tidy\_data

chuqi

2025-09-23

示例 我将继续使用与数据导入和数据处理相同的仓库/项目，但会创建一个新的.Rmd文件用于整理。同时，我将加载一些相关包，并限制tibble中打印的行数。

library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.4 ✔ readr 2.1.5  
## ✔ forcats 1.0.0 ✔ stringr 1.5.1  
## ✔ ggplot2 3.5.2 ✔ tibble 3.3.0  
## ✔ lubridate 1.9.4 ✔ tidyr 1.3.1  
## ✔ purrr 1.1.0   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

options(tibble.print\_min = 5)

## pivot\_longer

pivot\_longer(…) 这是 tidyr 包中的一个函数，用于将数据框从“宽格式（wide format）”转换为“长格式（long format）”。

在数据导入过程中，我们使用haven包从.sas7bdat文件加载了PULSE生物标志物数据集。 现在让我们重新加载这些数据并深入分析：

pulse\_df =   
 haven::read\_sas("./data/public\_pulse\_data.sas7bdat") |>  
 janitor::clean\_names()  
  
pulse\_df

## # A tibble: 1,087 × 7  
## id age sex bdi\_score\_bl bdi\_score\_01m bdi\_score\_06m bdi\_score\_12m  
## <dbl> <dbl> <chr> <dbl> <dbl> <dbl> <dbl>  
## 1 10003 48.0 male 7 1 2 0  
## 2 10015 72.5 male 6 NA NA NA  
## 3 10022 58.5 male 14 3 8 NA  
## 4 10026 72.7 male 20 6 18 16  
## 5 10035 60.4 male 4 0 1 2  
## # ℹ 1,082 more rows

基于对整洁数据的新认识，我们很快发现了一个问题：BDI评分分散在四个列中，分别对应四个观察时间点。 我们可以使用pivot\_longer函数解决这个问题：

pulse\_tidy\_df =   
 pivot\_longer(  
 pulse\_df, # 原始数据框  
 bdi\_score\_bl:bdi\_score\_12m, # 选取要转换的列，从 bdi\_score\_bl 到 bdi\_score\_12m  
 names\_to = "visit", # 新建一列名为 visit，用来存放原来的列名  
 values\_to = "bdi" # 新建一列名为 bdi，用来保存原来的数值（如 12、15 等）  
 )  
  
pulse\_tidy\_df

## # A tibble: 4,348 × 5  
## id age sex visit bdi  
## <dbl> <dbl> <chr> <chr> <dbl>  
## 1 10003 48.0 male bdi\_score\_bl 7  
## 2 10003 48.0 male bdi\_score\_01m 1  
## 3 10003 48.0 male bdi\_score\_06m 2  
## 4 10003 48.0 male bdi\_score\_12m 0  
## 5 10015 72.5 male bdi\_score\_bl 6  
## # ℹ 4,343 more rows

看起来好多了！然而，现在visit有一个问题。原始列名虽然信息量很大，但我们可能不需要bdi\_score\_在每种情况下都保留前缀。我将使用一个附加选项来pivot\_longer解决这个问题：

pulse\_tidy\_df =   
 pivot\_longer(  
 pulse\_df, # 原始数据框，包含多个时间点的抑郁评分（bdi）  
 bdi\_score\_bl:bdi\_score\_12m, # 要转换的列范围，从基线（bl）到12个月（12m）  
 names\_to = "visit", # 创建新变量 visit，用来保存原来的列名（如 bdi\_score\_bl）  
 names\_prefix = "bdi\_score\_", # 去掉原列名前缀 "bdi\_score\_"，只保留后面的部分（如 bl、6m、12m）  
 values\_to = "bdi" # 创建新变量 bdi，用来存放原来各列中的数值  
 )  
  
pulse\_tidy\_df

## # A tibble: 4,348 × 5  
## id age sex visit bdi  
## <dbl> <dbl> <chr> <chr> <dbl>  
## 1 10003 48.0 male bl 7  
## 2 10003 48.0 male 01m 1  
## 3 10003 48.0 male 06m 2  
## 4 10003 48.0 male 12m 0  
## 5 10015 72.5 male bl 6  
## # ℹ 4,343 more rows

在前面，我保存了中间数据集，以便清晰地展示每个步骤。 虽然这在您尝试代码时可能有所帮助，但通常来说，这不是一个好习惯。 为了完成数据整理过程，还需要进行一些额外的转换，例如，为了在访问中保持一致性，将其更改为，以及转换bl为因子变量。 （您可能希望将其转换为数值变量，这可以通过调用 来完成 。）00mvisitvisitmutate

总之，下面的代码将导入、整理并将 PULSE 数据集转换为可用格式：

pulse\_df =   
 haven::read\_sas("./data/public\_pulse\_data.sas7bdat") |> # 读取 SAS 格式的数据文件  
 janitor::clean\_names() |> # 清理列名（全部转为小写、下划线格式，更易用）  
   
 pivot\_longer( # 宽转长，把多个列转为两列（visit 和 bdi）  
 bdi\_score\_bl:bdi\_score\_12m, # 选中从 bdi\_score\_bl 到 bdi\_score\_12m 的列  
 names\_to = "visit", # 将原来的列名存入新列 visit  
 names\_prefix = "bdi\_score\_", # 去掉列名前缀 "bdi\_score\_"，只留下 bl、6m、12m  
 values\_to = "bdi" # 将原列中的数值存入新列 bdi  
 ) |>  
   
 relocate(id, visit) |> # 调整列顺序，把 id 和 visit 移到最前面  
   
 mutate( # mutate：变出新列、改已有列  
 visit = replace(visit, visit == "bl", "00m"), # 将 visit 中的 "bl"（基线）替换为 "00m"  
 visit = factor(visit) # 把 visit 列转换为因子（方便后续排序或分组分析）  
 )  
  
print(pulse\_df, n = 12) # 打印 pulse\_df 的前 12 行数据

## # A tibble: 4,348 × 5  
## id visit age sex bdi  
## <dbl> <fct> <dbl> <chr> <dbl>  
## 1 10003 00m 48.0 male 7  
## 2 10003 01m 48.0 male 1  
## 3 10003 06m 48.0 male 2  
## 4 10003 12m 48.0 male 0  
## 5 10015 00m 72.5 male 6  
## 6 10015 01m 72.5 male NA  
## 7 10015 06m 72.5 male NA  
## 8 10015 12m 72.5 male NA  
## 9 10022 00m 58.5 male 14  
## 10 10022 01m 58.5 male 3  
## 11 10022 06m 58.5 male 8  
## 12 10022 12m 58.5 male NA  
## # ℹ 4,336 more rows

现在我们的状况还不错:-)。

##学习评估：

在窝仔数据中，变量gd0\_weight和gd18\_weight分别表示母鼠在妊娠第0天和第18天的体重。 编写一个数据清理链，仅保留litter\_number和这些列；生成新的变量gd和weight； 并创建和的gd数值变量（对于最后一部分，你可能想使用……）。 这个版本“整洁”吗？018recode

## 解决方案

pivot\_wider

我们一直以来都专注于整理数据，但我们也承认，有时不整洁的数据更利于人类阅读。 因此，我们将稍微偏离主题，谈谈如何整理你的整洁数据。

下面的代码创建了一个整洁的数据集，可用于分析。这是进行进一步分析或可视化的正确格式，但不利于人类读者进行快速比较。

analysis\_result = # 创建一个新的数据框（tibble），赋值给变量 analysis\_result  
 tibble( # 使用 tibble() 函数，生成一个简洁的表格数据结构  
 group = c("treatment", "treatment", "placebo", "placebo"), # 定义一列 group，包含4个元素，表示组别  
 time = c("pre", "post", "pre", "post"), # 定义一列 time，表示时间点（干预前后）  
 mean = c(4, 8, 3.5, 4) # 定义一列 mean，表示对应组别和时间的平均值  
 )  
  
analysis\_result # 输出数据框内容，查看创建的 tibble

## # A tibble: 4 × 3  
## group time mean  
## <chr> <chr> <dbl>  
## 1 treatment pre 4   
## 2 treatment post 8   
## 3 placebo pre 3.5  
## 4 placebo post 4

相同数据的另一种呈现方式可能是将组放在行中，将时间放在列中，并将平均值放在表格单元格中。这显然不够整洁； 为了实现这一点，我们需要使用 pivot\_wider，它是pivot\_longer的逆：

pivot\_wider(  
 analysis\_result, # 输入数据框，长格式的分析结果  
 names\_from = "time", # 将 "time" 列的不同值（如 "pre", "post"）变成新的列名  
 values\_from = "mean" # 新列对应的值取自 "mean" 列  
)

## # A tibble: 2 × 3  
## group pre post  
## <chr> <dbl> <dbl>  
## 1 treatment 4 8  
## 2 placebo 3.5 4

现在我们已经基本完成了 - 在某些情况下，您可能会使用它 select来重新排序列，并且（取决于您的目标）使用它 knitr::kable()来生成更美观的阅读表格。

## Binding rows

我们研究了单表非整洁数据，但非整洁性通常源于分散在多个表中的相关数据。在最简单的情况下，这些表基本相同，可以堆叠起来生成一个整洁的数据集。这就是中的设置LotR\_words.xlsx，其中三部曲中每部电影中不同种族和性别的字数分布在不同的数据矩形中（这些数据基于此示例）。

为了生成所需的整洁数据集，我们首先需要读取每个表并进行一些清理。

fellowship\_ring =   
 readxl::read\_excel("./data/LotR\_Words.xlsx", range = "B3:D6") |>  
 mutate(movie = "fellowship\_ring")  
  
two\_towers =   
 readxl::read\_excel("./data/LotR\_Words.xlsx", range = "F3:H6") |>  
 mutate(movie = "two\_towers")  
  
return\_king =   
 readxl::read\_excel("./data/LotR\_Words.xlsx", range = "J3:L6") |>  
 mutate(movie = "return\_king")

这里需要为每个数据框添加一个变量来指示电影；这些信息存储在原始电子表格的其他位置。 顺便说一句，上面的三个代码片段除了范围和电影名称之外基本相同——稍后我们将看到一种更好的方法来处理这种情况， 即编写我们自己的函数，但目前这种方法已经可以正常工作。

一旦每个表都准备好了，我们就可以使用它们堆叠起来 bind\_rows并整理结果：

lotr\_tidy =   
 bind\_rows(fellowship\_ring, two\_towers, return\_king) |> # 合并三个数据框，按行绑定（堆叠）  
 janitor::clean\_names() |> # 清理列名，转换为小写加下划线格式  
 relocate(movie) |> # 将 movie 列移动到最前面  
 pivot\_longer(  
 female:male, # 将 female 到 male 这两列转换成长格式  
 names\_to = "gender", # 新建变量 gender，存放原列名（female/male）  
 values\_to = "words" # 新建变量 words，存放原列对应的数值  
 ) |>   
 mutate(race = str\_to\_lower(race)) |> # 把 race 列的字符串转换成小写  
 select(movie, everything()) # 把 movie 列放到最前，其他列保持原顺序  
  
lotr\_tidy # 查看转换后的整洁数据框

## # A tibble: 18 × 4  
## movie race gender words  
## <chr> <chr> <chr> <dbl>  
## 1 fellowship\_ring elf female 1229  
## 2 fellowship\_ring elf male 971  
## 3 fellowship\_ring hobbit female 14  
## 4 fellowship\_ring hobbit male 3644  
## 5 fellowship\_ring man female 0  
## 6 fellowship\_ring man male 1995  
## 7 two\_towers elf female 331  
## 8 two\_towers elf male 513  
## 9 two\_towers hobbit female 0  
## 10 two\_towers hobbit male 2463  
## 11 two\_towers man female 401  
## 12 two\_towers man male 3589  
## 13 return\_king elf female 183  
## 14 return\_king elf male 510  
## 15 return\_king hobbit female 2  
## 16 return\_king hobbit male 2673  
## 17 return\_king man female 268  
## 18 return\_king man male 2459

有了这种形式的数据，我们就可以更轻松地对电影之间进行比较、对三部曲中的种族进行汇总以及执行其他分析。

## Joining datasets

数据可能分布在多个相关表中，在这种情况下，需要在分析之前对它们进行合并或连接。 我们将只关注合并两个表的问题； 合并三个或更多表的步骤将使用相同的思路逐步完成。

有四种主要方式连接数据框x和y：

内部：保留同时出现在数据中的x和y 左：保留出现在x 右：保留出现在y 完整：保留出现在数据中的x和y 左连接是最常见的，因为它们将较小表中的数据添加y到较大表中，x而不会从中删除任何内容x。

举例来说，考虑FAS\_pups.csv 和中的数据表FAS\_litters.csv，它们通过Litter Number变量关联。 前者包含每只幼崽独有的数据，后者包含每窝幼崽独有的数据。我们可以使用左连接将窝仔数据与幼崽数据合并； 这样做可以保留每只幼崽的数据，并在新列中添加数据。

（在重新审视这个例子时，请查看group窝仔数据集中的变量： 它编码了剂量和治疗日期！我们将在处理流程中修复这个混乱的部分。 我还将解决我的一个烦恼，即将性别编码为一个模棱两可的数字变量。）

pup\_df =   
 read\_csv( # 读取 CSV 文件 FAS\_pups.csv  
 "./data/FAS\_pups.csv", # 文件路径  
 skip = 3, # 跳过前三行（可能是标题或说明）  
 na = c("NA", "", ".")) |> # 将 "NA"、空字符串和 "." 识别为缺失值  
 janitor::clean\_names() |> # 清理列名，转为小写加下划线格式  
 mutate(sex = recode(sex, `1` = "male", `2` = "female")) # 把 sex 列中数字转换为文字

## Rows: 313 Columns: 6  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (1): Litter Number  
## dbl (5): Sex, PD ears, PD eyes, PD pivot, PD walk  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

litter\_df =   
 read\_csv( # 读取 CSV 文件 FAS\_litters.csv  
 "./data/FAS\_litters.csv", # 文件路径  
 na = c("NA", ".", "")) |> # 将 "NA"、"."、空字符串识别为缺失值  
 janitor::clean\_names() |> # 清理列名，转为小写加下划线格式  
 relocate(litter\_number) |> # 将 litter\_number 列移到最前面  
 separate(group, into = c("dose", "day\_of\_tx"), sep = 3) |> # 按第3个字符拆分 group 列，生成 dose 和 day\_of\_tx 两列  
 mutate( # 创建或修改列  
 wt\_gain = gd18\_weight - gd0\_weight, # 计算体重增加：第18天体重减去第0天体重  
 dose = str\_to\_lower(dose) # 将 dose 列转换为小写  
 )

## Rows: 49 Columns: 8  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (2): Group, Litter Number  
## dbl (6): GD0 weight, GD18 weight, GD of Birth, Pups born alive, Pups dead @ ...  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

join them!

fas\_df =   
 left\_join(pup\_df, litter\_df, by = "litter\_number") |> # 按 litter\_number 列左连接 pup\_df 和 litter\_df，合并两个数据框  
 arrange(litter\_number) |> # 按 litter\_number 升序排列行  
 relocate(litter\_number, dose) # 将 litter\_number 和 dose 列移到最前面  
  
fas\_df # 查看合并整理后的数据框

## # A tibble: 313 × 15  
## litter\_number dose sex pd\_ears pd\_eyes pd\_pivot pd\_walk day\_of\_tx  
## <chr> <chr> <chr> <dbl> <dbl> <dbl> <dbl> <chr>   
## 1 #1/2/95/2 con male 5 13 7 9 7   
## 2 #1/2/95/2 con male 5 13 8 10 7   
## 3 #1/2/95/2 con female 4 13 7 9 7   
## 4 #1/2/95/2 con female 4 13 7 10 7   
## 5 #1/2/95/2 con female 5 13 8 10 7   
## # ℹ 308 more rows  
## # ℹ 7 more variables: gd0\_weight <dbl>, gd18\_weight <dbl>, gd\_of\_birth <dbl>,  
## # pups\_born\_alive <dbl>, pups\_dead\_birth <dbl>, pups\_survive <dbl>,  
## # wt\_gain <dbl>

我们在连接操作中明确指定了键。 默认情况下， \*\_join连接操作中的函数dplyr会尝试根据待连接数据集中的变量名来确定键。 这通常足够了，但并非总是如此，额外的步骤来明确键将有助于您和其他人阅读您的代码。

请注意，连接对于操作员来说并不是特别合适， |>因为它从根本上来说是非线性的：两个独立的数据集结合在一起，而不是逐步处理单个数据集。

最后一点，这些\*\_join函数与 SQL 语法非常相关，但强调数据分析中常见的操作。

##学习评估： 此 zip 文件中的数据集包含本课程往年调查问卷的去识别化回复。两个数据集都包含唯一的学生标识符；第一个数据集包含关于操作系统的问题的回复，第二个数据集包含关于学位课程和 Git 经验的问题的回复。请编写一个代码块，导入并清理这两个数据集，然后将它们连接起来。

##解决方案 我将两个数据集都放在了data我的 repo / 项目的目录中。下面的代码导入了这两个数据集，清理了变量名，并使用left\_join、 inner\_join和连接这两个数据集anti\_join。

surv\_os =   
 read\_csv("data/surv\_os.csv") |> # 读取 surv\_os.csv 文件  
 janitor::clean\_names() |> # 清理列名为小写加下划线格式  
 rename(id = what\_is\_your\_uni, # 重命名列：将 "what\_is\_your\_uni" 改为 "id"  
 os = what\_operating\_system\_do\_you\_use) # 重命名列："what\_operating\_system\_do\_you\_use" 改为 "os"

## Rows: 173 Columns: 2  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (2): What is your UNI?, What operating system do you use?  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

surv\_pr\_git =   
 read\_csv("data/surv\_program\_git.csv") |> # 读取 surv\_program\_git.csv 文件  
 janitor::clean\_names() |> # 清理列名  
 rename( # 重命名多列  
 id = what\_is\_your\_uni, # "what\_is\_your\_uni" 改为 "id"  
 prog = what\_is\_your\_degree\_program, # "what\_is\_your\_degree\_program" 改为 "prog"  
 git\_exp = which\_most\_accurately\_describes\_your\_experience\_with\_git)#改为 "git\_exp"

## Rows: 135 Columns: 3  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (3): What is your UNI?, What is your degree program?, Which most accurat...  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

left\_join(surv\_os, surv\_pr\_git) # 左连接，保留 surv\_os 所有行，匹配 surv\_pr\_git 的信息

## Joining with `by = join\_by(id)`

## Warning in left\_join(surv\_os, surv\_pr\_git): Detected an unexpected many-to-many relationship between `x` and `y`.  
## ℹ Row 7 of `x` matches multiple rows in `y`.  
## ℹ Row 66 of `y` matches multiple rows in `x`.  
## ℹ If a many-to-many relationship is expected, set `relationship =  
## "many-to-many"` to silence this warning.

## # A tibble: 175 × 4  
## id os prog git\_exp   
## <chr> <chr> <chr> <chr>   
## 1 student\_87 <NA> MS Pretty smooth: needed some work to connect Git, …  
## 2 student\_106 Windows 10 Other Pretty smooth: needed some work to connect Git, …  
## 3 student\_66 Mac OS X MPH Smooth: installation and connection with GitHub …  
## 4 student\_93 Windows 10 MS Smooth: installation and connection with GitHub …  
## 5 student\_99 Mac OS X MS Smooth: installation and connection with GitHub …  
## # ℹ 170 more rows

inner\_join(surv\_os, surv\_pr\_git) # 内连接，只保留两个表都存在的 id 行

## Joining with `by = join\_by(id)`

## Warning in inner\_join(surv\_os, surv\_pr\_git): Detected an unexpected many-to-many relationship between `x` and `y`.  
## ℹ Row 7 of `x` matches multiple rows in `y`.  
## ℹ Row 66 of `y` matches multiple rows in `x`.  
## ℹ If a many-to-many relationship is expected, set `relationship =  
## "many-to-many"` to silence this warning.

## # A tibble: 129 × 4  
## id os prog git\_exp   
## <chr> <chr> <chr> <chr>   
## 1 student\_87 <NA> MS Pretty smooth: needed some work to connect Git, …  
## 2 student\_106 Windows 10 Other Pretty smooth: needed some work to connect Git, …  
## 3 student\_66 Mac OS X MPH Smooth: installation and connection with GitHub …  
## 4 student\_93 Windows 10 MS Smooth: installation and connection with GitHub …  
## 5 student\_99 Mac OS X MS Smooth: installation and connection with GitHub …  
## # ℹ 124 more rows

anti\_join(surv\_os, surv\_pr\_git) # 反连接，保留 surv\_os 中不在 surv\_pr\_git 的 id 行

## Joining with `by = join\_by(id)`

## # A tibble: 46 × 2  
## id os   
## <chr> <chr>   
## 1 student\_86 Mac OS X   
## 2 student\_91 Windows 10  
## 3 student\_24 Mac OS X   
## 4 student\_103 Mac OS X   
## 5 student\_163 Mac OS X   
## # ℹ 41 more rows

anti\_join(surv\_pr\_git, surv\_os) # 反连接，保留 surv\_pr\_git 中不在 surv\_os 的 id 行

## Joining with `by = join\_by(id)`

## # A tibble: 15 × 3  
## id prog git\_exp   
## <chr> <chr> <chr>   
## 1 <NA> MPH "Pretty smooth: needed some work to connect Git, GitHub, an…  
## 2 student\_17 PhD "Pretty smooth: needed some work to connect Git, GitHub, an…  
## 3 <NA> MPH "Pretty smooth: needed some work to connect Git, GitHub, an…  
## 4 <NA> MPH "Pretty smooth: needed some work to connect Git, GitHub, an…  
## 5 <NA> MS "Pretty smooth: needed some work to connect Git, GitHub, an…  
## 6 student\_53 MS "Pretty smooth: needed some work to connect Git, GitHub, an…  
## 7 <NA> MS "Smooth: installation and connection with GitHub was easy"   
## 8 student\_80 PhD "Pretty smooth: needed some work to connect Git, GitHub, an…  
## 9 student\_16 MPH "Smooth: installation and connection with GitHub was easy"   
## 10 student\_98 MS "Smooth: installation and connection with GitHub was easy"   
## 11 <NA> MS "Pretty smooth: needed some work to connect Git, GitHub, an…  
## 12 <NA> MS "What's \"Git\" ...?"   
## 13 <NA> MS "Smooth: installation and connection with GitHub was easy"   
## 14 <NA> MPH "Pretty smooth: needed some work to connect Git, GitHub, an…  
## 15 <NA> MS "Pretty smooth: needed some work to connect Git, GitHub, an…

left\_join和inner\_join都给出了关于“多对多”关系的警告。这表明匹配的变量不唯一，应该对警告进行调查。 在下面的代码中，我们来分析一下“第 7 行中x 与 中的多行匹配y”这个警告。

surv\_os |> slice(7)

## # A tibble: 1 × 2  
## id os   
## <chr> <chr>   
## 1 student\_15 Windows 10

surv\_pr\_git |> filter(id == "student\_15")

## # A tibble: 2 × 3  
## id prog git\_exp   
## <chr> <chr> <chr>   
## 1 student\_15 MPH Pretty smooth: needed some work to connect Git, GitHub, and …  
## 2 student\_15 MPH Pretty smooth: needed some work to connect Git, GitHub, and …

##关于名字的简要说明 有一段时间，人们使用gatherand spread而不是pivot\_longer and pivot\_wider。 新功能的更新是有充分理由的；gatherandspread仍将存在，但随着时间的推移，它们将变得不那么常见，您可能永远不会再看到它们。