


# 第5章-2:数据整合和数据清洗



Python数据科学：全栈技术详解

讲师：Ben

# 自我介绍

- 天善商业智能和大数据社区      讲师 – Ben
- 天善社区 ID - Ben\_Chang
- <https://www.hellobi.com> – 学习过程中有任何相关的问题都可以提到技术社区数据挖掘版块。

- 数据整合的介绍
  - SQL语句介绍
  - 数据纵向合并
  - 数据横向合并
  - SQL进行汇总
- 数据清洗
  - 错误值处理
  - 缺失值处理
  - 噪声值处理

## 5.1 数据整合

分析师希望得到信用卡持卡人的人口学信息，如下所示：

card_id	disp_id	issued	type	sex	birth_date
1	9	1998-10-16	金卡	男	1935-10-16
2	19	1998-03-13	普通卡	男	1942-12-28
3	41	1995-09-03	金卡	男	1968-08-27
4	42	1998-11-26	普通卡	男	1935-08-17
5	51	1995-04-24	青年卡	女	1979-12-02
7	56	1998-06-11	普通卡	男	1960-03-31
8	60	1998-05-20	青年卡	男	1980-02-19
9	76	1997-10-25	普通卡	女	1967-10-01
10	77	1996-12-07	普通卡	女	1956-02-18
11	79	1997-10-25	金卡	女	1969-03-10
12	83	1996-09-11	青年卡	女	1978-12-25
13	87	1994-06-29	普通卡	女	1946-11-17

信用卡  
(CARD)

客户信息  
(CLIENTS)

## 5.1.1 SQL语句介绍

## SQL语言的主要特点

1. Structured Query Language 结构化查询语言
2. SQL语言类似于英语的自然语言，简洁易用。
3. SQL语言是一种非过程语言，即用户只要提出“干什么”即可，不必管具体操作过程，也不必了解数据的存取路径，只要指明所需的数据即可。
4. SQL语言是一种面向集合的语言，每个命令的操作对象是一个或多个关系，结果也是一个关系。
5. SQL语言既是自含式语言，又是嵌入式语言。可独立使用，也可嵌入到宿主语言中。

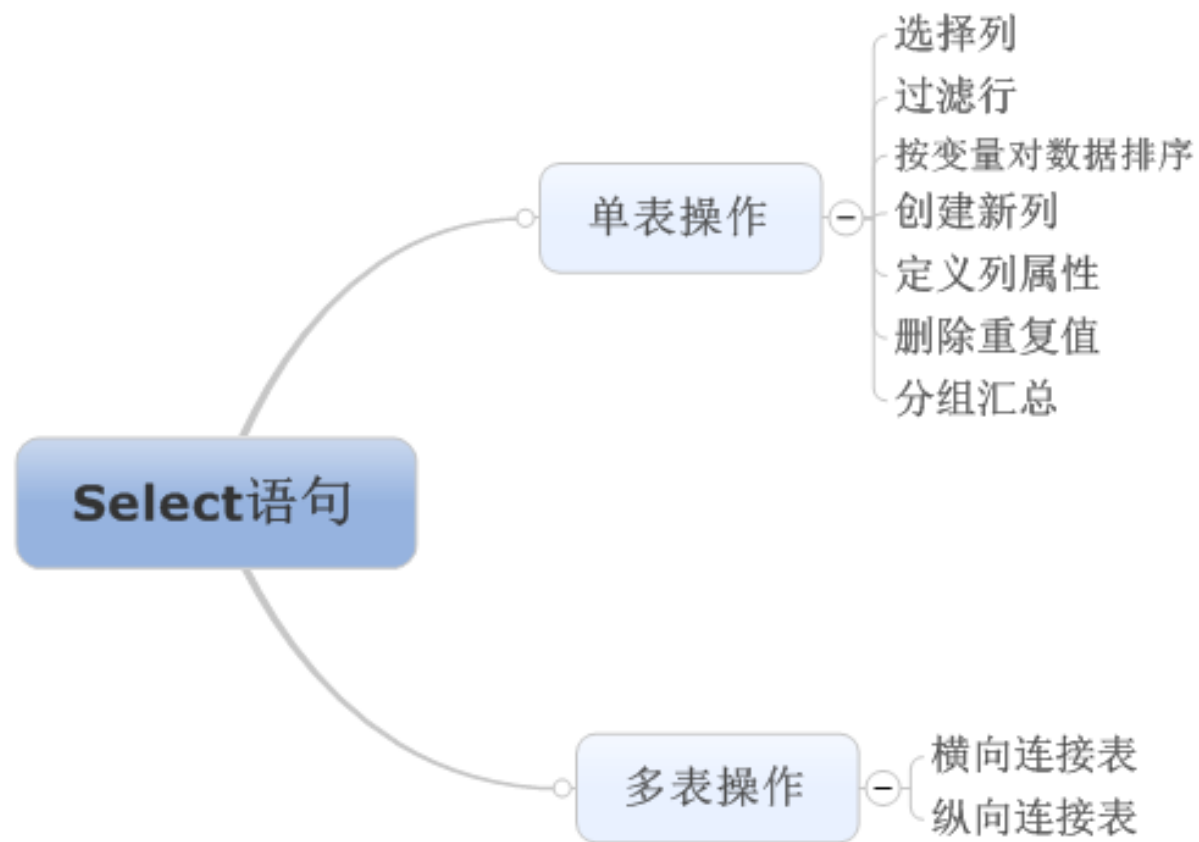
- SQL语句的动词只有九条。

数据定义DDL	CREATE, DROP, ALTER
数据查询DQL	SELECT
数据操纵DML	INSERT, UPDATE, DELETE
数据控制DCL	GRANT, REVOTE



# Select数据查询语句

- SELECT数据查询是最核心和常用的操作。



# SELECT一般格式

```
SELECT [ALL | DISTINCT] <目标列表达式> [别名] [ ,  
    <目标列表达式> [别名] ]...  
FROM <表名或视图名>[ , <表名或视图名>]...  
[WHERE <条件表达式>]--  
[GROUP BY <列名1>  
[HAVING <条件表达式> ]]  
[ORDER BY <列名2 >[ASC | DESC]] ;
```

# SELECT语句的特点

- SELECT语句有以下特点：
  - 选择满足特定条件的数据
  - 分组数据
  - 为数据指定顺序

# 选择表中指定列

- 在SELECT语句后指定需要输出的列。

例子：展示销售数据中的年份、市场、销售和利润。

```
select year,market,sale,profit  
from sale;
```

# 删除重复的行

- 使用DISTINCT关键字删除查询结果中重复的行。

例子：展示销售数据中的不同年份。

```
select DISTINCT year  
from sale;
```

# 选择满足条件的行

- 使用WHERE子句查询结果中满足条件的行。

例子：展示2012年的销售数据。

```
select *  
      from sale  
     where year=2012;
```

- 用 *WHERE* 语句 选择满足特定条件的观测。

- *WHERE* 语句的一般形式：

**WHERE** *where-表达式*;

- *where-表达式* 是由一系列运算符和操作数组成的用来选择观测的条件表达式。

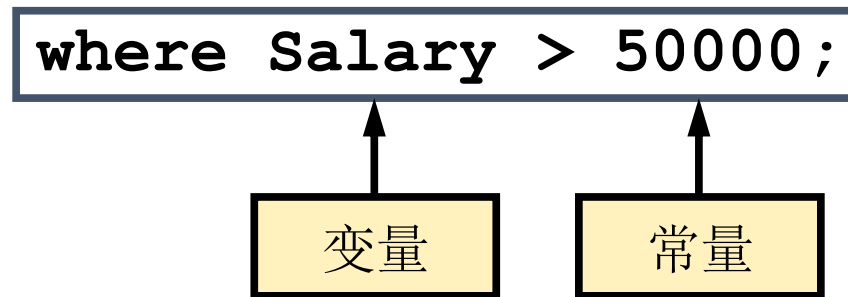
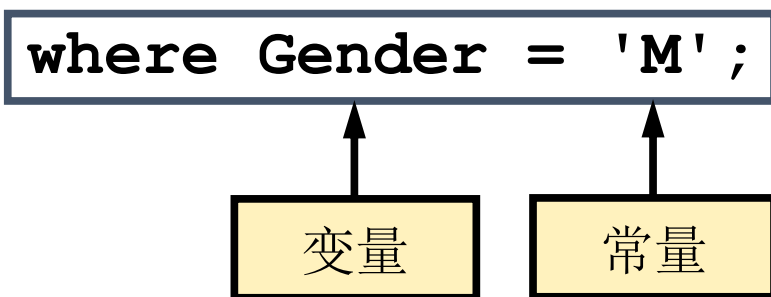
- 运算数包含常量和变量。
- 运算符包含算术运算符，比较算符和逻辑算符。

常量运算数是固定值。

- 字符值必须包含在引号中并且区分大小写。
- 数值不用引号。

一个变量运算数必须是来自输入数据集的一个变量。

例如：





# 运算符-比较运算符

- 比较运算符比较一个变量和一个值，或一个变量和另一个变量。

符号	说明
=	等于
!=	不等于
>	大于
<	小于
>=	大于或等于
<=	小于或等于
in	在列表中

- 例如：

```
where date >= ' 1998-12-01' ;
```

```
where com in ( 'AB' , 'CD' ) ;
```

# 运算符-算数运算符

- 算数运算符表示要进行算数计算。

- 例如：

```
where amount * 12 < 8000;
```

```
where amount / 12 * 1.10 >= 8500;
```

```
where (amount / 12 ) * 1.10 >= 8500;
```

```
where amount + Interest <= 10000;
```

符号	说明
*	乘法
/	除法
+	加法
-	减法

# 运算符-逻辑运算符

- 逻辑运算符合并或修改表达式。

符号	算符	说明
&&	AND	与
	OR	或
!	NOT	非

- 例如：

```
where date <= "1998-12-31" AND com  
in ('AB', 'CD');
```

# 对行进行排序

- 使用ORDER BY子句对指定行。

例子：按年份排序。

```
select year,market,sale,profit  
from sale  
order by year;
```

- 也可以使用变量在SELECT语句中的序号。

```
select year,market,sale,profit  
from sale  
order by 1;
```

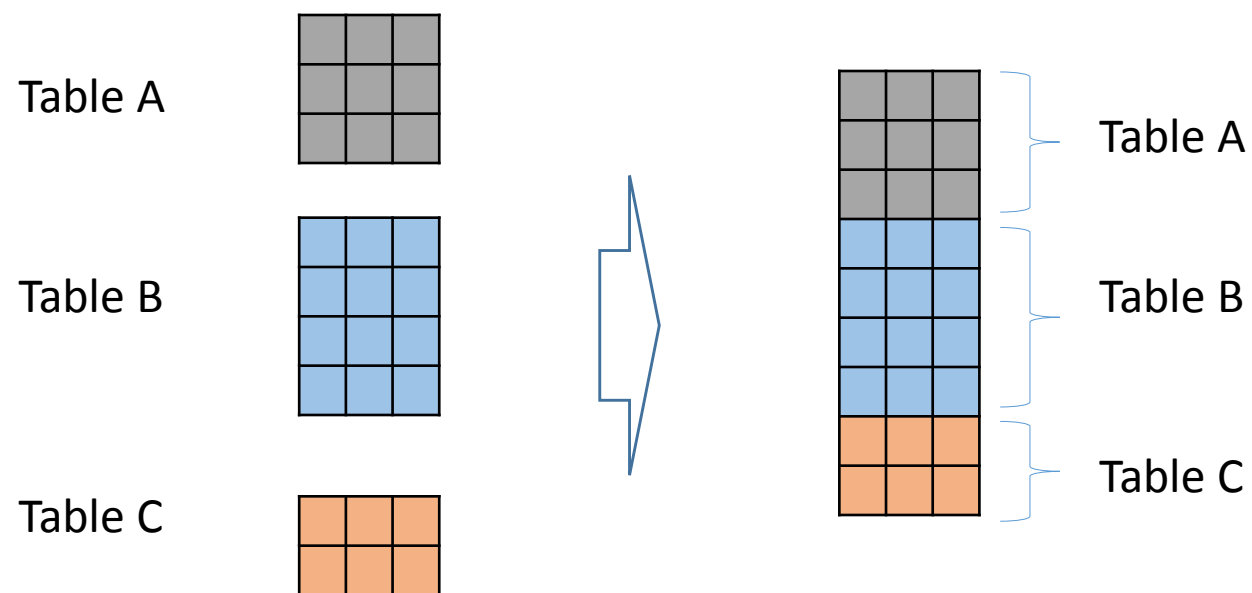
## 5.1.2 纵向连接表

# 学习目标

- 纵向连接多张表
- 比较 SQL 连接和pandas.DataFrame的连接方式

# 多表的纵向合并

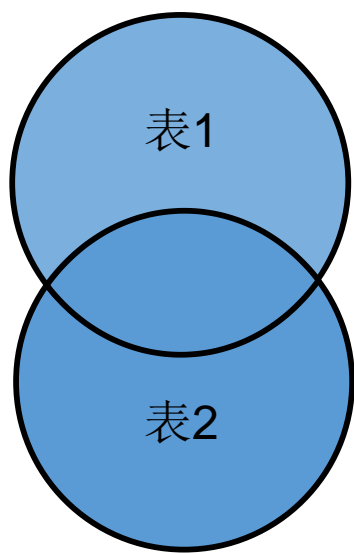
- 在SQL语句中使用集合运算来完成.



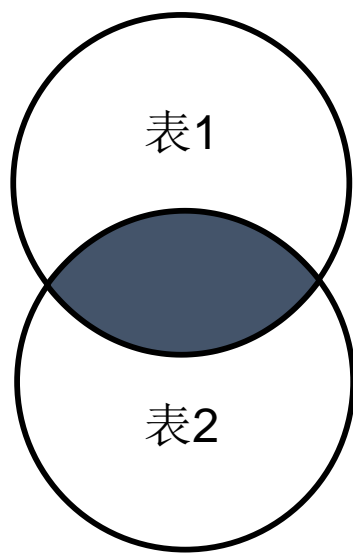
在pandas中使用concat() 函数完成。

# 集合查询—并、交、差

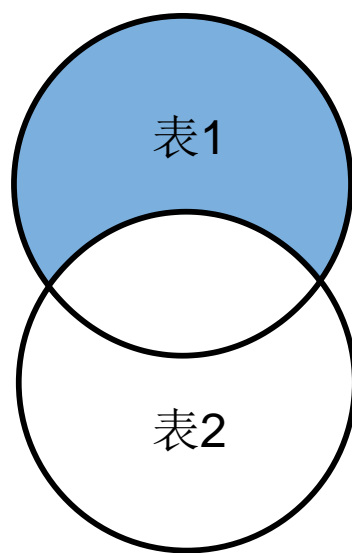
- 将两张表中的记录看作一个集合，则
  - 并集是两张表中重复的记录只保留一份，不重复都保留；
  - 交集是只保留一份重复的记录；
  - 差集是只保留表1中不重复的记录保留。



并UNION



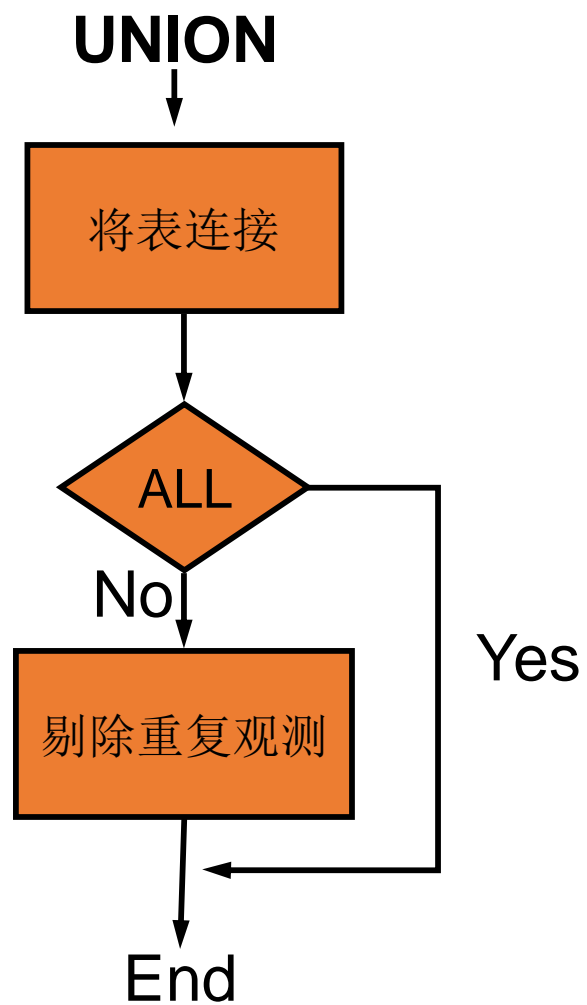
交INTERSECT



差EXCEPT



# 并集操作的流程图



# 并集操作示例

- SQL 操作会生成一个中间表（intermediate result set）

**Table ONE**

X	A
1	a
1	a
1	b
2	c
3	v
4	e
6	g

**Table TWO**

X	B
1	x
2	y
3	z
3	v
5	w

```
select *  
  from one  
union  
select *  
  from two;
```

**Intermediate Results**

①	②
1	a
1	a
1	b
1	x
2	c
2	y
3	v
3	v
3	z
4	e
5	w
6	g

# 并集操作示例

- 注意，union后面没有跟随all选项，因此剔除重复值。

**Table ONE**

X	A
1	a
1	a
1	b
2	c
3	v
4	e
6	g

**Table TWO**

X	B
1	x
2	y
3	z
3	v
5	w

```
select *  
  from one  
union  
select *  
  from two;
```

**Intermediate  
Results**

①	②
1	a
1	a
1	b
1	x
2	c
2	y
3	v
3	v
3	z
4	e
5	w
6	g

# 并集操作示例

- 最后结果

**Table ONE**

X	A
1	a
1	a
1	b
2	c
3	v
4	e
6	g

**Table TWO**

X	B
1	x
2	y
3	z
3	v
5	w

```
select *  
  from one  
union  
select *  
  from two;
```

**Final Results**

X	A
1	a
1	b
1	x
2	c
2	y
3	v
3	z
4	e
5	w
6	g

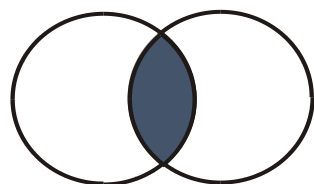
## 5.1.3 横向连接表

# 学习目标

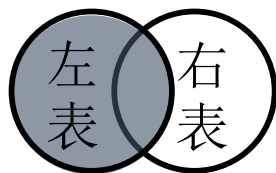
- sql根据共同列连接多张表
- 使用pandas的方法进行横向连接

# 横向连接查询基础

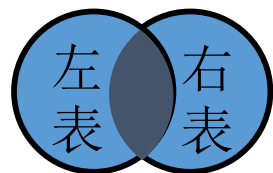
- 交叉连接(cross join, 笛卡尔乘积)：查询结果包括两张表观测的所有组合情况，这是SQL实现两表合并的基础，但是极少单独做这种操作；
- 内连接(inner join)：查询结果只包括两张表向匹配的观测，用法简单，但是在数据分析中谨慎使用，因为会造成样本的缺失；



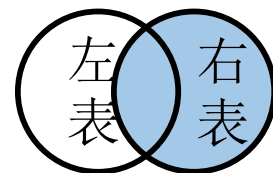
- 外连接(outer join)包括左连接、右连接，全连接



显示左表所有观测

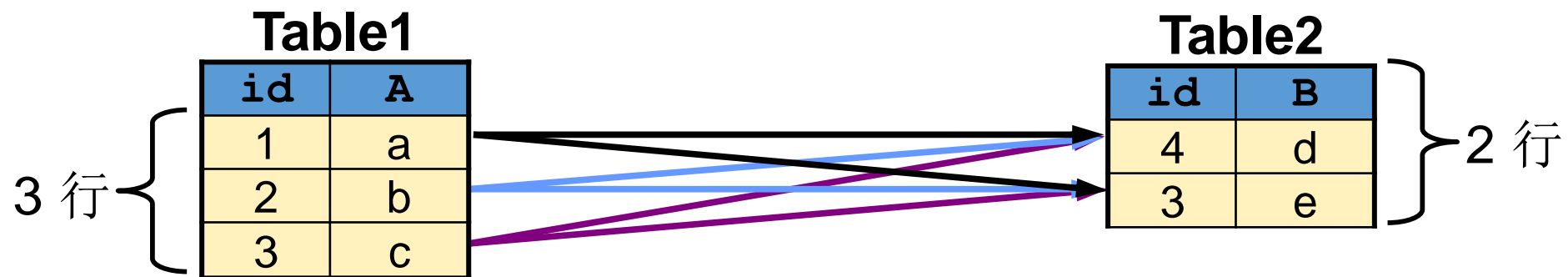


同时两张表的所有观测



显示右表所有观测

# 笛卡尔积



结果集

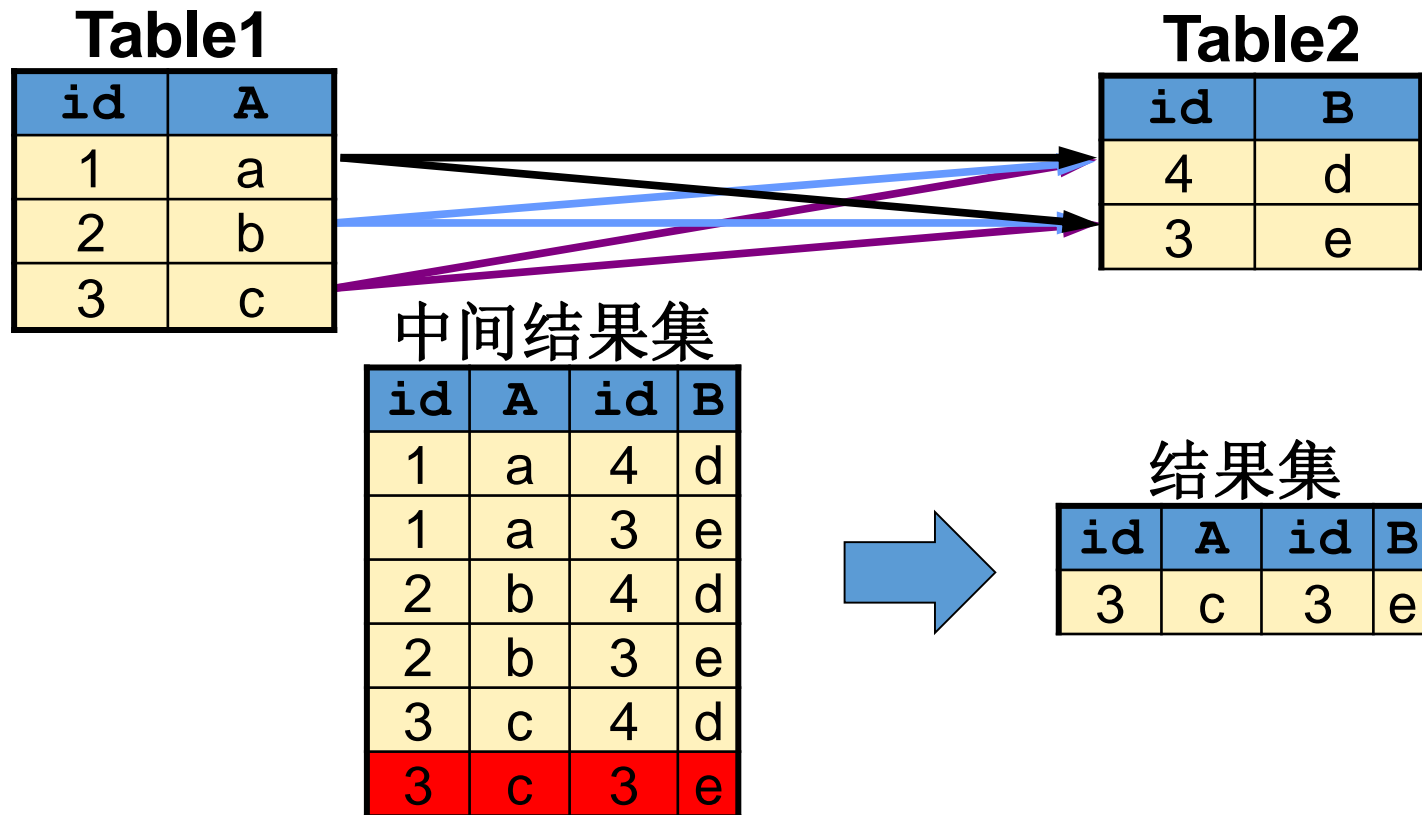
id	A	X	B
1	a	4	d
1	a	3	e
2	b	4	d
2	b	3	e
3	c	4	d
3	c	3	e

6 行



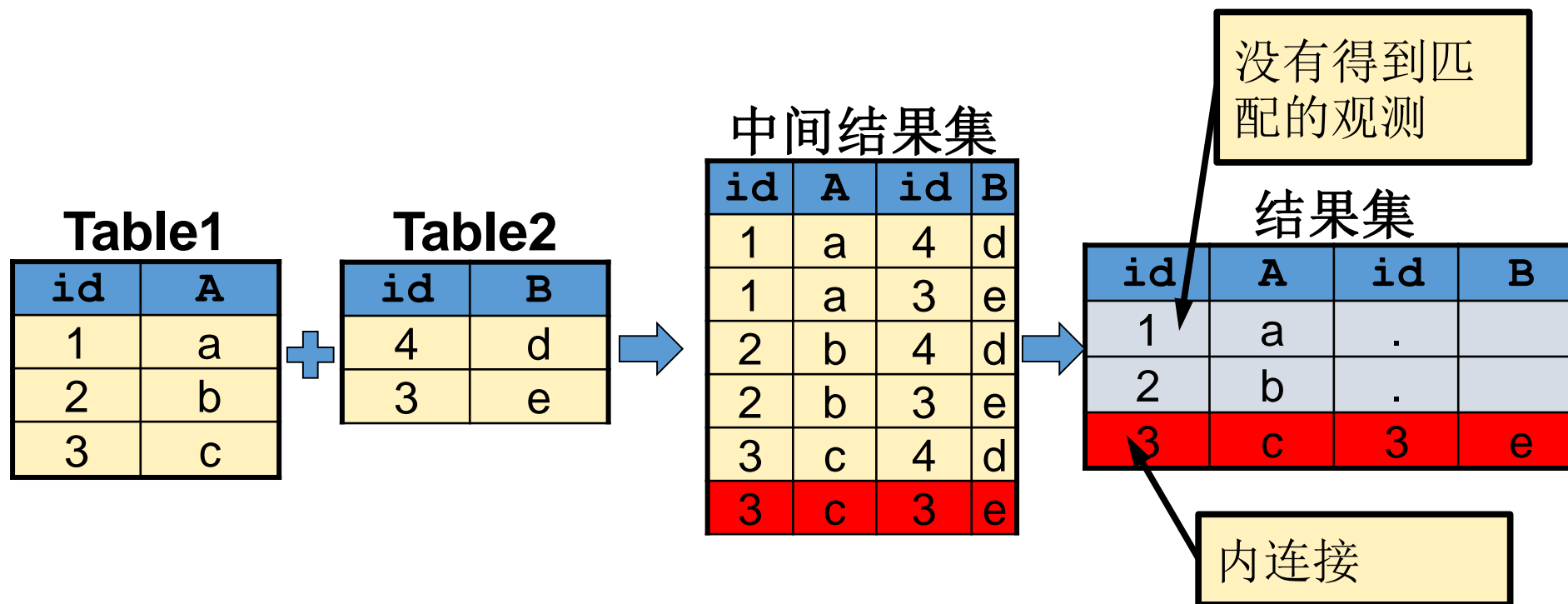
# 内连接

- 内连接在笛卡尔积基础上加入了连接条件，其实就是限制条件，这类似单表操作中对观测进行筛选



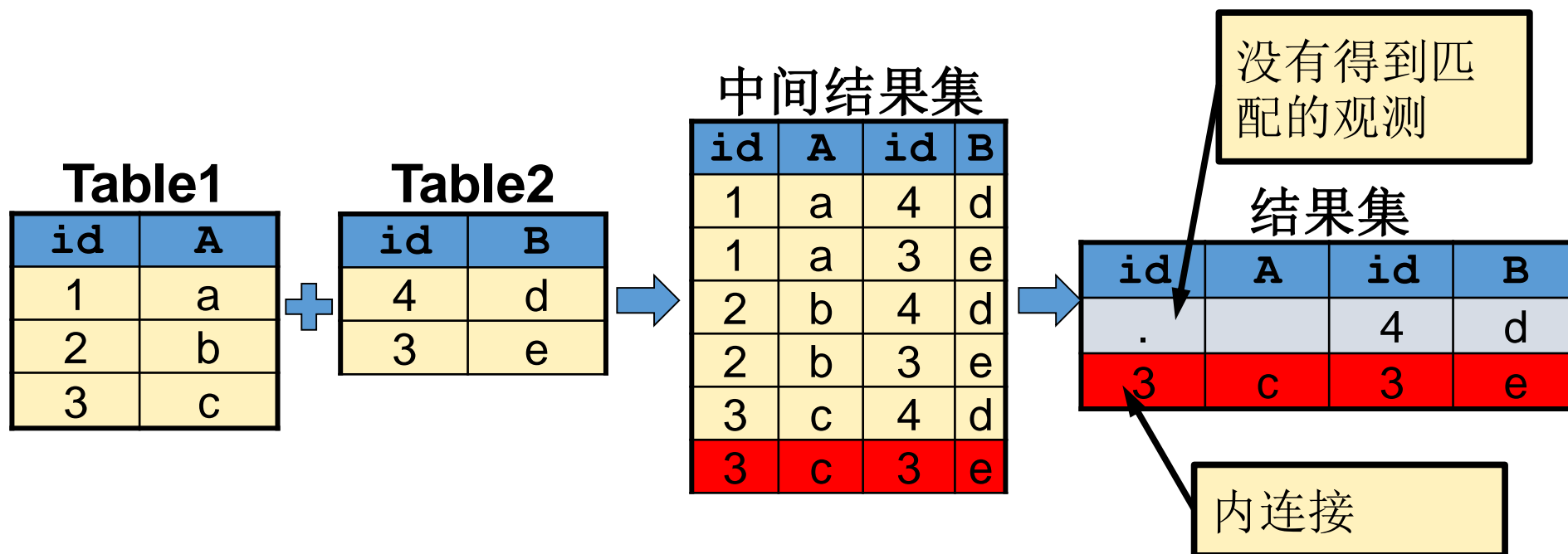
# 左连接

- 左连接等价于两部分的叠加：内连接+左表中没有匹配的观测。



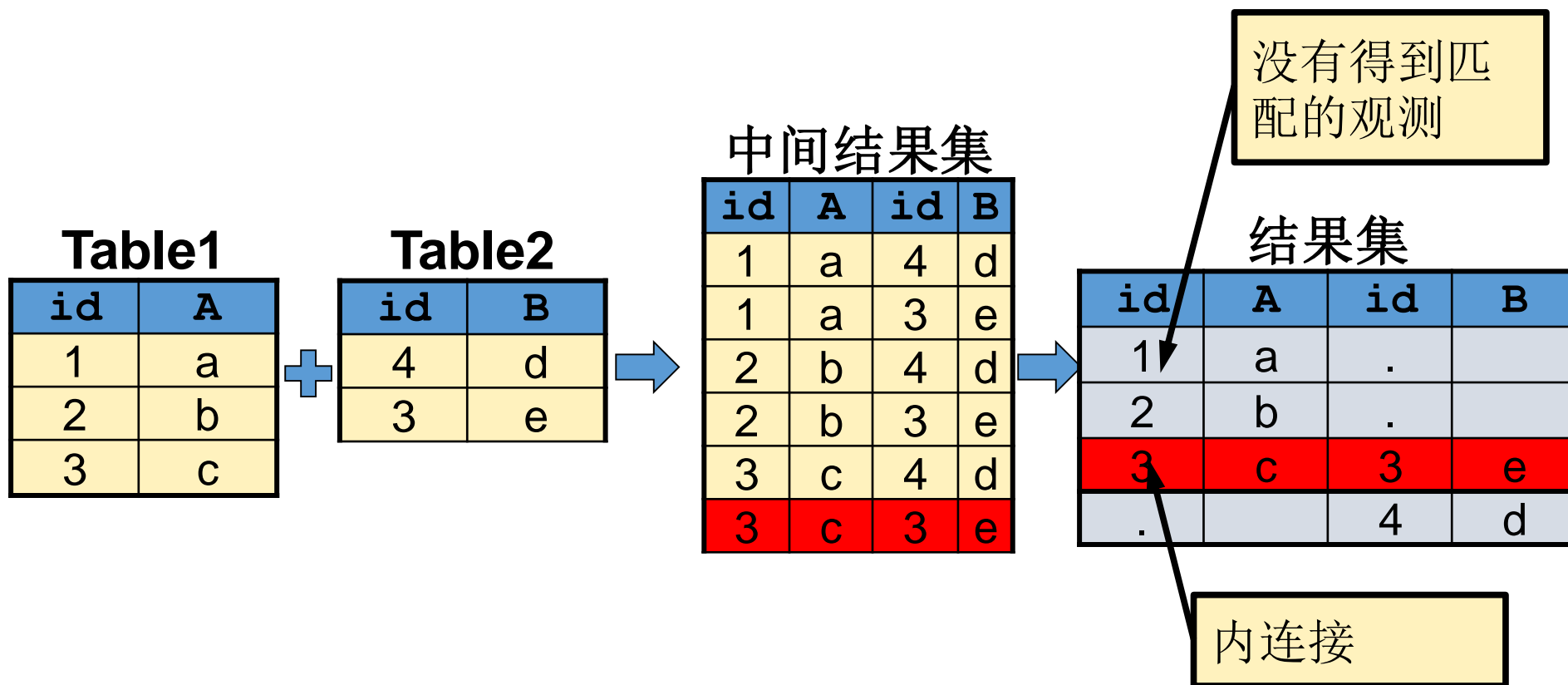
# 右连接

- 右连接等价于两部分的叠加：内连接+右表中没有匹配的观测。



# 全连接

- 全连接等价于三部分的叠加：内连接+左表中没有匹配的观测+右表中没有匹配的观测。



- 当Table1的观测所组成的集合是Table2的真子集，请说出以下哪个连接操作是等价的？
- A. 内连接
  - B. 左连接
  - C. 右连接
  - D. 全连接

## 5.1.4 SQL进行汇总

# 学习目标

- 在查询中分配一个分组变量
- 选择分析变量和需要计算的汇总统计量
- 过滤分组数据

- 根据销售数据（ SALE ）计算每年的销售和利润总额。

year	sum_sale	sum_profit
2010	132834	10587
2011	126953	10916
2012	127909	10785



上面操作的SQL语句如下：

```
select year,  
       sum(sale) as sum_sale,  
       sum(profit) as sum_profit  
from sale  
group by year
```

- 汇总函数一般和Group by语句同时使用；
- Group by语句出现的变量必须出现在Select语句的最前面，在Oracle 的SQL语句中，不允许Select语句中出现没有在Group by语句出现的变量，也不参与汇总的变量。R中sqldf由于是一个不完善的SQL语言，缺乏这种语法纠错机制，因此需要我们牢记正确的语法。

- 汇总函数也叫做集函数，包括：
  - count ( [DISTINCT] <列名> ) 统计一列中值的个数
  - sum ( [DISTINCT] <列名> ) 计算一列值的总和
  - avg ( [DISTINCT] <列名> ) 计算一列的平均值
  - stdev ( [DISTINCT] <列名> ) 计算一列的标准差
  - max ( [DISTINCT] <列名> ) 计算一列的最大值
  - min ( [DISTINCT] <列名> ) 计算一列的最小值
- 例. 获得sale表的总行数。

```
select count(*) from sale;
```

- 我们想知道哪些地域这三年的平均销售额低于全国平均水平。

market	mavg_sale
北	31394.33
南	31753.67

## SQL语句如下：

1、获取这三年全国销售的平均值：

```
select avg(sale) as all_avg_sale from sale
```

2、计算每个地域三年销售的平均值，并与全国数据做比较：

```
select market, avg(sale) as mavg_sale  
      from sale  
      group by market  
      having mavg_sale < 32308  
      order by mavg_sale
```

- 与上一条语句相比多了**Having**语句，该语句语法和**Where**语句基本一样，但是只能针对汇总语句生成的变量进行筛选。

使用嵌套语句（子查询）完成：

```
select market, avg(sale) as mavg_sale  
      from sale  
      group by market  
      having mavg_sale < (select avg(sale) as  
all_avg_sale from sale)  
      order by mavg_sale
```

- 在SQL中可以进行多层嵌套，并且语句从最深的一层开始执行。
- 比如上例中有两层SQL语句，having语句中的select语句为第二层，是最深的，要先运行，其返回的结果为32308，其作为having语句中逻辑判断的常数，这样外层的SQL就可以执行了。

## 5.2 数据清洗

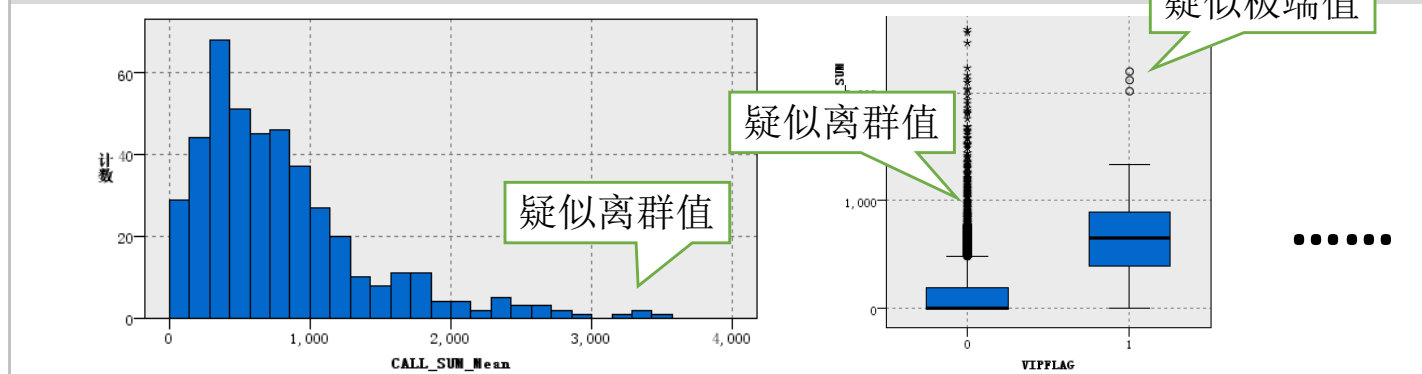
# 发现数据问题类型

- 脏数据或数据不正确
  - 比如 '0' 代表真实的0，还是代表缺失；Age = -2003
- 数据不一致
  - 比如收入单位是万元，利润单位是元，或者一个单位是美元，一个是人民币
- 数据重复
  - 这个问题在前面已经解决
- 缺失值
- 离群值

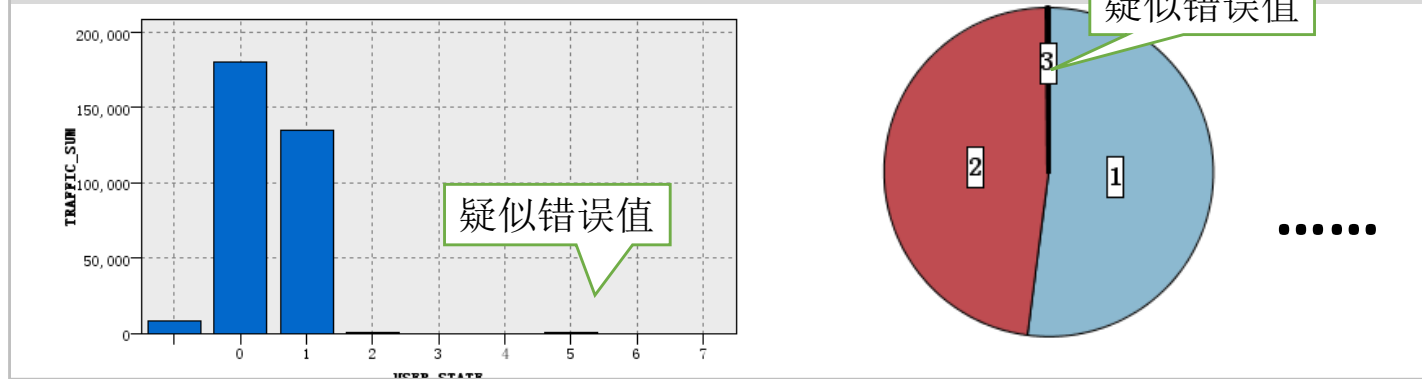
# 数据探索识别噪声

- 利用图形可以直观快速地对数据进行初步分析：
  - 直方图、饼图、条形图、折线图、散点图等

连续型变量:



离散型变量:





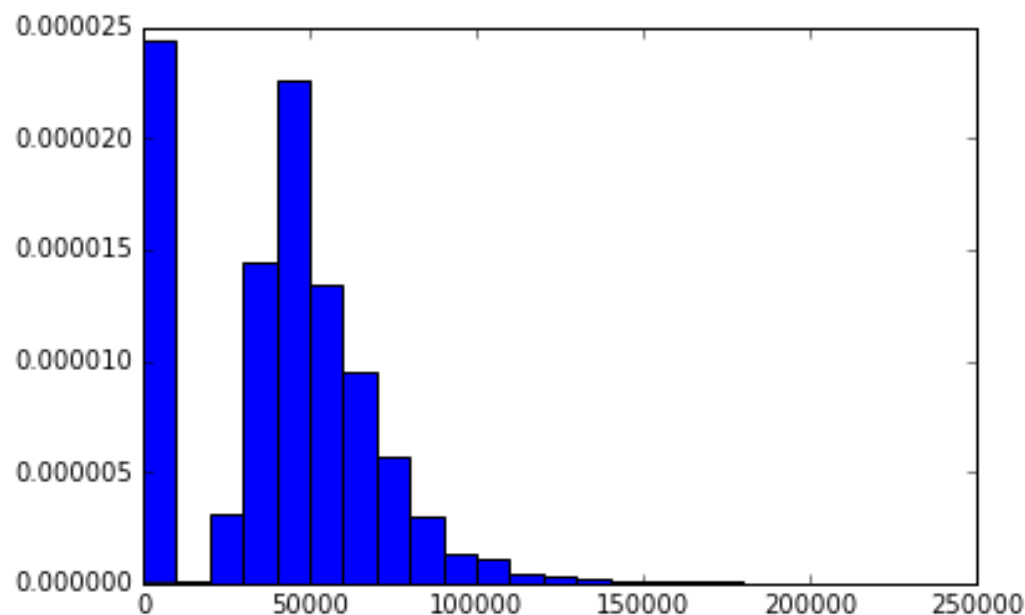
## 5.2.1 错误值处理

## 通过图形进行探索

发现错误值只能通过描述性统计的方法，逐一核实每个变量是否有问题，比如‘0’代表真实的0，还是代表缺失？


外呼营销数据

（teleco\_camp\_orig）的当地人均收入（AvgIncome），出现了大量0值，我们有理由怀疑是错误值。可以使用缺失值替代，然后再用缺失值填补的方法处理。



# 处理错误值

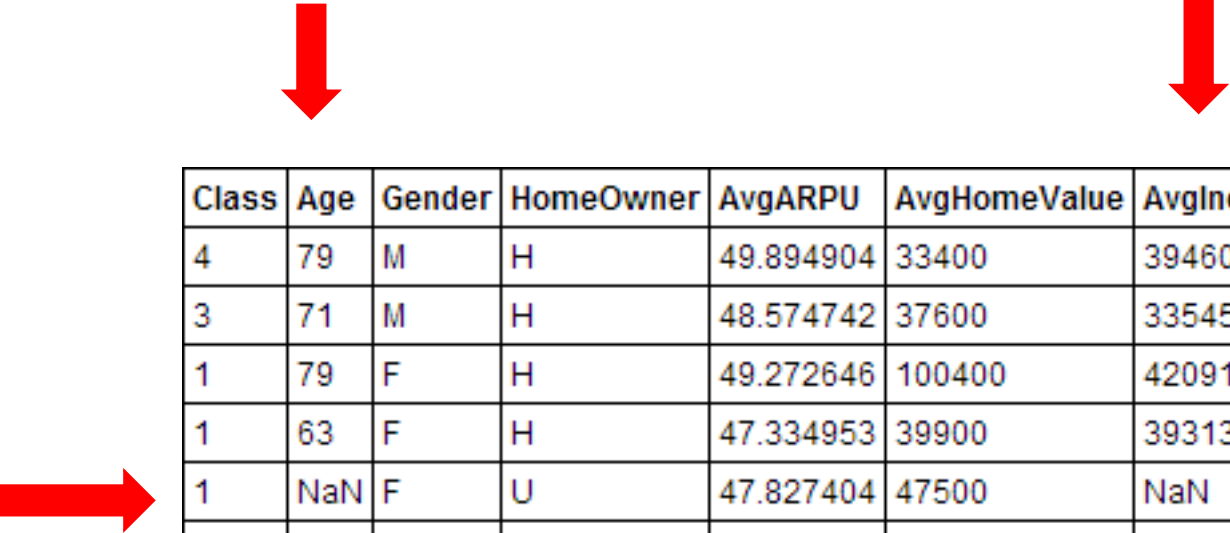
- 修正
  - 补充正确信息
  - 对照其他信息源
  - 视为空值
- 删除
  - 删除记录
  - 删除字段



明确错误原因很关键！

## 5.2.2 缺失值处理

# 发现缺失值



Class	Age	Gender	HomeOwner	AvgARPU	AvgHomeValue	AvgIncome
4	79	M	H	49.894904	33400	39460
3	71	M	H	48.574742	37600	33545
1	79	F	H	49.272646	100400	42091
1	63	F	H	47.334953	39900	39313
1	NaN	F	U	47.827404	47500	NaN
2	81	M	U	48.673449	53000	49487
2	NaN	F	U	48.560389	91000	NaN
3	69	F	H	49.644237	66300	49047

首选基于业务的填补方法，其次根据单变量分析进行填补，多重插补进行所有变量统一填补的方法只有在粗略清洗时才会使用。

- 缺失值少于20%
  - 连续变量使用均值或中位数填补。
  - 分类变量不需要填补，单算一类即可，或者用众数填补
- 缺失值在20%-80%
  - 填补方法同上
  - 另外每个有缺失值的变量生成一个指示哑变量，参与后续的建模
- 缺失值在大于80%
  - 每个有缺失值的变量生成一个指示哑变量，参与后续的建模，原始变量不使用。

## 填补 + 指示变量

不完整数据

34
63
.
22
26
54
18
.
47
20

填补后的变量

34
63
30
22
26
54
18
30
49
20

缺失值指示变量

0
0
1
0
0
0
0
1
0
0

Median = 30

# 处理缺失值示例

ID	gender	年龄	教育程度	所在区域	峰值月时长	营销次数
1	0	22	3	郊区	1232	2
2	1	18	3	市区	522	2
3		45	3	市区	845	1
4	1		2		1321	1
5	0	15	2	市区	611	0
6	1	22	0		967	1
7	1	25	1	郊区	662	0
8				市区	10710	
9	0	27		市区	996	1
10	0	30	3	郊区	422	2
11	0	18	3		776	

对比其它数据来源获取

连续变量可填均值/中位数

离散变量可填众数

缺失过多直接删除

可填“未知”并增加指示变量

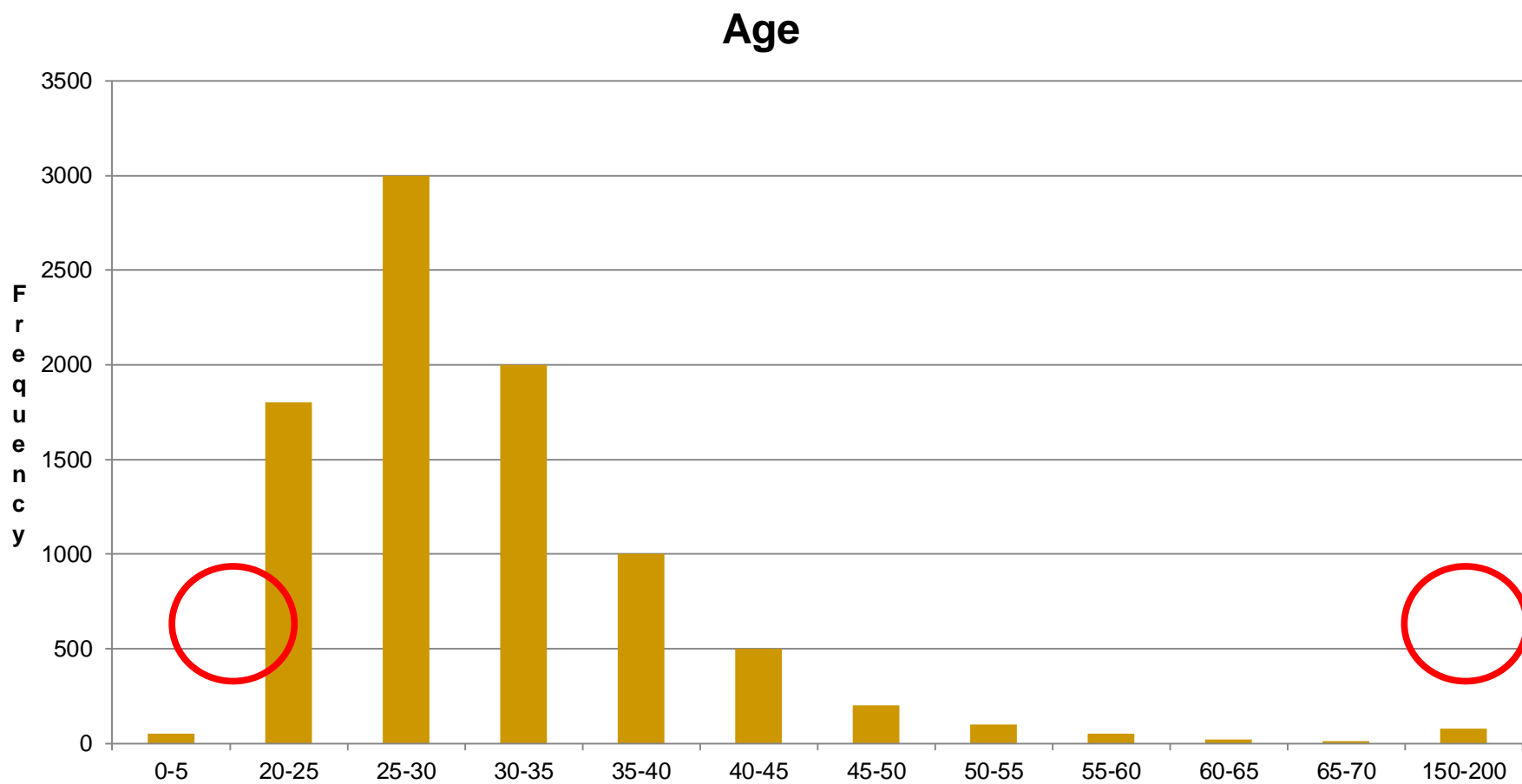
分类建模，聚类均值



## 5.2.3 噪声值处理

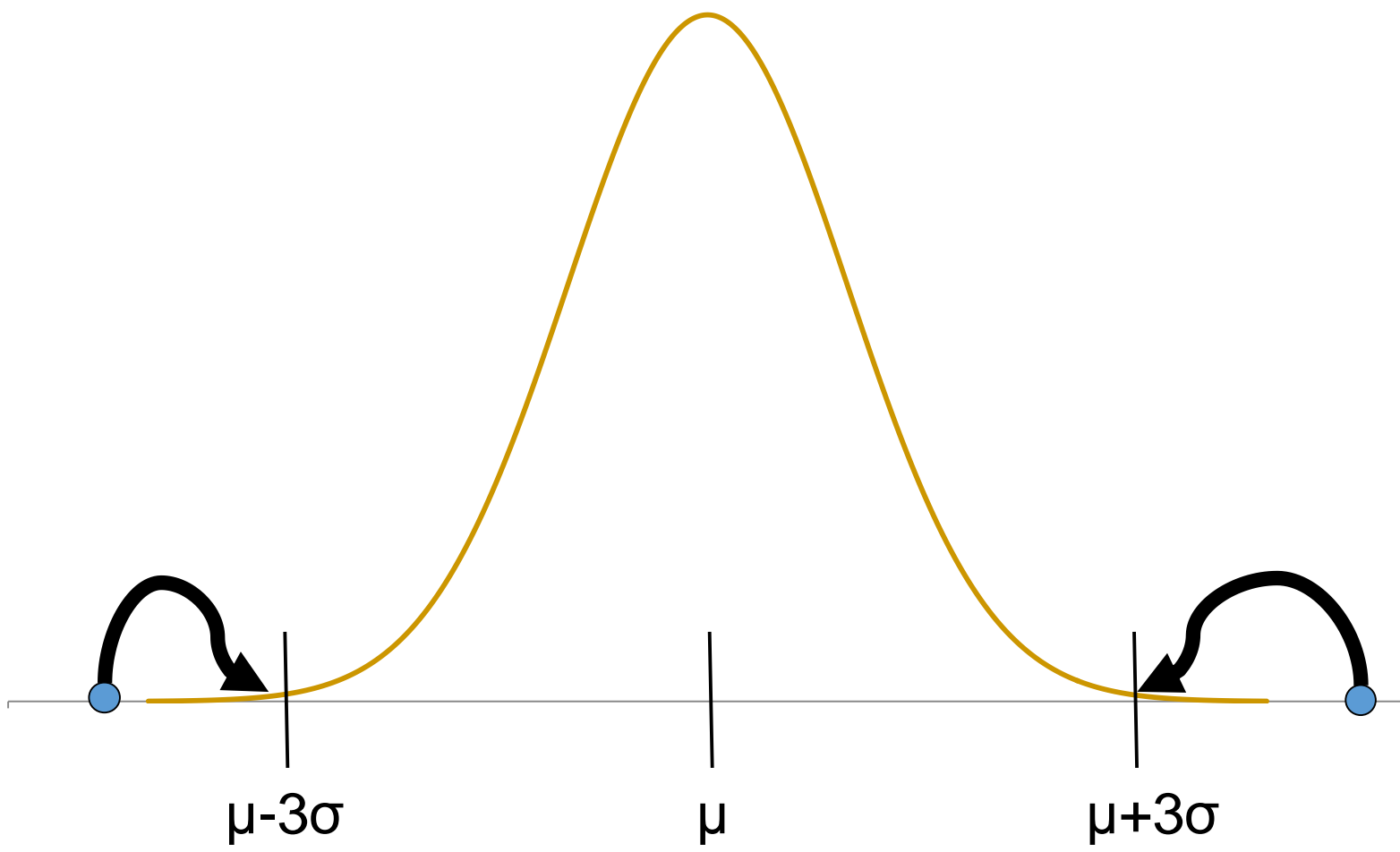


# 单变量离群值发现



# 单变量离群值发现

- 极端值
  - 设置标准，如：5倍标准差之外的数据
  - 极值有时意味着错误，应重新理解数据，例如：特殊用户的超大额消费
- 离群值
  - 平均值法：平均值 $\pm n$ 倍标准差之外的数据
    - 建议的临界值：
      - $|SR| > 2$ ，用于观察值较少的数据集
      - $|SR| > 3$ ，用于观察值较多的数据集
  - 四分位数法：
    - $IQR = Q3 - Q1$
    - $Q1 - 1.5 \times IQR \sim Q3 + 1.5 \times IQR$  \* 更适用于对称分布的数据



- 分箱方法通过考察数据的“近邻”来光滑有序数据的值。有序值分布到一些桶或箱中。
- 等深分箱：每个分箱中的样本量一致；
- 等宽分箱：每个分箱中的取值范围一致。

比如价格排序后数据：4, 8, 15, 21, 21, 24, 25, 28, 34

划分为（等深）箱：

- 箱1：4, 8, 15
- 箱2：21, 21, 24
- 箱3：25, 28, 34

划分为（等宽）箱：

- 箱1：4, 8
- 箱2：15, 21, 21, 24
- 箱3：25, 28, 34

—— 秦路主讲 ——  
**七周成为数据分析师**  
七周为期，Get一条数据分析师职业黄金通道！



—— Python ——  
**数据分析与挖掘**  
集Python爬虫、数据采集、数据处理、数据分析与数据挖掘于一体，打造Python全栈工程师  
主讲老师: 韦玮  
VIP会员群+在线答疑+录播复习+1年反复观看



**案例为师, 实战为王**  
开启Python机器学习之路  
科学规划全套课程体系，从入门到进阶，从理论到技巧，嵌入丰富课程案例讲解，逐步推进  
讲师: 唐宇迪 深度学习领域多年一线实践研究专家



**独一无二的  
数据仓库**建模指南系列教程升级版  
• 从企业视角进行数据规划以及数据仓库模型的搭建  
• 高质量的数据库模型和技巧，以及丰富的例子  
• 数据仓库架构理论和实践要领  
资深讲师: BAO胖子 15年+BI从业经验  
涉足电力、快消品、医药、信息服务行业的BI老兵



**业务知识一站通**  
技术+业务，挣钱有门路！  
—— 讲师: 陈文 ——



自己动手 丰衣足食  
**Python3网络爬虫实战案例**  
— 循序渐进，案例为王，诠释全面，思路制胜 —  
讲师: 崔庆才 北航硕士，百万级热度爬文博主



讲师 丘祐玮  
**人人都爱数据科学家**  
Python数据科学精华实战课程



**数据分析报告制作**  
秘籍升级版  
讲师: 陈丹奕 知乎大神，前百度资深数据分析师



**先机致胜 破冰AI**  
—— 深度学习模型/框架与实战 ——  
讲师: 唐宇迪 同济大学硕士  
深度学习领域多年一线实践研究专家



BI、商业智能  
数据挖掘 大数据  
数据分析师  
R语言 Python  
机器学习  
深度学习  
人工智能  
Hive Hadoop  
Tableau  
BIEE ETL  
数据科学家  
PowerBI