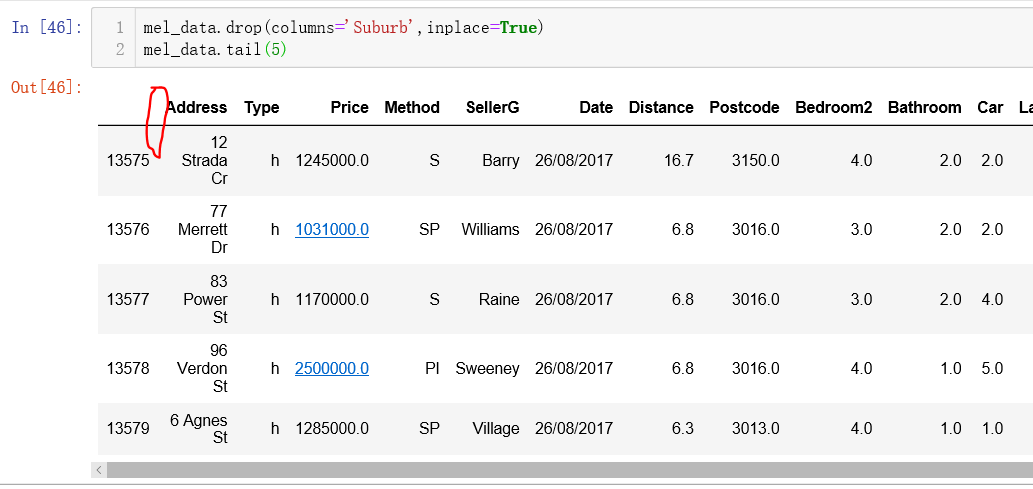


1. **删除操作：**
2. **pop方法，只能删除列**



**b)drop方法删除行和删除列**



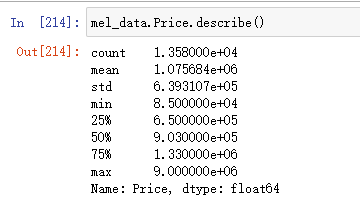


## 数据摘要

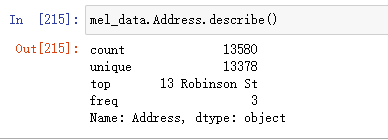
摘要功能

pandas提供了许多简单的“汇总函数”，它以某种有用的方式重构数据。

e.g.数值型 describe方法

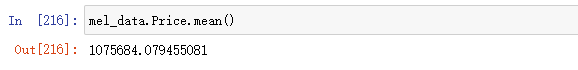


**e.g. 字符串 describe方法**

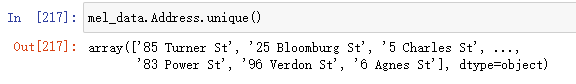


获得有关DataFrame或Series中的列的特定简单摘要统计信息

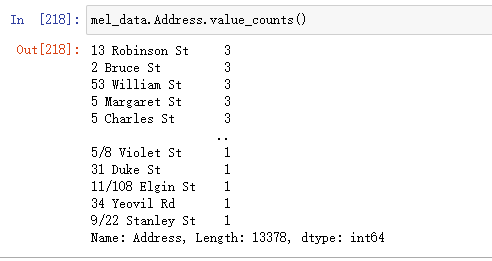
e.g. 查看Price的平均值, 可以使用mean()函数



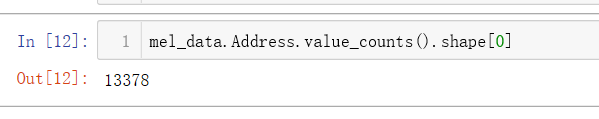
**e.g. 查看Address列的唯一值, 可以使用unique()函数**



**e.g.** 查看唯一值列表以及它们在数据集中出现的频率，可以使用value\_counts()函数



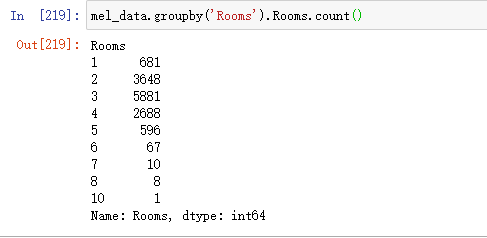
**e.g.** 查看数据形状可以使用shape()函数，用value\_counts().shape[0]找出某字段不重复项数目



# 分组聚合和排序

我们通常希望对数据进行分组，然后对数据所在的组执行特定操作。为此，我们可以使用groupby操作。

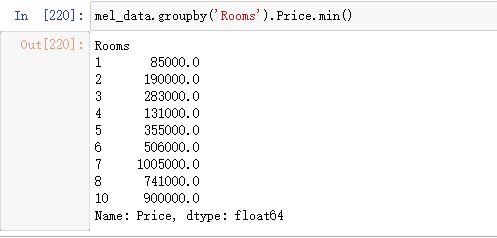
e.g. 计算不同房间数出现的次数



可以通过执行groupby操作来复制value\_counts执行的操作, value\_counts只是此groupby操作的快捷方式

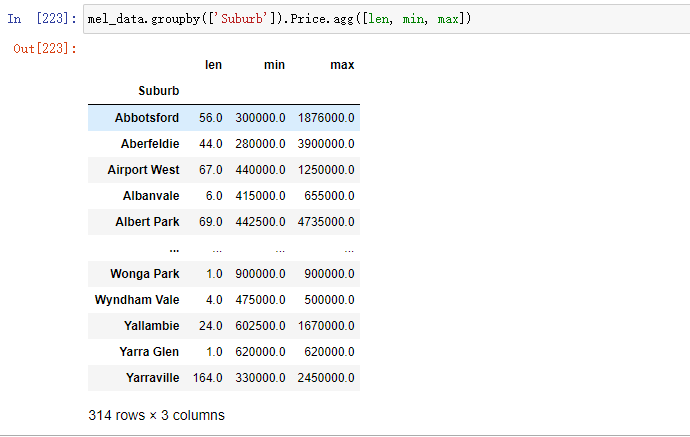
我们可以使用我们之前使用过的任何汇总函数来获得想要的数据.

e.g. 获得不同房间数的房子中最便宜的房子



另一个值得一提的groupby方法是agg，它允许你同时在DataFrame上运行多个不同的函数。

e.g. 可以生成数据集的简单统计摘要



排序

sort\_values()方法

e.g. 房子按照价格的降序排列



**ascending参数决定了排序顺序**，等于Flase则是从大到小的降序，设置为True则是升序。

注: 无论是删空的dropna，还是去重的drop\_duplicates，或者是排序的sort\_values，在对源数据进行操作后，源数据并未改变，这是因为我们没有对这几个函数的inplace值进行设置，如果设置成inplace = True，删空、去重和排序都会在源数据上生效。

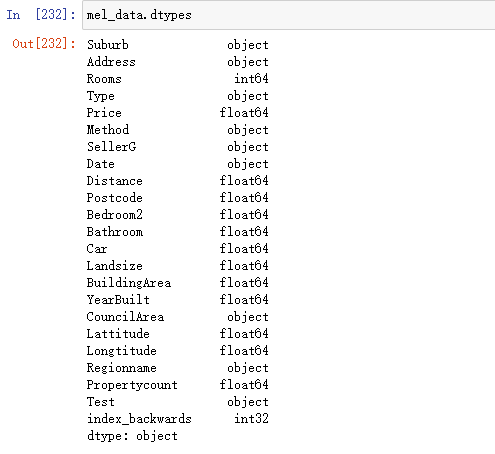
但这里为了避免出现不必要的错误而无法更改，更建议大家把操作后的源数据赋值给新的变量，如new = df.dropna()，而不是将源数据的inplace参数设置为True。

# 数据类型和缺失值

数据类型

DataFrame或Series中列的数据类型称为dtype, 可以使用dtype属性来获取特定列的类型.

e.g. 使用**dtypes**, 返回数据中每一列的数据类型



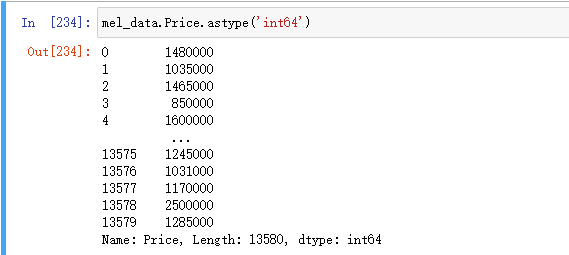
e.g.使用**dtype**, 返回指定列的数据类型



转换数据类型

使用astype函数，只要这种转换有意义，就可以将一种类型的列转换为另一种类型。

e.g. 将Price列从其现有的float64数据类型转换为int64数据类型



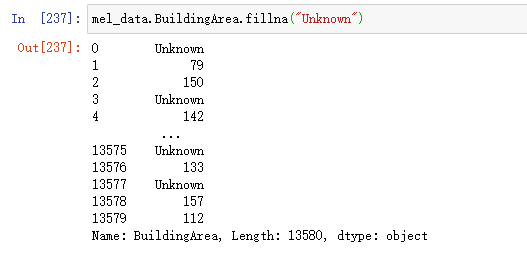
缺失数据

条目缺失值为NaN，“非数字”的缩写. pandas提供了一些特定于缺失数据的方法, 替换缺失值是一种常见操作。 pandas为这个问题提供了一个非常方便的方法：fillna。

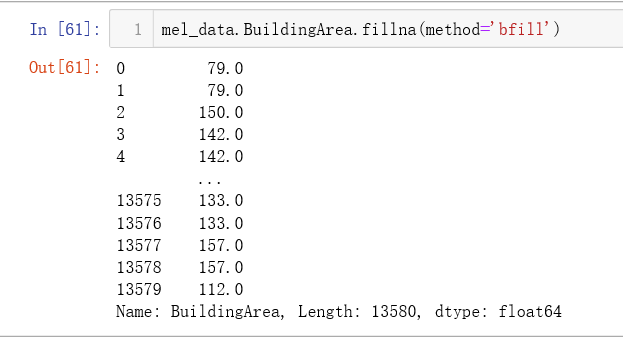
e.g. 首先判断缺失值, BuildingArea



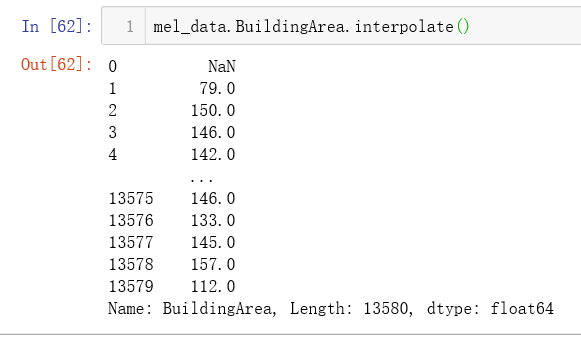
用“Unknown”替换每个NaN



上下文填充：method=‘ffill’依上文，’bfill’依下文

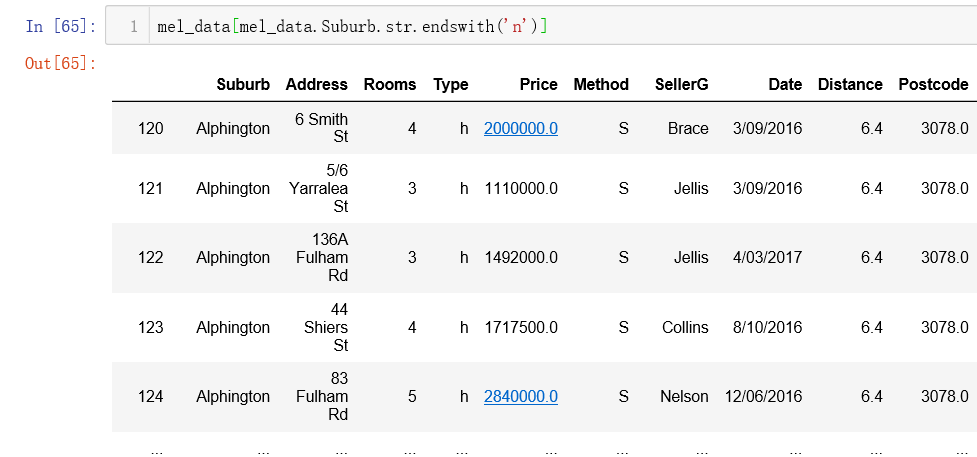


上下文线性插值填充：



文本数据处理

用data..str.调用方法，有 replace()，len(), lower(), upper(), find(), count(), strip(), split() ,contains() , startswith(), endswith等

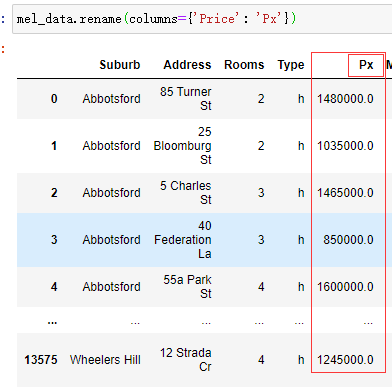


# 数据重命名和合并

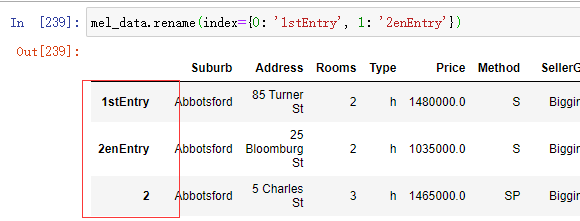
数据重命名

通常，数据将通过列名，索引名或其他命名约定来获取。 在这种情况下，我们可能会使用pandas重命名rename()函数来更改有问题的条目的名称。rename()函数允许重命名索引名称或者列名.

e.g. 重命名列名Price为PX



e.g. 重命名索引上的一些元素, 通常使用字典传入更方便



注: 可能经常重命名列，但很少重命名索引值。 为此，set\_index通常更方便。

合并

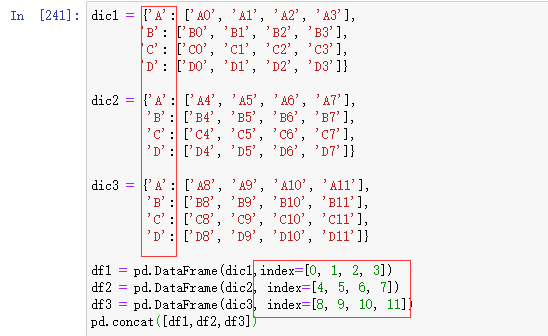
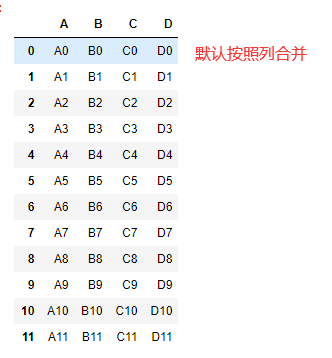
在对数据集执行操作时，我们有时需要以特别的方式组合不同的DataFrame和/或Series.

1. **级联**

根据指定的行或列, 进行值的拼接，不参与任何计算，只是把多个DataFrame变成1个.

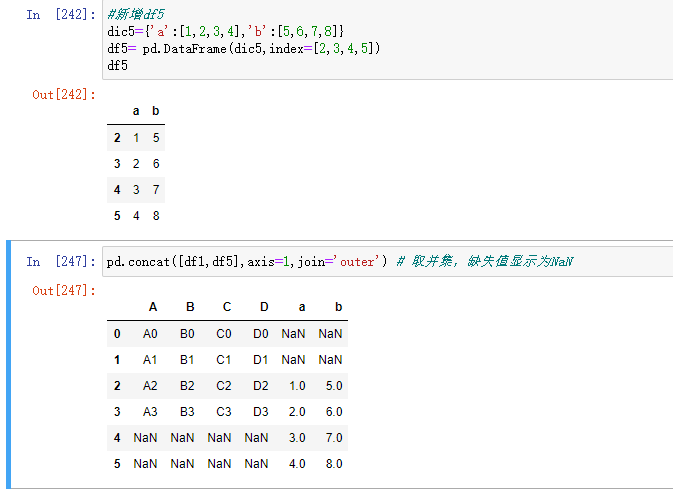
常用函数: **pd.concat pd.append**

1. pd.concat关键参数
2. axis=0 列方向级联拼接，axis=1行方向级联拼接，默认为0
3. join为级联方式，outer会将所有的项进行级联（忽略匹配和不匹配），取并集，而inner只会将匹配的项级联到一起，不匹配的不级联,取交集。
4. join\_axes index对象列表，用于其他n-1轴的特定索引，可以指定根据哪个轴来对齐数据

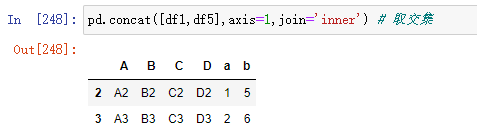
 

join的用法，取并集或交集

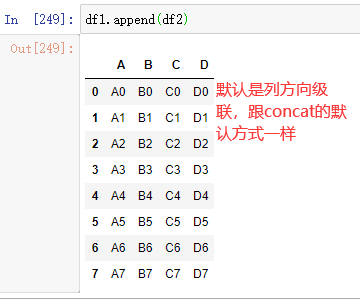
取并集:



取交集:



1. **df.append()方法**



**pd.merge() :** 需要根据合并条件进行两表合并，即两个DataFrame需要具有相同的列，然后再进行条件连接合并，而concat单纯根据索引就能进行拼接。

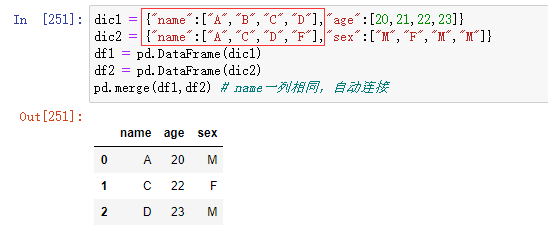
pd.merge()重要参数

* **left 和right 定义左表和右表，指定需要连接的两个DateFrame**
* **how连接方式**{'left', 'right', 'outer', 'inner'}, 默认'inner'

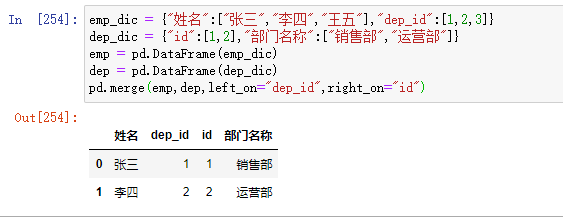
1. inner 内连接，只连接两个DataFrame的交集
2. left 左连接，以左表为主，会显示左表全部内容，右表根据左表匹配，未匹配上不显示
3. right 右连接，以右表为主
4. outer 全外连接，会显示所有内容

* **on**如果两个Frame 指定连接条件的列名一致，可以使用这种方式
* **left\_on** 当两个frame列名不一致时，根据定义的左表和右表分别定义连接的字段名
* **right\_on** 当两个frame列名不一致时，根据定义的左表和右表分别定义连接的字段名
* **left\_index** 为True时，代表以索引作为连接条件
* **right\_index** 同上

e.g. **两个DataFrame列名相同且内容一致，默认自动连接**



e.g. 连接条件列名不一致时，使用left\_on 和 right\_on



SQL操作方法: select \* from emp e left join dep d on e.dep\_id = d.id

References

<https://pandas.pydata.org/pandas-docs/stable/getting_started/10min.html>

EDA: Exploratory Data Analysis(探索性数据分析)

包: pandas\_profiling