# 玩转数据 120 题——R 语言 tidyverse 版本 (2023 版)

张敬信

2023-03-25

**关于作者：**

- 张敬信，哈尔滨商业大学，数学与应用数学，副教授
- 热爱学习，热爱编程，热爱 R 语言
- 我的代表作《R 语言编程：基于 tidyverse》已于 2023 年 2 月正式上市，京东、天猫、当当等各大平台有售：


- 也有免费学习资源：课件和鲸在线版（可调试代码），知乎交流平台，欢迎您的阅读品鉴、交流讨论！
- 该书的 QQ 读者 1 群、2 群：875664831、222427909, 交流、答疑，欢迎您的加入！

**玩转数据 120 题**来自刘早起的 *Pandas 进阶修炼 120 题*，涵盖了数据处理、计算、可视化等常用操作，希望通过 120 道精心挑选的习题吃透 pandas.

后来，中山大学博士陈熹提供了 R 语言版本。我[1][2][3]再来个更能体现 R 语言最新技术的 tidyverse 版本。

关于**更新版**：感谢 @ 鼠大米对部分解法不够 tidyverse 的题目，提供了新解法 (再加上我稍微修正)，主要是加入更好用的新函数 `slice_*()`。2023 年，部分题目又做了一些改进。

- 先加载包：

```
library(tidyverse)
```

## Part I 入门

**题目 1（创建数据框）：将下面的字典创建为 DataFrame**

`data = {"grammer": ["Python","C","Java","GO",np.nan,"SQL","PHP","Python"], "score": [1,2,np.nan,4,5,6,7,10]}`

**难度：** ⋆

**代码及运行结果：**

```
df = tibble(
  grammer = c("Python","C","Java","GO", NA,"SQL","PHP","Python"),
  score = c(1,2,NA,4,5,6,7,10))
df
```

```
## # A tibble: 8 x 2
##   grammer score
##   <chr>   <dbl>
## 1 Python      1
## 2 C           2
## 3 Java       NA
## 4 GO          4
## 5 <NA>        5
## 6 SQL         6
## # ... with 2 more rows
```

- **补充：**按行录入式创建数据框

```
df = tribble(
  ~ grammer, ~ score,
  "Python",  1,
```

[1]我的 Github: https://github.com/zhjx19

[2]我的知乎: https://www.zhihu.com/people/huc_zhangjingxin

[3]我的和鲸项目（包含本文件的在线可调试版）：https://www.heywhale.com/home/user/profile/5d4a42a636e903002c0e0c66/project

```
  "C",       2,
  "Java",    NA,
  "GO",      4,
  NA,        5,
  "SQL",     6,
  "PHP",     7,
  "Python",  10)
```

问题 **2**（筛选行）：提取含有字符串”**Python**” 的行

难度：⋆

代码及运行结果：

```
df %>%
  filter(grammer == "Python")
```

```
## # A tibble: 2 x 2
##   grammer score
##   <chr>   <dbl>
## 1 Python      1
## 2 Python     10
```

题目 **3**（查看列名）：输出 **df** 的所有列名

难度：⋆

代码及运行结果：

```
names(df)
```

```
## [1] "grammer" "score"
```

题目 **4**（修改列名）：修改第 **2** 列列名为”**popularity**”

难度：⋆⋆

代码及运行结果：

```
df = df %>%
  rename(popularity = score)
df
```

```
## # A tibble: 8 x 2
##   grammer popularity
##   <chr>        <dbl>
```

```
## 1 Python           1
## 2 C                2
## 3 Java            NA
## 4 GO               4
## 5 <NA>             5
## 6 SQL              6
## # ... with 2 more rows
```

题目 **5**（统计频数）：统计 **grammer** 列中每种编程语言出现的次数

难度：★★

代码及运行结果：

```
df %>%
  count(grammer)
```

```
## # A tibble: 7 x 2
##   grammer     n
##   <chr>   <int>
## 1 C           1
## 2 GO          1
## 3 Java        1
## 4 PHP         1
## 5 Python      2
## 6 SQL         1
## # ... with 1 more row
```

题目 **6**（缺失值处理）：将空值用上下值的平均值填充

难度：★★★

代码及运行结果：

```
df = df %>%
  mutate(popularity = zoo::na.approx(popularity))
df
```

```
## # A tibble: 8 x 2
##   grammer popularity
##   <chr>        <dbl>
## 1 Python           1
## 2 C                2
## 3 Java             3
```

```
## 4 GO                4
## 5 <NA>              5
## 6 SQL               6
## # ... with 2 more rows
```

注： tidyr 包提供了 `fill()` 函数，可以用前值或后值插补缺失值。

**题目 7（筛选行）：提取 popularity 列中值大于 3 的行**

难度：★★

代码及运行结果：

```
df %>%
  filter(popularity > 3)
```

```
## # A tibble: 5 x 2
##   grammer popularity
##   <chr>        <dbl>
## 1 GO               4
## 2 <NA>             5
## 3 SQL              6
## 4 PHP              7
## 5 Python          10
```

**题目 8（数据去重）：按 grammer 列进行去重**

难度：★★

代码及运行结果：

```
df %>%
  distinct(grammer, .keep_all = TRUE)
```

```
## # A tibble: 7 x 2
##   grammer popularity
##   <chr>        <dbl>
## 1 Python           1
## 2 C                2
## 3 Java             3
## 4 GO               4
## 5 <NA>             5
## 6 SQL              6
## # ... with 1 more row
```

**题目 9（数据计算）：计算 popularity 列平均值**

难度：★★

代码及运行结果：

```
df %>%
  summarise(avg = mean(popularity))
```

```
## # A tibble: 1 x 1
##     avg
##   <dbl>
## 1  4.75
```

**题目 10（格式转换）：将 grammer 列转换为序列**

难度：★

代码及运行结果：

```
df %>%
  pull(grammer)    # 或者 df$grammer
```

```
## [1] "Python" "C"      "Java"   "GO"     NA       "SQL"    "PHP"    "Python"
```

注：R 从数据框中提取出来就是字符向量。

**题目 11（数据保存）：将数据框保存为 Excel**

难度：★★

代码及运行结果：

```
writexl::write_xlsx(df, "data/filename.xlsx")
```

**题目 12（数据查看）：查看数据的行数列数**

难度：★

代码及运行结果：

```
dim(df)
```

```
## [1] 8 2
```

**题目 13（筛选行）：提取 popularity 列值大于 3 小于 7 的行**

难度：★★

代码及运行结果：

```
df %>%
  filter(popularity > 3 & popularity < 7)
```

```
## # A tibble: 3 x 2
##   grammer popularity
##   <chr>        <dbl>
## 1 GO               4
## 2 <NA>             5
## 3 SQL              6
```

**题目 14（调整列位置）：交互两列的位置**

难度：★★

代码及运行结果：

```
df %>%
  select(popularity, grammer)
```

```
## # A tibble: 8 x 2
##   popularity grammer
##        <dbl> <chr>
## 1          1 Python
## 2          2 C
## 3          3 Java
## 4          4 GO
## 5          5 <NA>
## 6          6 SQL
## # ... with 2 more rows
```

注：可配合 `everything()` 放置“其余列”，更强大的调整列位置的函数是 dplyr1.0 将提供的 `relocate()`.

**题目 15（筛选行）：提取 popularity 列最大值所在的行**

难度：★★

代码及运行结果：

```
df %>%
  slice_max(popularity, n = 1)
```

```
## # A tibble: 1 x 2
##   grammer popularity
##   <chr>        <dbl>
## 1 Python          10
```

```
# 或者
# df %>%
#  filter(popularity == max(popularity))
```

**题目 16（查看数据）：查看最后几行数据**

难度：⋆

代码及运行结果：

```
df %>%
  slice_tail(n = 6)   # 或者 tail(df, 6)
```

```
## # A tibble: 6 x 2
##   grammer popularity
##   <chr>        <dbl>
## 1 Java             3
## 2 GO               4
## 3 <NA>             5
## 4 SQL              6
## 5 PHP              7
## 6 Python          10
```

注：此外，dplyr 包还提供了 slice_head() 查看前 n 行或前某比例的行，slice_sample() 随机查看 n 行或某比例的行。

**题目 17（修改数据）：删除最后一行数据**

难度：⋆⋆

代码及运行结果：

```
df %>%
  slice(-n())
```

```
## # A tibble: 7 x 2
##   grammer popularity
##   <chr>        <dbl>
## 1 Python           1
## 2 C                2
## 3 Java             3
## 4 GO               4
## 5 <NA>             5
## 6 SQL              6
```

```
## # ... with 1 more row
```

题目 18（修改数据）：添加一行数据：**"Perl", 6**

难度：★★

代码及运行结果：

```
df %>%
  add_row(grammer = "Perl", popularity = 6)
```

```
## # A tibble: 9 x 2
##   grammer popularity
##   <chr>        <dbl>
## 1 Python           1
## 2 C                2
## 3 Java             3
## 4 GO               4
## 5 <NA>             5
## 6 SQL              6
## # ... with 3 more rows
```

```
# 或者
# df %>%
#   bind_rows(tibble(grammer = "Perl", popularity = 6))
```

题目 19（数据整理）：对数据按 **popularity** 列值从到大到小排序

难度：★★

代码及运行结果：

```
df %>%
  arrange(-popularity)
```

```
## # A tibble: 8 x 2
##   grammer popularity
##   <chr>        <dbl>
## 1 Python          10
## 2 PHP              7
## 3 SQL              6
## 4 <NA>             5
## 5 GO               4
## 6 Java             3
## # ... with 2 more rows
```

注：默认从小到大排序。

**题目 20（字符统计）：统计 grammer 列每个字符串的长度**

难度：★★

代码及运行结果：

```
df %>%
  mutate(strlen = str_length(grammer))
```

```
## # A tibble: 8 x 3
##   grammer popularity strlen
##   <chr>        <dbl>  <int>
## 1 Python           1      6
## 2 C                2      1
## 3 Java             3      4
## 4 GO               4      2
## 5 <NA>             5     NA
## 6 SQL              6      3
## # ... with 2 more rows
```

## Part II 基础

**题目 21（读取数据）：读取本地 Excel 数据**

难度：★

代码及运行结果：

```
df = readxl::read_xlsx("data/21-50 数据.xlsx")
df
```

```
## # A tibble: 135 x 3
##   createTime          education salary
##   <dttm>              <chr>     <chr>
## 1 2020-03-16 11:30:18 本科      20k-35k
## 2 2020-03-16 10:58:48 本科      20k-40k
## 3 2020-03-16 10:46:39 不限      20k-35k
## 4 2020-03-16 10:45:44 本科      13k-20k
## 5 2020-03-16 10:20:41 本科      10k-20k
## 6 2020-03-16 10:33:48 本科      10k-18k
## # ... with 129 more rows
```

**题目 22（查看数据）：查看 df 数据的前几行**

难度：⋆

代码及运行结果：

```
head(df, 5)
```

```
## # A tibble: 5 x 3
##   createTime          education salary
##   <dttm>              <chr>     <chr>
## 1 2020-03-16 11:30:18 本科      20k-35k
## 2 2020-03-16 10:58:48 本科      20k-40k
## 3 2020-03-16 10:46:39 不限      20k-35k
## 4 2020-03-16 10:45:44 本科      13k-20k
## 5 2020-03-16 10:20:41 本科      10k-20k
```

**题目 23（数据计算）：将 salary 列数据转换为最大值与最小值的平均值**

难度：⋆⋆⋆

代码及运行结果：

```
df = df %>%
  separate(salary, into = c("low", "high"), sep = "-") %>%  # sep="-" 也可以省略
  mutate(salary = (parse_number(low) + parse_number(high)) * 1000 / 2) %>%
  select(-c(low, high))
df
```

```
## # A tibble: 135 x 3
##   createTime          education salary
##   <dttm>              <chr>      <dbl>
## 1 2020-03-16 11:30:18 本科       27500
## 2 2020-03-16 10:58:48 本科       30000
## 3 2020-03-16 10:46:39 不限       27500
## 4 2020-03-16 10:45:44 本科       16500
## 5 2020-03-16 10:20:41 本科       15000
## 6 2020-03-16 10:33:48 本科       14000
## # ... with 129 more rows
```

或者来个高级的，用正则表达式提取数字，定义做计算的函数，再 purrr::map_dbl 做循环计算：

```
calc = function(x) sum(as.numeric(unlist(x))) * 1000 / 2

df %>%
  mutate(salary = map_dbl(str_extract_all(salary, "\\d+"), calc))   # 结果同上（略）
```

题目 **24**（分组汇总）：根据学历分组，并计算平均薪资

难度：⋆⋆⋆

代码及运行结果：

```
df %>%
  group_by(education) %>%
  summarise(salary_avg = mean(salary))
```

```
## # A tibble: 4 x 2
##   education salary_avg
##   <chr>          <dbl>
## 1 不限           19600
## 2 大专           10000
## 3 本科           19361.
## 4 硕士           20643.
```

题目 **25**（时间转换）：将 **createTime** 列转换为” 月-日”

难度：⋆⋆⋆

代码及运行结果：

```
library(lubridate)
df %>%
  mutate(createTime = str_sub(createTime, 6, 10))
```

```
## # A tibble: 135 x 3
##   createTime education salary
##   <chr>      <chr>      <dbl>
## 1 03-16      本科       27500
## 2 03-16      本科       30000
## 3 03-16      不限       27500
## 4 03-16      本科       16500
## 5 03-16      本科       15000
## 6 03-16      本科       14000
## # ... with 129 more rows
```

题目 **26**（查看数据）：查看数据结构信息

难度：⋆

代码及运行结果：

```r
glimpse(df)       # 或者用 str()
```

```
## Rows: 135
## Columns: 3
## $ createTime <dttm> 2020-03-16 11:30:18, 2020-03-16 10:58:48, 2020-03-16 10:46~
## $ education  <chr> "本科", "本科", "不限", "本科", "本科", "本科", "硕士", "本~
## $ salary     <dbl> 27500, 30000, 27500, 16500, 15000, 14000, 23000, 12500, 700~
```

```r
object.size(df)   # 查看对象占用内存
```

```
## 5112 bytes
```

**题目 27（查看数据）：查看数据汇总信息**

难度：⋆

代码及运行结果：

```r
summary(df)
```

```
##    createTime                     education             salary
##  Min.   :2020-03-13 18:01:31.00   Length:135         Min.   : 3500
##  1st Qu.:2020-03-16 10:41:19.50   Class :character   1st Qu.:14000
##  Median :2020-03-16 11:00:27.00   Mode  :character   Median :17500
##  Mean   :2020-03-16 10:16:35.36                      Mean   :19159
##  3rd Qu.:2020-03-16 11:19:03.00                      3rd Qu.:25000
##  Max.   :2020-03-16 11:36:07.00                      Max.   :45000
```

**题目 28（修改列）：新增一列将 salary 离散化为三水平值**

难度：⋆⋆⋆⋆

代码及运行结果：

```r
df = df %>%
  mutate(class = case_when(
    salary >= 0 & salary < 5000     ~ " 低",
    salary >= 5000 & salary < 20000 ~ " 中",
    .default =  " 高"))
df
```

```
## # A tibble: 135 x 4
##   createTime          education salary class
##   <dttm>              <chr>      <dbl> <chr>
## 1 2020-03-16 11:30:18 本科       27500 高
```

```
## 2 2020-03-16 10:58:48 本科        30000 高
## 3 2020-03-16 10:46:39 不限        27500 高
## 4 2020-03-16 10:45:44 本科        16500 中
## 5 2020-03-16 10:20:41 本科        15000 中
## 6 2020-03-16 10:33:48 本科        14000 中
## # ... with 129 more rows
```

- 或者用 cut() 函数：

```
df %>%
  mutate(class = cut(salary,
                     breaks = c(0,5000,20000,Inf),
                     labels = c(" 低", " 中", " 高"),
                     right = FALSE))
```

- 或者用 sjmisc 包中的 rec()，和 SPSS 的重新编码一样强大。

```
df %>%
  mutate(class = sjmisc::rec(salary,
    rec = "min:5000 = 低; 5000:20000 = 中; 20000:max = 高"))
```

**题目 29（数据整理）：按 salary 列对数据降序排列**

难度：★★

代码及运行结果：

```
df %>%
  arrange(-salary)
```

```
## # A tibble: 135 x 4
##   createTime          education salary class
##   <dttm>              <chr>      <dbl> <chr>
## 1 2020-03-16 11:30:17 本科       45000 高
## 2 2020-03-16 11:04:00 本科       40000 高
## 3 2020-03-16 10:36:57 本科       37500 高
## 4 2020-03-16 11:01:39 本科       37500 高
## 5 2020-03-16 09:54:47 硕士       37500 高
## 6 2020-03-16 11:01:22 本科       35000 高
## # ... with 129 more rows
```

**题目 30（筛选行）：提取第 33 行数据**

难度：★

代码及运行结果：

```
df %>%
  slice(33)       # 或者 df[33,]
```

```
## # A tibble: 1 x 4
##   createTime          education salary class
##   <dttm>              <chr>      <dbl> <chr>
## 1 2020-03-16 10:07:25 硕士       22500 高
```

**题目 31（数据计算）：计算 salary 列的中位数**

难度：⋆

代码及运行结果：

```
df %>%
  summarise(med = median(salary))  # 或者 median(df$salary)
```

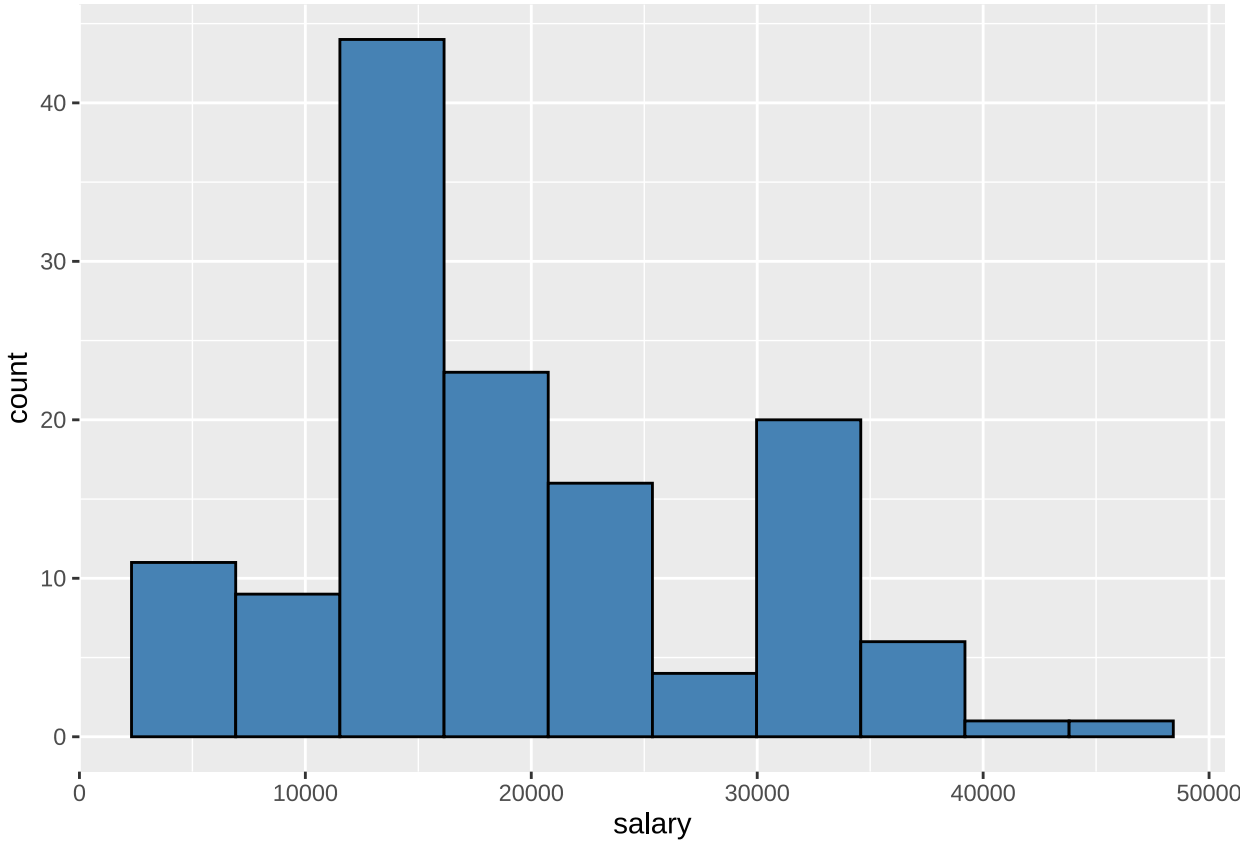
```
## # A tibble: 1 x 1
##     med
##   <dbl>
## 1 17500
```

**题目 32（数据可视化）：绘制 salary 的频率分布直方图**

难度：⋆⋆⋆

代码及运行结果：

```
df %>%
  ggplot(aes(salary)) +
  geom_histogram(bins = 10, fill = "steelblue", color = "black")
```

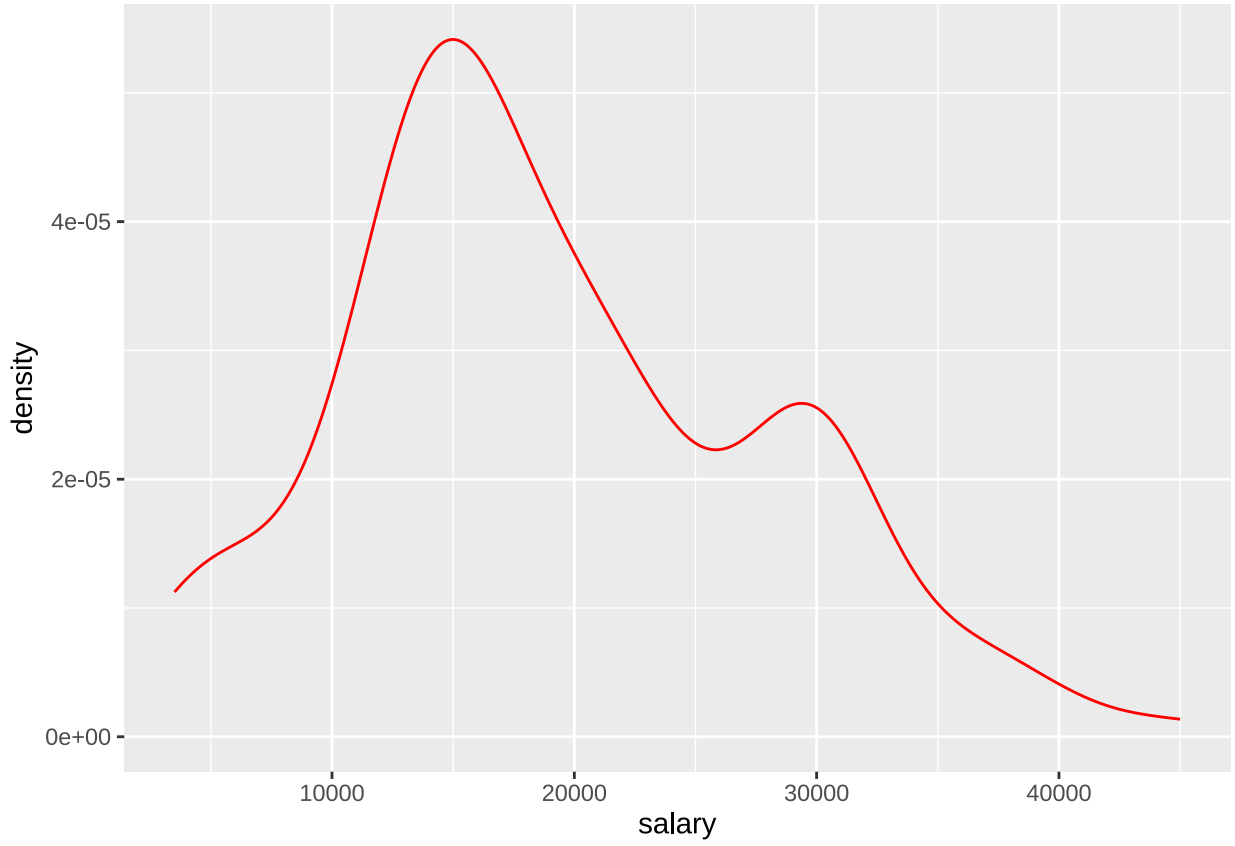



**题目 33（数据可视化）：绘制 salary 的频率密度曲线图**

难度：★★★

代码及运行结果：

```
df %>%
  ggplot(aes(salary)) +
  geom_density(color = "red")
```




**题目 34（数据删除）：删除最后一列 class**

难度：⋆

代码及运行结果：

```
df %>%
  select(-class)
```

```
## # A tibble: 135 x 3
##   createTime          education salary
##   <dttm>              <chr>      <dbl>
## 1 2020-03-16 11:30:18 本科       27500
## 2 2020-03-16 10:58:48 本科       30000
## 3 2020-03-16 10:46:39 不限       27500
## 4 2020-03-16 10:45:44 本科       16500
## 5 2020-03-16 10:20:41 本科       15000
## 6 2020-03-16 10:33:48 本科       14000
## # ... with 129 more rows
```

```
# 或者
# df %>%
#  select(-last_col())  # 同 last_col(0)
```

**题目 35（数据操作）：将 df 的第 1 列与第 2 列合并为新的一列**

难度：★★

代码及运行结果：

```
df %>%
  unite("newcol", 1:2, sep = " ")
```

```
## # A tibble: 135 x 3
##   newcol                   salary class
##   <chr>                     <dbl> <chr>
## 1 2020-03-16 11:30:18 本科  27500 高
## 2 2020-03-16 10:58:48 本科  30000 高
## 3 2020-03-16 10:46:39 不限  27500 高
## 4 2020-03-16 10:45:44 本科  16500 中
## 5 2020-03-16 10:20:41 本科  15000 中
## 6 2020-03-16 10:33:48 本科  14000 中
## # ... with 129 more rows
```

**题目 36（数据操作）：将 education 列与第 salary 列合并为新的一列**

难度：★★

代码及运行结果：

```
df %>%
  unite("newcol", c(education, salary), sep = " ")
```

```
## # A tibble: 135 x 3
##   createTime          newcol     class
##   <dttm>              <chr>      <chr>
## 1 2020-03-16 11:30:18 本科 27500 高
## 2 2020-03-16 10:58:48 本科 30000 高
## 3 2020-03-16 10:46:39 不限 27500 高
## 4 2020-03-16 10:45:44 本科 16500 中
## 5 2020-03-16 10:20:41 本科 15000 中
## 6 2020-03-16 10:33:48 本科 14000 中
## # ... with 129 more rows
```

**题目 37（数据计算）：计算 salary 最大值与最小值之差**

难度：★★

代码及运行结果：

```
max(df$salary) - min(df$salary)
```

```
## [1] 41500
```

或者用

```
df %>%
  summarise(range = max(salary) - min(salary))
```

```
## # A tibble: 1 x 1
##   range
##   <dbl>
## 1 41500
```

**题目 38（数据操作）：将第一行与最后一行拼接**

难度：★★

代码及运行结果：

```
df %>%
  slice(1, n())
```

```
## # A tibble: 2 x 4
##   createTime          education salary class
##   <dttm>              <chr>      <dbl> <chr>
## 1 2020-03-16 11:30:18 本科       27500 高
## 2 2020-03-16 11:19:38 本科       30000 高
```

**题目 39（数据操作）：将第 8 行添加到末尾**

难度：★★

代码及运行结果：

```
df %>%
  bind_rows(slice(., 8))
```

```
## # A tibble: 136 x 4
##   createTime          education salary class
##   <dttm>              <chr>      <dbl> <chr>
## 1 2020-03-16 11:30:18 本科       27500 高
```

```
## 2 2020-03-16 10:58:48 本科        30000 高
## 3 2020-03-16 10:46:39 不限        27500 高
## 4 2020-03-16 10:45:44 本科        16500 中
## 5 2020-03-16 10:20:41 本科        15000 中
## 6 2020-03-16 10:33:48 本科        14000 中
## # ... with 130 more rows
```

**题目 40（查看数据）：查看每一列的数据类型**

难度：⋆

代码及运行结果：

```
glimpse(df)     # 或者用 str()
```

```
## Rows: 135
## Columns: 4
## $ createTime <dttm> 2020-03-16 11:30:18, 2020-03-16 10:58:48, 2020-03-16 10:46~
## $ education  <chr> "本科", "本科", "不限", "本科", "本科", "本科", "硕士", "本~
## $ salary     <dbl> 27500, 30000, 27500, 16500, 15000, 14000, 23000, 12500, 700~
## $ class      <chr> "高", "高", "高", "中", "中", "中", "高", "中", "中", "中",~
```

**题目 41（数据操作）：将 createTime 列设置为行索引**

难度：⋆⋆

代码及运行结果：

```
df %>%
  distinct(createTime, .keep_all = TRUE) %>%
  column_to_rownames("createTime")
```

```
##                     education salary class
## 2020-03-16 11:30:18      本科  27500    高
## 2020-03-16 10:58:48      本科  30000    高
## 2020-03-16 10:46:39      不限  27500    高
## 2020-03-16 10:45:44      本科  16500    中
## 2020-03-16 10:20:41      本科  15000    中
## 2020-03-16 10:33:48      本科  14000    中
## 2020-03-16 10:11:54      硕士  23000    高
## 2020-03-16 09:49:12      本科  12500    中
## 2020-03-16 09:25:48      不限   7000    中
## 2020-03-16 09:35:50      本科  16000    中
## 2020-03-16 10:34:19      本科  20000    高
```

```
## 2020-03-16 09:30:40      本科  10000    中
## 2020-03-16 11:30:17      本科  30000    高
## 2020-03-16 10:54:56      本科  25000    高
## 2020-03-15 12:14:45      本科  30000    高
## 2020-03-16 10:52:14      硕士  12500    中
## 2020-03-16 10:36:57      本科  37500    高
## 2020-03-16 11:01:23      本科  12500    中
## 2020-03-16 11:01:22      本科  35000    高
## 2020-03-16 11:03:56      本科  20000    高
## 2020-03-16 11:00:27      本科  11500    中
## 2020-03-16 11:04:44      本科  30000    高
## 2020-03-16 11:30:15      本科  20000    高
## 2020-03-16 11:02:12      本科  22500    高
## 2020-03-16 11:19:06      本科  15000    中
## 2020-03-16 11:04:45      本科  30000    高
## 2020-03-16 11:04:49      本科  30000    高
## 2020-03-16 10:27:11      本科  17500    中
## 2020-03-16 10:43:58      硕士  22500    高
## 2020-03-16 11:01:46      本科  22500    高
## 2020-03-16 10:07:25      硕士  22500    高
## 2020-03-16 11:04:00      本科  30000    高
## 2020-03-16 10:43:53      本科  24000    高
## 2020-03-13 18:01:31      本科  16000    中
## 2020-03-16 10:15:50      本科  22500    高
## 2020-03-16 11:01:58      不限  30000    高
## 2020-03-16 11:35:36      本科  30000    高
## 2020-03-16 11:30:14      本科  20000    高
## 2020-03-16 11:30:02      本科  30000    高
## 2020-03-16 11:18:00      本科  12500    中
## 2020-03-16 11:16:49      本科  30000    高
## 2020-03-16 10:59:08      本科  12000    中
## 2020-03-16 11:35:39      本科  11500    中
## 2020-03-16 11:30:09      本科  17500    中
## 2020-03-16 11:29:50      本科  16000    中
## 2020-03-16 11:35:34      本科  12500    中
## 2020-03-16 11:28:48      本科  13500    中
## 2020-03-16 11:32:46      本科  15000    中
## 2020-03-16 11:33:00      本科  16500    中
## 2020-03-16 11:09:18      本科  15000    中
```

```
## 2020-03-16 11:35:33      本科  25000    高
## 2020-03-16 11:01:07      本科  22500    高
## 2020-03-16 10:57:55      本科  20000    高
## 2020-03-16 10:51:27      本科   9500    中
## 2020-03-16 10:46:50      本科  22500    高
## 2020-03-16 10:54:10      本科  17500    中
## 2020-03-16 10:41:20      本科  35000    高
## 2020-03-16 10:41:19      本科  25000    高
## 2020-03-16 10:46:31      本科  18000    中
## 2020-03-16 10:34:43      本科  27500    高
## 2020-03-16 10:43:47      本科  14000    中
## 2020-03-16 10:34:27      本科  17500    中
## 2020-03-16 10:26:23      本科  15000    中
## 2020-03-16 10:18:39      本科  12000    中
## 2020-03-16 11:20:44      本科   6500    中
## 2020-03-16 11:19:03      本科  14000    中
## 2020-03-16 11:17:58      本科  14000    中
## 2020-03-16 11:10:42      本科  20000    高
## 2020-03-16 10:18:35      本科  25000    高
## 2020-03-16 11:01:08      本科  15000    中
## 2020-03-16 11:30:10      本科  32500    高
## 2020-03-16 10:52:45      本科  16000    中
## 2020-03-16 10:50:23      本科   5000    中
## 2020-03-16 10:43:49      本科  30000    高
## 2020-03-16 10:43:46      本科  14000    中
## 2020-03-16 10:26:50      本科  15000    中
## 2020-03-16 11:33:08      本科  15000    中
## 2020-03-16 10:27:10      硕士  14000    中
## 2020-03-16 10:44:41      本科  30000    高
## 2020-03-16 11:12:04      本科  15000    中
## 2020-03-16 10:44:23      不限   3500    低
## 2020-03-16 10:27:45      本科  30000    高
## 2020-03-16 11:01:39      本科  37500    高
## 2020-03-16 10:21:52      大专  15000    中
## 2020-03-16 11:21:24      本科  16000    中
## 2020-03-16 10:26:12      大专   5000    中
## 2020-03-16 10:59:15      本科  16000    中
## 2020-03-16 11:13:50      本科  12500    中
## 2020-03-16 11:13:16      本科  26500    高
```

```
## 2020-03-16 10:48:43      本科   3500    低
## 2020-03-16 11:32:07      本科   8500    中
## 2020-03-16 11:19:36      本科  30000    高
## 2020-03-16 11:19:38      本科  30000    高
## 2020-03-16 09:28:37      本科  18500    中
## 2020-03-16 10:09:18      本科  11500    中
## 2020-03-16 11:33:13      本科   5000    中
## 2020-03-16 10:00:03      本科  20000    高
## 2020-03-16 09:44:05      硕士  12500    中
## 2020-03-16 10:57:27      本科  22500    高
## 2020-03-16 09:46:26      本科  20000    高
## 2020-03-16 11:36:07      本科  14000    中
## 2020-03-16 09:54:47      硕士  37500    高
## 2020-03-16 10:48:32      本科  30000    高
```

注：行索引不允许有重复，所以先做了一步去重。

**题目 42（数据创建）：生成一个和 df 长度相同的随机数数据框**

难度：★★

代码及运行结果：

```
df1 = tibble(rnums = sample(10, nrow(df), replace = TRUE))
df1
```

```
## # A tibble: 135 x 1
##   rnums
##   <int>
## 1     4
## 2     2
## 3     3
## 4     4
## 5     5
## 6    10
## # ... with 129 more rows
```

**题目 43（数据连接）：将上面生成的数据框与 df 按列合并**

难度：★★

代码及运行结果：

```
df = bind_cols(df, df1)
df
```

```
## # A tibble: 135 x 5
##   createTime          education salary class rnums
##   <dttm>              <chr>      <dbl> <chr> <int>
## 1 2020-03-16 11:30:18 本科       27500 高        4
## 2 2020-03-16 10:58:48 本科       30000 高        2
## 3 2020-03-16 10:46:39 不限       27500 高        3
## 4 2020-03-16 10:45:44 本科       16500 中        4
## 5 2020-03-16 10:20:41 本科       15000 中        5
## 6 2020-03-16 10:33:48 本科       14000 中       10
## # ... with 129 more rows
```

注：实际上，42，43 题应该合并成一个题，这是数据操作中最常规的修改列：

```
df %>%
  mutate(rnums = sample(10, n(), replace = TRUE))
```

```
## # A tibble: 135 x 5
##   createTime          education salary class rnums
##   <dttm>              <chr>      <dbl> <chr> <int>
## 1 2020-03-16 11:30:18 本科       27500 高        3
## 2 2020-03-16 10:58:48 本科       30000 高        5
## 3 2020-03-16 10:46:39 不限       27500 高       10
## 4 2020-03-16 10:45:44 本科       16500 中        8
## 5 2020-03-16 10:20:41 本科       15000 中        9
## 6 2020-03-16 10:33:48 本科       14000 中        4
## # ... with 129 more rows
```

**题目 44（修改列）：生成新列 new 为 salary 列减去随机数列**

难度：⋆⋆

代码及运行结果：

```
df = df %>%
  mutate(new = salary - rnums)
df
```

```
## # A tibble: 135 x 6
##   createTime          education salary class rnums   new
##   <dttm>              <chr>      <dbl> <chr> <int> <dbl>
## 1 2020-03-16 11:30:18 本科       27500 高        4 27496
```

```
## 2 2020-03-16 10:58:48 本科        30000 高         2 29998
## 3 2020-03-16 10:46:39 不限        27500 高         3 27497
## 4 2020-03-16 10:45:44 本科        16500 中         4 16496
## 5 2020-03-16 10:20:41 本科        15000 中         5 14995
## 6 2020-03-16 10:33:48 本科        14000 中        10 13990
## # ... with 129 more rows
```

**题目 45（检查缺失值）：检查数据中是否含有任何缺失值**

难度：★★

代码及运行结果：

```
anyNA(df)
```

```
## [1] FALSE
```

```
anyNA(df$salary)
```

```
## [1] FALSE
```

注：naniar 包提供了更强大的探索缺失值及缺失模式的函数，其中 `miss_var_summary()` 和 `miss_case_summary()` 可检查各列和各行缺失情况。

**题目 46（类型转换）：将 salary 列的类型转换为浮点数**

难度：★★

代码及运行结果：

```
df %>%
  mutate(rnums = as.double(rnums))
```

```
## # A tibble: 135 x 6
##   createTime          education salary class rnums   new
##   <dttm>              <chr>      <dbl> <chr> <dbl> <dbl>
## 1 2020-03-16 11:30:18 本科        27500 高         4 27496
## 2 2020-03-16 10:58:48 本科        30000 高         2 29998
## 3 2020-03-16 10:46:39 不限        27500 高         3 27497
## 4 2020-03-16 10:45:44 本科        16500 中         4 16496
## 5 2020-03-16 10:20:41 本科        15000 中         5 14995
## 6 2020-03-16 10:33:48 本科        14000 中        10 13990
## # ... with 129 more rows
```

**题目 47（数据汇总）：计算 salary 列大于 10000 的次数**

难度：★★★

代码及运行结果：

```
df %>%
  count(salary > 10000)
```

```
## # A tibble: 2 x 2
##   `salary > 10000`     n
##   <lgl>            <int>
## 1 FALSE               16
## 2 TRUE               119
```

或者用

```
df %>%
  summarise(n = sum(salary > 10000))
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1   119
```

**题目 48（统计频数）：查看每种学历出现的次数**

难度：★★

代码及运行结果：

```
df %>%
  count(education)
```

```
## # A tibble: 4 x 2
##   education     n
##   <chr>     <int>
## 1 不限          5
## 2 大专          4
## 3 本科        119
## 4 硕士          7
```

**题目 49（数据汇总）：查看 education 列共有几种学历**

难度：★★

代码及运行结果：

```
df %>%
  distinct(education)
```

```
## # A tibble: 4 x 1
##   education
##   <chr>
## 1 本科
## 2 不限
## 3 硕士
## 4 大专
```

**题目 50（筛选行）：提取 salary 与 new 列之和大于 60000 的最后 3 行**

难度：★★★★

代码及运行结果：

```
df %>%
  filter(salary + new > 60000) %>%
  slice_tail(n = 3)
```

```
## # A tibble: 3 x 6
##   createTime          education salary class rnums   new
##   <dttm>              <chr>      <dbl> <chr> <int> <dbl>
## 1 2020-03-16 10:41:20 本科       35000 高        8 34992
## 2 2020-03-16 11:01:39 本科       37500 高        8 37492
## 3 2020-03-16 09:54:47 硕士       37500 高       10 37490
```

## Part III 提高

**题目 51（读取数据）：使用绝对路径读取本地 Excel 数据**

难度：★

代码及运行结果：

```
df = readxl::read_xls("data/51-80 数据.xls")
df
```

```
## # A tibble: 327 x 18
##   代码      简称     日期                前收~1 开盘价~2 最高~3 最低价~4 收盘~5
##   <chr>     <chr>    <dttm>               <dbl>    <dbl>  <dbl>    <dbl>  <dbl>
## 1 600000.SH 浦发银行 2016-01-04 00:00:00   16.1     16.1   16.1     15.5   15.7
## 2 600000.SH 浦发银行 2016-01-05 00:00:00   15.7     15.5   16.0     15.4   15.9
## 3 600000.SH 浦发银行 2016-01-06 00:00:00   15.9     15.8   16.0     15.6   16.0
## 4 600000.SH 浦发银行 2016-01-07 00:00:00   16.0     15.7   15.8     15.4   15.5
## 5 600000.SH 浦发银行 2016-01-08 00:00:00   15.5     15.7   15.8     14.9   15.4
## 6 600000.SH 浦发银行 2016-01-11 00:00:00   15.4     15.2   15.4     15.0   15.1
```

```
## # ... with 321 more rows, 10 more variables: `成交量(股)` <chr>,
## #   `成交金额(元)` <chr>, `涨跌(元)` <dbl>, `涨跌幅(%)` <dbl>,
## #   `均价(元)` <chr>, `换手率(%)` <chr>, `A股流通市值(元)` <dbl>,
## #   `总市值(元)` <dbl>, `A股流通股本(股)` <dbl>, 市盈率 <dbl>, and abbreviated
## #   variable names 1: `前收盘价(元)`, 2: `开盘价(元)`, 3: `最高价(元)`,
## #   4: `最低价(元)`, 5: `收盘价(元)`
```

**题目 52（查看数据）：查看数据框的前 3 行**

**难度：** ⋆

**代码及运行结果：**

```
head(df, 3)
```

```
## # A tibble: 3 x 18
##   代码      简称     日期                前收~1 开盘价~2 最高~3 最低价~4 收盘~5
##   <chr>     <chr>    <dttm>               <dbl>    <dbl>  <dbl>    <dbl>  <dbl>
## 1 600000.SH 浦发银行 2016-01-04 00:00:00   16.1     16.1   16.1     15.5   15.7
## 2 600000.SH 浦发银行 2016-01-05 00:00:00   15.7     15.5   16.0     15.4   15.9
## 3 600000.SH 浦发银行 2016-01-06 00:00:00   15.9     15.8   16.0     15.6   16.0
## # ... with 10 more variables: `成交量(股)` <chr>, `成交金额(元)` <chr>,
## #   `涨跌(元)` <dbl>, `涨跌幅(%)` <dbl>, `均价(元)` <chr>, `换手率(%)` <chr>,
## #   `A股流通市值(元)` <dbl>, `总市值(元)` <dbl>, `A股流通股本(股)` <dbl>,
## #   市盈率 <dbl>, and abbreviated variable names 1: `前收盘价(元)`,
## #   2: `开盘价(元)`, 3: `最高价(元)`, 4: `最低价(元)`, 5: `收盘价(元)`
```

**说明：**当前数据不包含缺失值，接下来关于缺失值的题目 53-56，改用自带的 starwars 数据演示。

**题目 53（查看缺失值）：查看每列数据缺失值情况**

**难度：** ⋆⋆

**代码及运行结果：**

```
map_int(starwars, ~ sum(is.na(.x)))
```

```
##       name     height       mass hair_color skin_color  eye_color birth_year
##          0          6         28          5          0          0         44
##        sex     gender  homeworld    species      films   vehicles  starships
##          4          4         10          4          0          0          0
```

**注：**也可以用 naniar 包中的 miss_var_summary() 函数。

题目 **54**（查看缺失值）：查看日期列含有缺失值的行

难度：⋆⋆

代码及运行结果：

```
starwars %>%
  filter(is.na(hair_color))
```

```
## # A tibble: 5 x 14
##   name         height  mass hair_~1 skin_~2 eye_c~3 birth~4 sex   gender homew~5
##   <chr>         <int> <dbl> <chr>   <chr>   <chr>     <dbl> <chr> <chr>  <chr>
## 1 C-3PO           167    75 <NA>    gold    yellow      112 none  mascu~ Tatooi~
## 2 R2-D2            96    32 <NA>    white,~ red          33 none  mascu~ Naboo
## 3 R5-D4            97    32 <NA>    white,~ red          NA none  mascu~ Tatooi~
## 4 Greedo          173    74 <NA>    green   black        44 male  mascu~ Rodia
## 5 Jabba Desil~    175  1358 <NA>    green-~ orange      600 herm~ mascu~ Nal Hu~
## # ... with 4 more variables: species <chr>, films <list>, vehicles <list>,
## #   starships <list>, and abbreviated variable names 1: hair_color,
## #   2: skin_color, 3: eye_color, 4: birth_year, 5: homeworld
```

题目 **55**（查看缺失值）：查看每列缺失值在哪些行

难度：⋆⋆⋆

代码及运行结果：

```
map(starwars, ~ which(is.na(.x)))
```

```
## $name
## integer(0)
##
## $height
## [1] 28 82 83 84 85 86
##
## $mass
##  [1] 12 27 28 33 36 37 38 40 41 43 46 51 53 54 56 58 59 63 65 70 71 73 75 82 83
## [26] 84 85 86
##
## $hair_color
## [1]  2  3  8 15 16
##
## $skin_color
## integer(0)
```

```
##
## $eye_color
## integer(0)
##
## $birth_year
##  [1]  8 18 28 30 32 35 36 37 38 39 43 45 46 47 50 51 52 53 54 56 57 58 60 63 67
## [26] 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86
##
## $sex
## [1] 37 40 80 86
##
## $gender
## [1] 37 40 80 86
##
## $homeworld
##  [1] 19 22 28 31 73 82 83 84 85 86
##
## $species
## [1] 37 40 80 86
##
## $films
## integer(0)
##
## $vehicles
## integer(0)
##
## $starships
## integer(0)
```

**题目 56（缺失值处理）：删除所有存在缺失值的行**

**难度：⋆⋆**

**代码及运行结果：**

```
starwars %>%
  drop_na()
```

```
## # A tibble: 29 x 14
##   name         height  mass hair_~1 skin_~2 eye_c~3 birth~4 sex   gender homew~5
##   <chr>         <int> <dbl> <chr>   <chr>   <chr>     <dbl> <chr> <chr>  <chr>
## 1 Luke Skywal~    172    77 blond   fair    blue       19   male  mascu~ Tatooi~
```

```
## 2 Darth Vader      202   136 none    white   yellow      41.9 male  mascu~ Tatooi~
## 3 Leia Organa      150    49 brown   light   brown       19   fema~ femin~ Aldera~
## 4 Owen Lars        178   120 brown,~ light   blue        52   male  mascu~ Tatooi~
## 5 Beru Whites~     165    75 brown   light   blue        47   fema~ femin~ Tatooi~
## 6 Biggs Darkl~     183    84 black   light   brown       24   male  mascu~ Tatooi~
## # ... with 23 more rows, 4 more variables: species <chr>, films <list>,
## #   vehicles <list>, starships <list>, and abbreviated variable names
## #   1: hair_color, 2: skin_color, 3: eye_color, 4: birth_year, 5: homeworld
```

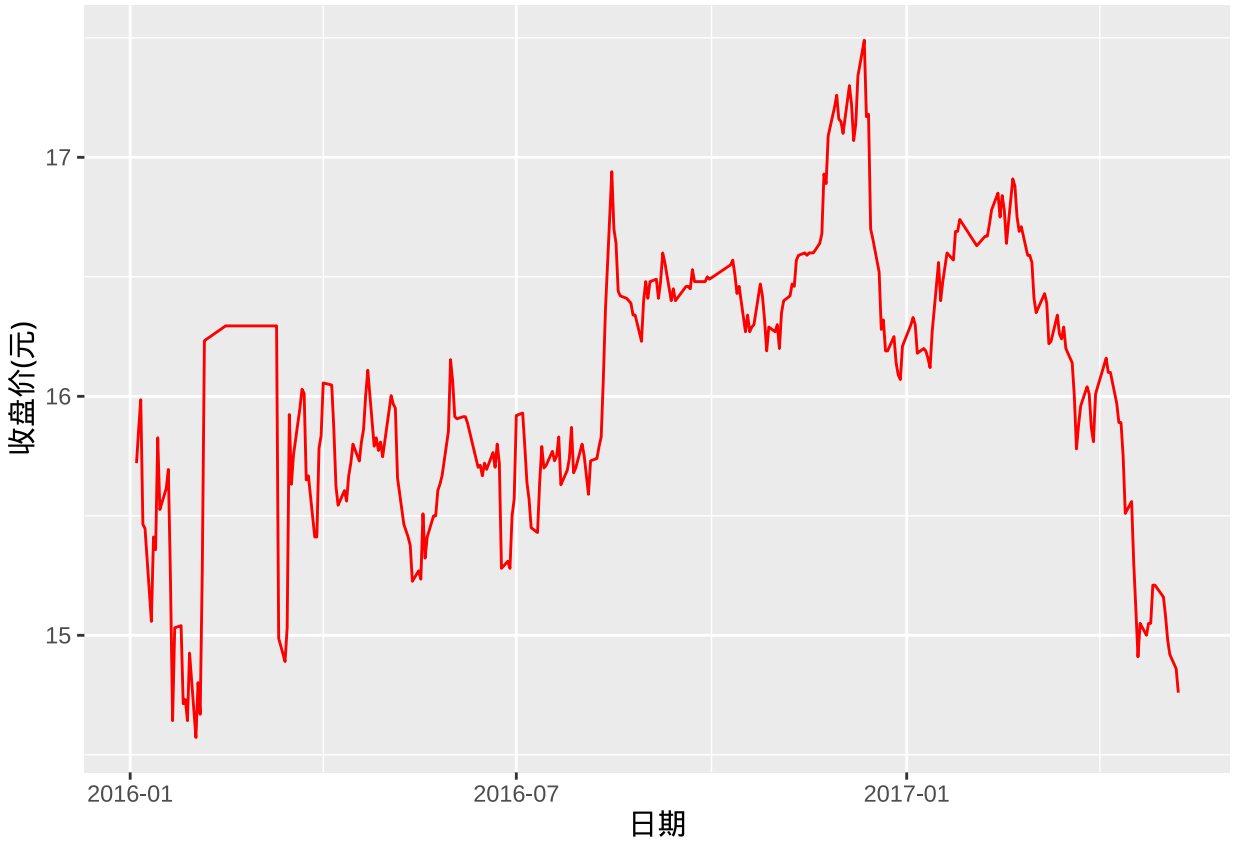
注：若要删除某些列包含缺失值的行，提供列名即可。

**题目 57（数据可视化）：绘制收盘价的折线图**

难度：★★

代码及运行结果：

```
df %>%
    ggplot(aes(日期, `收盘价（元）`)) +
    geom_line(color = "red")
```



题目 58（数据可视化）：同时绘制开盘价与收盘价

难度：★★★

代码及运行结果：

```
df %>%
  select(日期, `开盘价（元）`, `收盘价（元）`) %>%
  pivot_longer(-日期, names_to = "type", values_to = "price") %>%
  ggplot(aes(日期, price, color = type)) +
  geom_line()
```

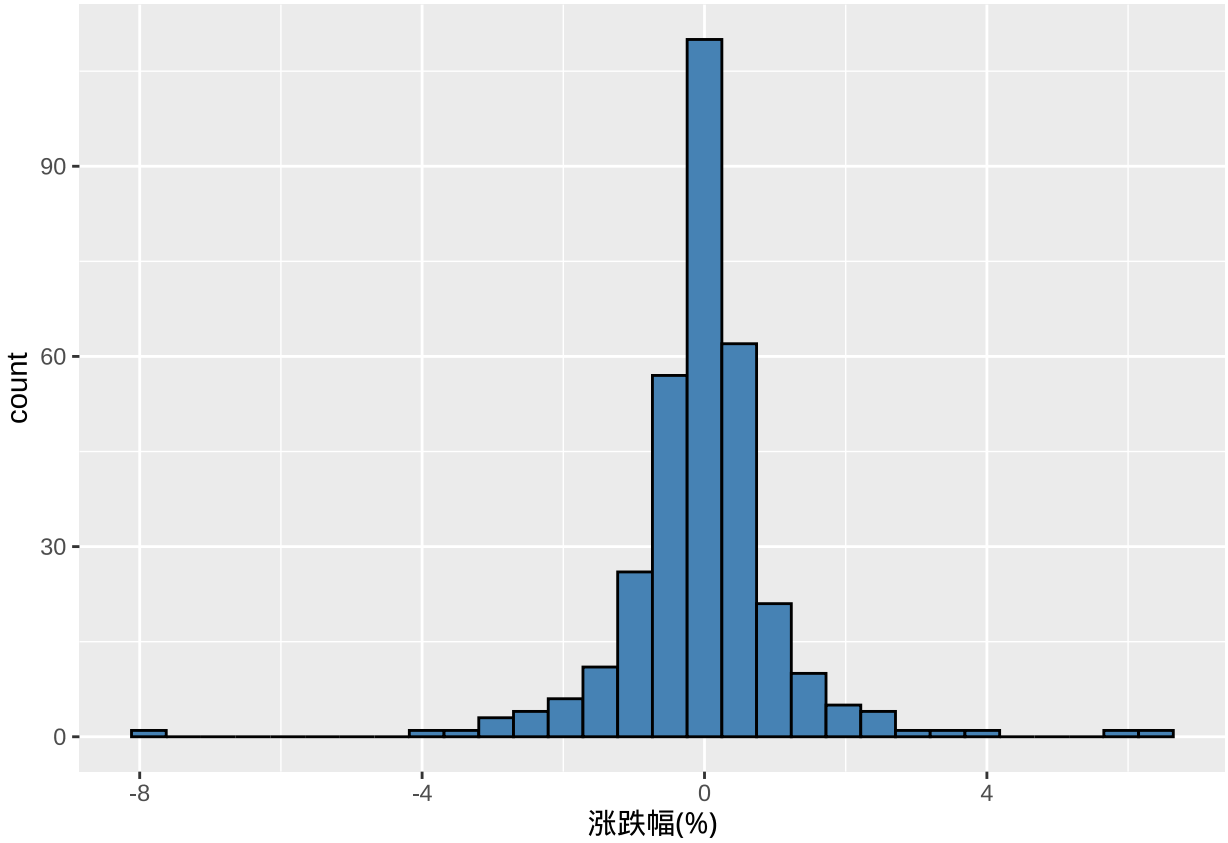

注：为了自动添加图例，先对数据做了宽变长转换。

题目 59（数据可视化）：绘制涨跌幅的直方图

难度：★★

代码及运行结果：

```
df %>%
  ggplot(aes(`涨跌幅（%）`)) +
  geom_histogram(fill = "steelblue", color = "black")
```

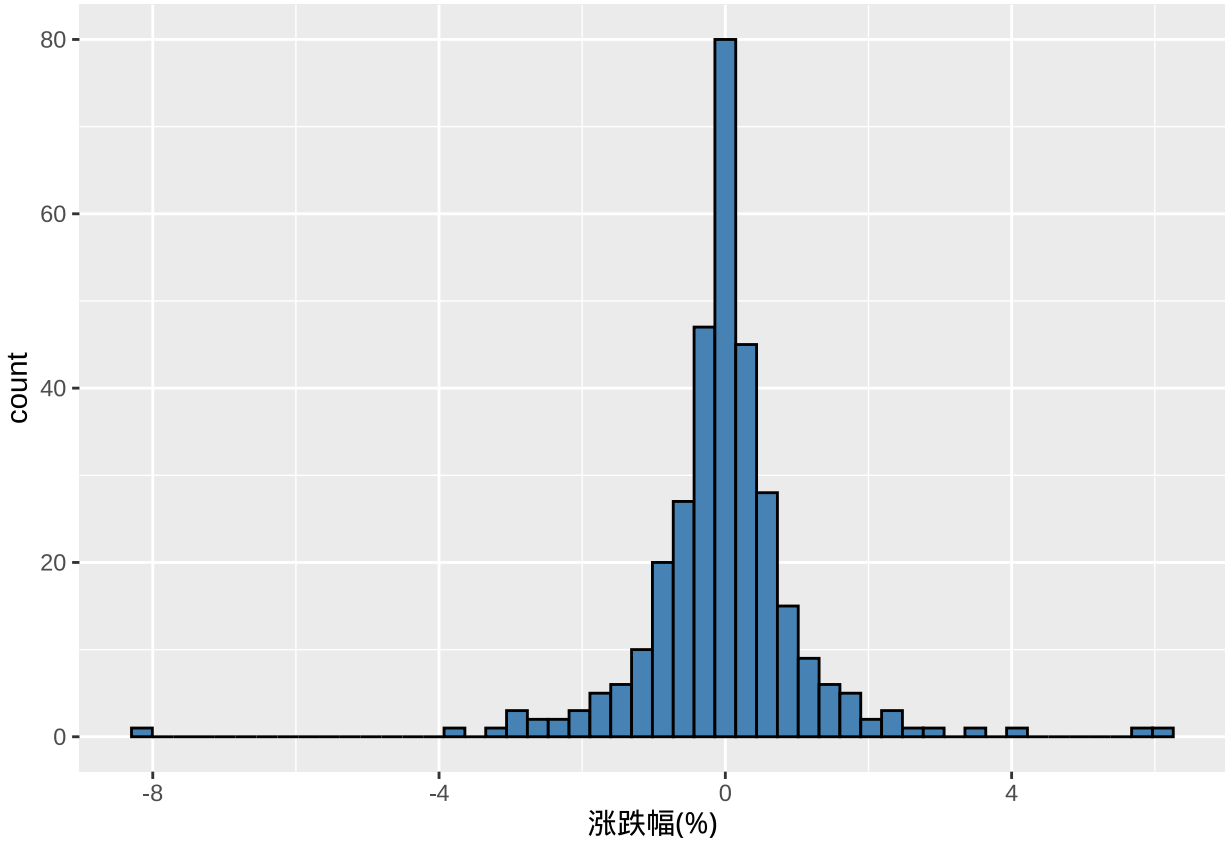



题目 60（数据可视化）：让直方图更细致

难度：★★

代码及运行结果：

```
df %>%
  ggplot(aes(`涨跌幅 (%)`)) +
  geom_histogram(bins = 50, fill = "steelblue", color = "black")
```




题目 **61**（数据创建）：用 **df** 的列名创建数据框

难度：⋆⋆

代码及运行结果：

```
tibble(Name = names(df))
```

```
## # A tibble: 18 x 1
##   Name
##   <chr>
## 1 代码
## 2 简称
## 3 日期
## 4 前收盘价(元)
## 5 开盘价(元)
## 6 最高价(元)
## # ... with 12 more rows
```

题目 **62**（异常值处理）：输出所有换手率不是数字的行

难度：⋆⋆

代码及运行结果：

```
df %>%
  mutate(`换手率 (%)` = parse_number(`换手率 (%)`)) %>%
  filter(is.na(`换手率 (%)`))
```

```
## # A tibble: 18 x 18
##   代码      简称     日期                前收~1 开盘价~2 最高~3 最低价~4 收盘~5
##   <chr>     <chr>    <dttm>               <dbl>    <dbl>  <dbl>    <dbl>  <dbl>
## 1 600000.SH 浦发银行 2016-02-16 00:00:00   16.3     16.3   16.3     16.3   16.3
## 2 600000.SH 浦发银行 2016-02-17 00:00:00   16.3     16.3   16.3     16.3   16.3
## 3 600000.SH 浦发银行 2016-02-18 00:00:00   16.3     16.3   16.3     16.3   16.3
## 4 600000.SH 浦发银行 2016-02-19 00:00:00   16.3     16.3   16.3     16.3   16.3
## 5 600000.SH 浦发银行 2016-02-22 00:00:00   16.3     16.3   16.3     16.3   16.3
## 6 600000.SH 浦发银行 2016-02-23 00:00:00   16.3     16.3   16.3     16.3   16.3
## # ... with 12 more rows, 10 more variables: `成交量(股)` <chr>,
## #   `成交金额(元)` <chr>, `涨跌(元)` <dbl>, `涨跌幅(%)` <dbl>,
## #   `均价(元)` <chr>, `换手率(%)` <dbl>, `A股流通市值(元)` <dbl>,
## #   `总市值(元)` <dbl>, `A股流通股本(股)` <dbl>, 市盈率 <dbl>, and abbreviated
## #   variable names 1: `前收盘价(元)`, 2: `开盘价(元)`, 3: `最高价(元)`,
## #   4: `最低价(元)`, 5: `收盘价(元)`
```

**题目 63（异常值处理）：输出所有换手率为--的行**

难度：★★

代码及运行结果：

```
df %>%
  filter(`换手率 (%)` == "--")
```

```
## # A tibble: 18 x 18
##   代码      简称     日期                前收~1 开盘价~2 最高~3 最低价~4 收盘~5
##   <chr>     <chr>    <dttm>               <dbl>    <dbl>  <dbl>    <dbl>  <dbl>
## 1 600000.SH 浦发银行 2016-02-16 00:00:00   16.3     16.3   16.3     16.3   16.3
## 2 600000.SH 浦发银行 2016-02-17 00:00:00   16.3     16.3   16.3     16.3   16.3
## 3 600000.SH 浦发银行 2016-02-18 00:00:00   16.3     16.3   16.3     16.3   16.3
## 4 600000.SH 浦发银行 2016-02-19 00:00:00   16.3     16.3   16.3     16.3   16.3
## 5 600000.SH 浦发银行 2016-02-22 00:00:00   16.3     16.3   16.3     16.3   16.3
## 6 600000.SH 浦发银行 2016-02-23 00:00:00   16.3     16.3   16.3     16.3   16.3
## # ... with 12 more rows, 10 more variables: `成交量(股)` <chr>,
## #   `成交金额(元)` <chr>, `涨跌(元)` <dbl>, `涨跌幅(%)` <dbl>,
## #   `均价(元)` <chr>, `换手率(%)` <chr>, `A股流通市值(元)` <dbl>,
```

```
## #   `总市值(元)` <dbl>, `A股流通股本(股)` <dbl>, 市盈率 <dbl>, and abbreviated
## #   variable names 1: `前收盘价(元)`, 2: `开盘价(元)`, 3: `最高价(元)`,
## #   4: `最低价(元)`, 5: `收盘价(元)`
```

**题目 64（数据操作）：重置 df 的行号**

**难度：** ⋆

**代码及运行结果：**

```
rownames(df) = NULL     # R 中无行号就是数字索引
```

**题目 65（异常值处理）：删除所有换手率为非数字的行**

**难度：** ⋆⋆

**代码及运行结果：**

```
df %>%
  mutate(`换手率 (%)` = parse_number(`换手率 (%)`)) %>%
  filter(!is.na(`换手率 (%)`))
```

```
## # A tibble: 309 x 18
##   代码      简称     日期                前收~1 开盘价~2 最高~3 最低价~4 收盘~5
##   <chr>     <chr>    <dttm>               <dbl>    <dbl>  <dbl>    <dbl>  <dbl>
## 1 600000.SH 浦发银行 2016-01-04 00:00:00   16.1     16.1   16.1     15.5   15.7
## 2 600000.SH 浦发银行 2016-01-05 00:00:00   15.7     15.5   16.0     15.4   15.9
## 3 600000.SH 浦发银行 2016-01-06 00:00:00   15.9     15.8   16.0     15.6   16.0
## 4 600000.SH 浦发银行 2016-01-07 00:00:00   16.0     15.7   15.8     15.4   15.5
## 5 600000.SH 浦发银行 2016-01-08 00:00:00   15.5     15.7   15.8     14.9   15.4
## 6 600000.SH 浦发银行 2016-01-11 00:00:00   15.4     15.2   15.4     15.0   15.1
## # ... with 303 more rows, 10 more variables: `成交量(股)` <chr>,
## #   `成交金额(元)` <chr>, `涨跌(元)` <dbl>, `涨跌幅(%)` <dbl>,
## #   `均价(元)` <chr>, `换手率(%)` <dbl>, `A股流通市值(元)` <dbl>,
## #   `总市值(元)` <dbl>, `A股流通股本(股)` <dbl>, 市盈率 <dbl>, and abbreviated
## #   variable names 1: `前收盘价(元)`, 2: `开盘价(元)`, 3: `最高价(元)`,
## #   4: `最低价(元)`, 5: `收盘价(元)`
```

```
library(lubridate)
df = df %>%
  mutate(across(4:18, as.numeric), 日期 = as_date(日期))
df
```

```
## # A tibble: 327 x 18
##   代码   简称   日期       前收~1 开盘价~2 最高~3 最低价~4 收盘~5 成交量~6  成交~7
```

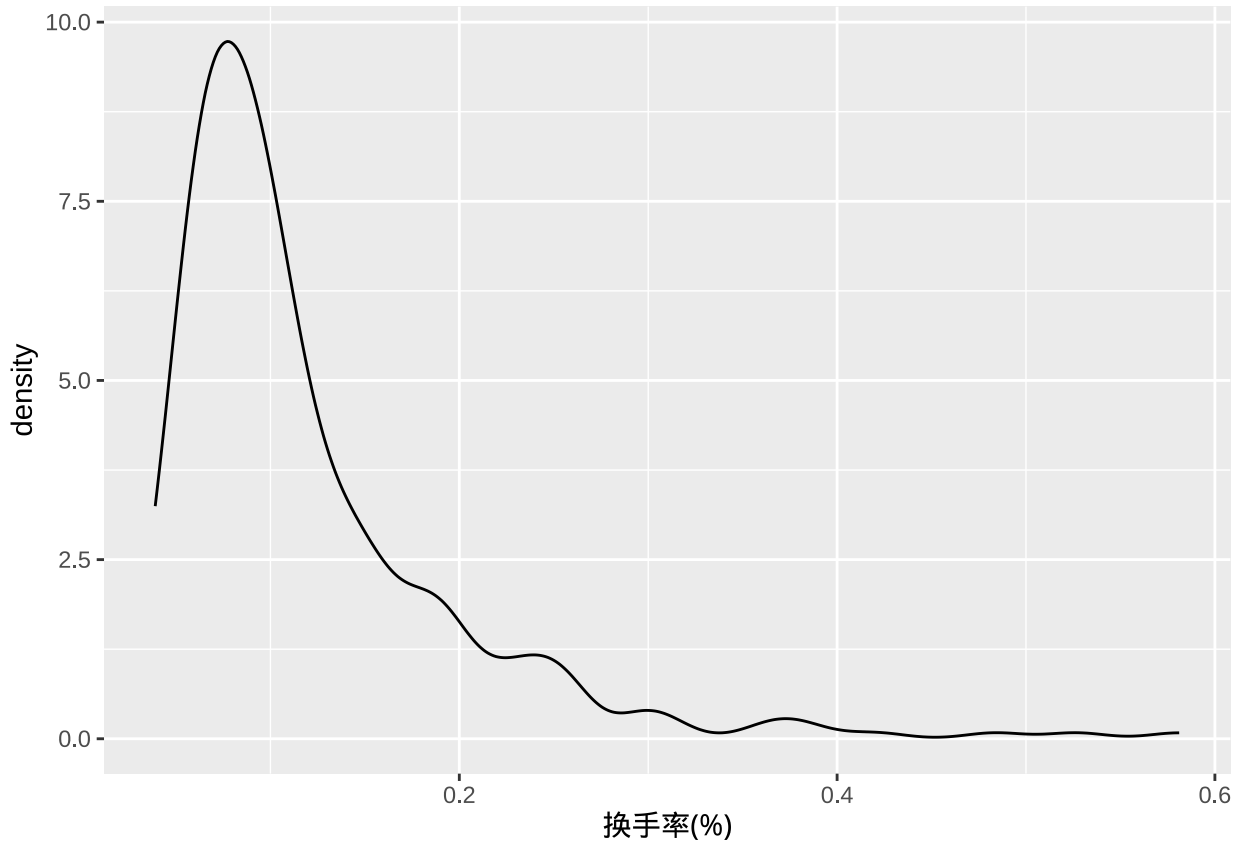
```
##   <chr> <chr> <date>      <dbl>    <dbl>  <dbl>    <dbl>  <dbl>    <dbl>   <dbl>
## 1 6000~ 浦发~ 2016-01-04   16.1     16.1   16.1     15.5   15.7 42240610  7.54e8
## 2 6000~ 浦发~ 2016-01-05   15.7     15.5   16.0     15.4   15.9 58054793  1.03e9
## 3 6000~ 浦发~ 2016-01-06   15.9     15.8   16.0     15.6   16.0 46772653  8.39e8
## 4 6000~ 浦发~ 2016-01-07   16.0     15.7   15.8     15.4   15.5 11350479  2.00e8
## 5 6000~ 浦发~ 2016-01-08   15.5     15.7   15.8     14.9   15.4 71918296  1.26e9
## 6 6000~ 浦发~ 2016-01-11   15.4     15.2   15.4     15.0   15.1 90177135  1.55e9
## # ... with 321 more rows, 8 more variables: `涨跌(元)` <dbl>,
## #   `涨跌幅(%)` <dbl>, `均价(元)` <dbl>, `换手率(%)` <dbl>,
## #   `A股流通市值(元)` <dbl>, `总市值(元)` <dbl>, `A股流通股本(股)` <dbl>,
## #   市盈率 <dbl>, and abbreviated variable names 1: `前收盘价(元)`,
## #   2: `开盘价(元)`, 3: `最高价(元)`, 4: `最低价(元)`, 5: `收盘价(元)`,
## #   6: `成交量(股)`, 7: `成交金额(元)`
```

**题目 66（数据可视化）：绘制换手率的密度曲线**

**难度：⋆⋆**

**代码及运行结果：**

```
df %>%
  ggplot(aes(`换手率 (%)`)) +
  geom_density()
```




题目 67（数据计算）：计算前一天与后一天收盘价的差值

难度：★★

代码及运行结果：

```
df %>%
  mutate(delta = `收盘价 (元)` - lag(`收盘价 (元)`)) %>%
  select(日期, `收盘价 (元)`, delta)
```

```
## # A tibble: 327 x 3
##    日期       `收盘价(元)`   delta
##    <date>            <dbl>   <dbl>
## 1 2016-01-04         15.7 NA
## 2 2016-01-05         15.9  0.141
## 3 2016-01-06         16.0  0.124
## 4 2016-01-07         15.5 -0.521
## 5 2016-01-08         15.4 -0.0177
## 6 2016-01-11         15.1 -0.389
## # ... with 321 more rows
```

题目 68（数据计算）：计算前一天与后一天收盘价的变化率

难度：★★

代码及运行结果：

```
df %>%
  mutate(change = (`收盘价 (元)` - lag(`收盘价 (元)`)) / `收盘价 (元)`) %>%
  select(日期, `收盘价 (元)`, change)
```

```
## # A tibble: 327 x 3
##    日期       `收盘价(元)`   change
##   <date>            <dbl>    <dbl>
## 1 2016-01-04         15.7 NA
## 2 2016-01-05         15.9  0.00891
## 3 2016-01-06         16.0  0.00774
## 4 2016-01-07         15.5 -0.0337
## 5 2016-01-08         15.4 -0.00115
## 6 2016-01-11         15.1 -0.0258
## # ... with 321 more rows
```

题目 69（数据操作）：设置日期为行索引

难度：★

代码及运行结果：

```
df %>%
  column_to_rownames(" 日期")   # 将从 tibble 变成 data.frame
```

```
##                  代码     简称 前收盘价(元) 开盘价(元) 最高价(元) 最低价(元)
## 2016-01-04 600000.SH 浦发银行      16.1356    16.1444    16.1444    15.4997
## 2016-01-05 600000.SH 浦发银行      15.7205    15.4644    15.9501    15.3672
## 2016-01-06 600000.SH 浦发银行      15.8618    15.8088    16.0208    15.6234
## 2016-01-07 600000.SH 浦发银行      15.9855    15.7205    15.8088    15.3672
## 2016-01-08 600000.SH 浦发银行      15.4644    15.6675    15.7912    14.9345
## 2016-01-11 600000.SH 浦发银行      15.4467    15.1994    15.4114    14.9786
## 2016-01-12 600000.SH 浦发银行      15.0581    15.1641    15.4732    15.0846
## 2016-01-13 600000.SH 浦发银行      15.4114    15.5174    15.8088    15.3231
## 2016-01-14 600000.SH 浦发银行      15.3584    15.0140    15.8883    14.9168
## 2016-01-15 600000.SH 浦发银行      15.8265    15.7205    16.0296    15.4732
## 2016-01-18 600000.SH 浦发银行      15.5262    15.3937    15.8883    15.2966
## 2016-01-19 600000.SH 浦发银行      15.6145    15.7029    15.7912    15.4556
## 2016-01-20 600000.SH 浦发银行      15.6940    15.5527    15.6322    15.0846
```

```
## 2016-01-21 600000.SH 浦发银行      15.2348    15.2083    15.4202    14.5724
## 2016-01-22 600000.SH 浦发银行      14.6430    14.8727    15.1111    14.5636
## 2016-01-25 600000.SH 浦发银行      15.0316    15.0140    15.1641    14.7667
## 2016-01-26 600000.SH 浦发银行      15.0405    14.9786    15.1376    14.4841
## 2016-01-27 600000.SH 浦发银行      14.7137    14.7932    14.9963    14.5812
## 2016-01-28 600000.SH 浦发银行      14.7314    14.8020    14.8550    14.0248
## 2016-01-29 600000.SH 浦发银行      14.6430    14.5901    15.0581    14.4929
## 2016-02-01 600000.SH 浦发银行      14.9257    15.0051    15.0140    14.4929
## 2016-02-02 600000.SH 浦发银行      14.5724    14.5724    14.9610    14.5371
## 2016-02-03 600000.SH 浦发银行      14.8020    14.6607    14.7137    14.5282
## 2016-02-04 600000.SH 浦发银行      14.6695    14.8727    15.2966    14.6872
## 2016-02-05 600000.SH 浦发银行      15.2789    15.2259    16.2416    15.1465
## 2016-02-15 600000.SH 浦发银行      16.2328    15.8795    16.3211    15.7205
## 2016-02-16 600000.SH 浦发银行      16.2946    16.2946    16.2946    16.2946
## 2016-02-17 600000.SH 浦发银行      16.2946    16.2946    16.2946    16.2946
## 2016-02-18 600000.SH 浦发银行      16.2946    16.2946    16.2946    16.2946
## 2016-02-19 600000.SH 浦发银行      16.2946    16.2946    16.2946    16.2946
## 2016-02-22 600000.SH 浦发银行      16.2946    16.2946    16.2946    16.2946
## 2016-02-23 600000.SH 浦发银行      16.2946    16.2946    16.2946    16.2946
## 2016-02-24 600000.SH 浦发银行      16.2946    16.2946    16.2946    16.2946
## 2016-02-25 600000.SH 浦发银行      16.2946    16.2946    16.2946    16.2946
## 2016-02-26 600000.SH 浦发银行      16.2946    16.2946    16.2946    16.2946
## 2016-02-29 600000.SH 浦发银行      16.2946    16.2946    16.2946    16.2946
## 2016-03-01 600000.SH 浦发银行      16.2946    16.2946    16.2946    16.2946
## 2016-03-02 600000.SH 浦发银行      16.2946    16.2946    16.2946    16.2946
## 2016-03-03 600000.SH 浦发银行      16.2946    16.2946    16.2946    16.2946
## 2016-03-04 600000.SH 浦发银行      16.2946    16.2946    16.2946    16.2946
## 2016-03-07 600000.SH 浦发银行      16.2946    16.2946    16.2946    16.2946
## 2016-03-08 600000.SH 浦发银行      16.2946    16.2946    16.2946    16.2946
## 2016-03-09 600000.SH 浦发银行      16.2946    16.2946    16.2946    16.2946
## 2016-03-10 600000.SH 浦发银行      16.2946    16.2946    16.2946    16.2946
## 2016-03-11 600000.SH 浦发银行      16.2946    15.4644    15.4821    14.6695
## 2016-03-14 600000.SH 浦发银行      14.9875    14.9080    15.4291    14.8550
## 2016-03-15 600000.SH 浦发银行      14.8903    14.9698    15.1818    14.8462
## 2016-03-16 600000.SH 浦发银行      15.0316    14.9433    15.9590    14.9345
## 2016-03-17 600000.SH 浦发银行      15.9236    15.7293    16.0650    15.4114
## 2016-03-18 600000.SH 浦发银行      15.6322    15.7117    15.8972    15.5880
## 2016-03-21 600000.SH 浦发银行      15.7558    15.7029    16.1179    15.7029
## 2016-03-22 600000.SH 浦发银行      15.9501    15.9766    16.4536    15.9148
```

```
## 2016-03-23 600000.SH 浦发银行      16.0296    16.0385    16.2063    15.9501
## 2016-03-24 600000.SH 浦发银行      16.0120    15.9943    15.9943    15.6145
## 2016-03-25 600000.SH 浦发银行      15.6499    15.6499    15.7823    15.6145
## 2016-03-28 600000.SH 浦发银行      15.6675    15.7029    15.8972    15.3407
## 2016-03-29 600000.SH 浦发银行      15.4114    15.4821    15.5969    15.3407
## 2016-03-30 600000.SH 浦发银行      15.4114    15.5880    15.8530    15.5880
## 2016-03-31 600000.SH 浦发银行      15.7823    15.8795    15.9678    15.6499
## 2016-04-01 600000.SH 浦发银行      15.8353    15.8795    16.1179    15.5527
## 2016-04-05 600000.SH 浦发银行      16.0561    15.8442    16.1533    15.6852
## 2016-04-06 600000.SH 浦发银行      16.0473    15.9236    16.0208    15.7823
## 2016-04-07 600000.SH 浦发银行      15.8618    15.8530    15.8972    15.5704
## 2016-04-08 600000.SH 浦发银行      15.6234    15.5086    15.6852    15.4821
## 2016-04-11 600000.SH 浦发银行      15.5439    15.5439    15.7382    15.4997
## 2016-04-12 600000.SH 浦发银行      15.6057    15.6234    15.6322    15.4821
## 2016-04-13 600000.SH 浦发银行      15.5615    15.6587    15.8883    15.6234
## 2016-04-14 600000.SH 浦发银行      15.6675    15.8088    15.9060    15.6764
## 2016-04-15 600000.SH 浦发银行      15.7205    15.8530    15.9501    15.7205
## 2016-04-18 600000.SH 浦发银行      15.8000    15.6764    15.9236    15.5969
## 2016-04-19 600000.SH 浦发银行      15.7293    15.8530    15.8795    15.7029
## 2016-04-20 600000.SH 浦发银行      15.8088    15.8000    15.8883    15.2878
## 2016-04-21 600000.SH 浦发银行      15.8618    15.7382    16.2063    15.6764
## 2016-04-22 600000.SH 浦发银行      16.0031    15.8972    16.2151    15.8442
## 2016-04-25 600000.SH 浦发银行      16.1091    16.0296    16.0385    15.7382
## 2016-04-26 600000.SH 浦发银行      15.7912    15.7823    15.9678    15.6764
## 2016-04-27 600000.SH 浦发银行      15.8265    15.8707    15.8707    15.7205
## 2016-04-28 600000.SH 浦发银行      15.7735    15.8353    15.9855    15.7470
## 2016-04-29 600000.SH 浦发银行      15.8088    15.7470    15.8442    15.6764
## 2016-05-03 600000.SH 浦发银行      15.7470    15.8530    16.1091    15.7470
## 2016-05-04 600000.SH 浦发银行      16.0031    15.9501    16.1003    15.9148
## 2016-05-05 600000.SH 浦发银行      15.9678    15.9501    16.0031    15.8795
## 2016-05-06 600000.SH 浦发银行      15.9501    15.9590    15.9855    15.6410
## 2016-05-09 600000.SH 浦发银行      15.6587    15.6410    15.7205    15.3672
## 2016-05-10 600000.SH 浦发银行      15.4644    15.3496    15.5086    15.3496
## 2016-05-11 600000.SH 浦发银行      15.4379    15.4997    15.5174    15.3672
## 2016-05-12 600000.SH 浦发银行      15.4114    15.3937    15.4556    15.1906
## 2016-05-13 600000.SH 浦发银行      15.3761    15.3407    15.4202    15.2083
## 2016-05-16 600000.SH 浦发银行      15.2259    15.2171    15.3231    15.1994
## 2016-05-17 600000.SH 浦发银行      15.2701    15.2701    15.3143    15.1553
## 2016-05-18 600000.SH 浦发银行      15.2348    15.2348    15.5615    15.0316
```

```
## 2016-05-19 600000.SH 浦发银行      15.5086    15.4026    15.4467    15.3054
## 2016-05-20 600000.SH 浦发银行      15.3231    15.2878    15.4556    15.2524
## 2016-05-23 600000.SH 浦发银行      15.4114    15.4556    15.6145    15.4026
## 2016-05-24 600000.SH 浦发银行      15.4997    15.4379    15.5439    15.3937
## 2016-05-25 600000.SH 浦发银行      15.4997    15.6145    15.6587    15.5439
## 2016-05-26 600000.SH 浦发银行      15.6057    15.5880    15.7735    15.5880
## 2016-05-27 600000.SH 浦发银行      15.6322    15.6234    15.6940    15.5704
## 2016-05-30 600000.SH 浦发银行      15.6675    15.7558    15.8707    15.6145
## 2016-05-31 600000.SH 浦发银行      15.8530    15.8353    16.2504    15.8265
## 2016-06-01 600000.SH 浦发银行      16.1533    16.1621    16.2063    16.0120
## 2016-06-02 600000.SH 浦发银行      16.0650    16.0738    16.1621    15.8353
## 2016-06-03 600000.SH 浦发银行      15.9148    16.0031    16.0120    15.8088
## 2016-06-06 600000.SH 浦发银行      15.9060    15.9413    15.9943    15.8707
## 2016-06-07 600000.SH 浦发银行      15.9148    15.9855    16.0120    15.8707
## 2016-06-08 600000.SH 浦发银行      15.9148    15.9060    15.9590    15.8707
## 2016-06-13 600000.SH 浦发银行      15.8883    15.7735    15.8442    15.6940
## 2016-06-14 600000.SH 浦发银行      15.7029    15.6940    15.7470    15.6322
## 2016-06-15 600000.SH 浦发银行      15.7117    15.5527    15.6852    15.4997
## 2016-06-16 600000.SH 浦发银行      15.6675    15.6057    15.8353    15.5439
## 2016-06-17 600000.SH 浦发银行      15.7205    15.6764    15.8883    15.6675
## 2016-06-20 600000.SH 浦发银行      15.6940    15.8353    15.8353    15.6764
## 2016-06-21 600000.SH 浦发银行      15.7647    15.7912    15.8530    15.6764
## 2016-06-22 600000.SH 浦发银行      15.7029    15.7205    15.8088    15.7029
## 2016-06-23 600000.SH 浦发银行      15.8000    15.9000    15.9000    15.7100
## 2016-06-24 600000.SH 浦发银行      15.7200    15.7000    15.7900    15.0800
## 2016-06-27 600000.SH 浦发银行      15.2800    15.2900    15.3900    15.1800
## 2016-06-28 600000.SH 浦发银行      15.3100    15.2700    15.3000    15.1800
## 2016-06-29 600000.SH 浦发银行      15.2800    15.2900    15.5400    15.2800
## 2016-06-30 600000.SH 浦发银行      15.5000    15.5000    15.6600    15.4400
## 2016-07-01 600000.SH 浦发银行      15.5700    15.5700    15.9600    15.5400
## 2016-07-04 600000.SH 浦发银行      15.9200    15.8600    16.0300    15.8000
## 2016-07-05 600000.SH 浦发银行      15.9300    15.9400    15.9500    15.7500
## 2016-07-06 600000.SH 浦发银行      15.7900    15.7100    15.7600    15.6100
## 2016-07-07 600000.SH 浦发银行      15.6400    15.6300    15.6400    15.4500
## 2016-07-08 600000.SH 浦发银行      15.5700    15.5500    15.5600    15.4400
## 2016-07-11 600000.SH 浦发银行      15.4500    15.5600    15.5700    15.4000
## 2016-07-12 600000.SH 浦发银行      15.4300    15.4800    15.6500    15.4500
## 2016-07-13 600000.SH 浦发银行      15.6400    15.6500    15.9700    15.6400
## 2016-07-14 600000.SH 浦发银行      15.7900    15.7700    15.8400    15.6400
```

```
## 2016-07-15 600000.SH 浦发银行      15.7000    15.7100    15.7500    15.6500
## 2016-07-18 600000.SH 浦发银行      15.7100    15.7300    15.9100    15.6600
## 2016-07-19 600000.SH 浦发银行      15.7700    15.7700    15.8000    15.6500
## 2016-07-20 600000.SH 浦发银行      15.7300    15.7000    15.7600    15.6300
## 2016-07-21 600000.SH 浦发银行      15.7500    15.7400    15.8700    15.7000
## 2016-07-22 600000.SH 浦发银行      15.8300    15.8100    15.8200    15.5800
## 2016-07-25 600000.SH 浦发银行      15.6300    15.6300    15.7600    15.6300
## 2016-07-26 600000.SH 浦发银行      15.6900    15.6900    15.7900    15.6500
## 2016-07-27 600000.SH 浦发银行      15.7400    15.7300    15.8800    15.4500
## 2016-07-28 600000.SH 浦发银行      15.8700    15.7700    15.7900    15.6500
## 2016-07-29 600000.SH 浦发银行      15.6800    15.6500    15.7100    15.6100
## 2016-08-01 600000.SH 浦发银行      15.7000    15.6800    15.8500    15.6500
## 2016-08-02 600000.SH 浦发银行      15.8000    15.7900    15.8500    15.6800
## 2016-08-03 600000.SH 浦发银行      15.7500    15.6400    15.7200    15.6200
## 2016-08-04 600000.SH 浦发银行      15.6700    15.6900    15.6900    15.4900
## 2016-08-05 600000.SH 浦发银行      15.5900    15.6000    15.8000    15.5900
## 2016-08-08 600000.SH 浦发银行      15.7300    15.7600    15.7600    15.6600
## 2016-08-09 600000.SH 浦发银行      15.7400    15.7500    15.8000    15.7100
## 2016-08-10 600000.SH 浦发银行      15.7900    15.8000    15.8500    15.7300
## 2016-08-11 600000.SH 浦发银行      15.8300    15.7500    16.4000    15.7400
## 2016-08-12 600000.SH 浦发银行      16.0700    16.1200    16.3900    16.0000
## 2016-08-15 600000.SH 浦发银行      16.3600    16.5000    17.0200    16.4500
## 2016-08-16 600000.SH 浦发银行      16.9400    16.9400    16.9900    16.6700
## 2016-08-17 600000.SH 浦发银行      16.7000    16.7200    16.8500    16.5200
## 2016-08-18 600000.SH 浦发银行      16.6400    16.5500    16.7700    16.3500
## 2016-08-19 600000.SH 浦发银行      16.4400    16.4300    16.5000    16.3600
## 2016-08-22 600000.SH 浦发银行      16.4200    16.4200    16.4700    16.3100
## 2016-08-23 600000.SH 浦发银行      16.4100    16.4100    16.5600    16.3200
## 2016-08-24 600000.SH 浦发银行      16.4000    16.4100    16.5100    16.3400
## 2016-08-25 600000.SH 浦发银行      16.3900    16.3600    16.4100    16.2000
## 2016-08-26 600000.SH 浦发银行      16.3400    16.3600    16.4000    16.2900
## 2016-08-29 600000.SH 浦发银行      16.3400    16.3100    16.3700    16.0800
## 2016-08-30 600000.SH 浦发银行      16.2300    16.2400    16.4900    16.2400
## 2016-08-31 600000.SH 浦发银行      16.4000    16.3500    16.5600    16.3200
## 2016-09-01 600000.SH 浦发银行      16.4800    16.4800    16.5500    16.3800
## 2016-09-02 600000.SH 浦发银行      16.4100    16.3800    16.5000    16.3500
## 2016-09-05 600000.SH 浦发银行      16.4800    16.4900    16.5800    16.4200
## 2016-09-06 600000.SH 浦发银行      16.4900    16.4900    16.5300    16.3700
## 2016-09-07 600000.SH 浦发银行      16.4100    16.4100    16.5000    16.3500
```

```
## 2016-09-08 600000.SH 浦发银行      16.4800    16.4700    16.6100    16.4300
## 2016-09-09 600000.SH 浦发银行      16.6000    16.6200    16.6600    16.5000
## 2016-09-12 600000.SH 浦发银行      16.5600    16.3800    16.4700    16.2600
## 2016-09-13 600000.SH 浦发银行      16.4000    16.4000    16.4600    16.3900
## 2016-09-14 600000.SH 浦发银行      16.4500    16.4300    16.4800    16.3800
## 2016-09-19 600000.SH 浦发银行      16.4000    16.4500    16.5100    16.4200
## 2016-09-20 600000.SH 浦发银行      16.4600    16.4700    16.4800    16.4200
## 2016-09-21 600000.SH 浦发银行      16.4600    16.4700    16.5100    16.4300
## 2016-09-22 600000.SH 浦发银行      16.4500    16.4900    16.5900    16.4700
## 2016-09-23 600000.SH 浦发银行      16.5300    16.5500    16.5500    16.4800
## 2016-09-26 600000.SH 浦发银行      16.4800    16.4500    16.5700    16.4100
## 2016-09-27 600000.SH 浦发银行      16.4800    16.4200    16.5200    16.4200
## 2016-09-28 600000.SH 浦发银行      16.4800    16.4800    16.5300    16.4600
## 2016-09-29 600000.SH 浦发银行      16.4800    16.5200    16.5500    16.4900
## 2016-09-30 600000.SH 浦发银行      16.5000    16.5000    16.5300    16.4800
## 2016-10-10 600000.SH 浦发银行      16.4900    16.5300    16.6000    16.4800
## 2016-10-11 600000.SH 浦发银行      16.5500    16.5700    16.5800    16.5000
## 2016-10-12 600000.SH 浦发银行      16.5700    16.5400    16.5400    16.4800
## 2016-10-13 600000.SH 浦发银行      16.5100    16.5100    16.5200    16.4200
## 2016-10-14 600000.SH 浦发银行      16.4300    16.4300    16.4700    16.3800
## 2016-10-17 600000.SH 浦发银行      16.4600    16.4100    16.4400    16.2200
## 2016-10-18 600000.SH 浦发银行      16.2700    16.2800    16.3900    16.2400
## 2016-10-19 600000.SH 浦发银行      16.3400    16.3500    16.3900    16.2400
## 2016-10-20 600000.SH 浦发银行      16.2700    16.2700    16.3400    16.2500
## 2016-10-21 600000.SH 浦发银行      16.2900    16.2900    16.3400    16.2200
## 2016-10-24 600000.SH 浦发银行      16.3000    16.3000    16.5800    16.2700
## 2016-10-25 600000.SH 浦发银行      16.4700    16.4800    16.5000    16.3600
## 2016-10-26 600000.SH 浦发银行      16.4200    16.4000    16.4200    16.3000
## 2016-10-27 600000.SH 浦发银行      16.3200    16.3000    16.3100    16.1800
## 2016-10-28 600000.SH 浦发银行      16.1900    16.1900    16.3400    16.1800
## 2016-10-31 600000.SH 浦发银行      16.2900    16.2300    16.3100    16.0100
## 2016-11-01 600000.SH 浦发银行      16.2700    16.2600    16.3300    16.2100
## 2016-11-02 600000.SH 浦发银行      16.3000    16.2100    16.3000    16.1800
## 2016-11-03 600000.SH 浦发银行      16.2000    16.1900    16.3800    16.1900
## 2016-11-04 600000.SH 浦发银行      16.3500    16.3000    16.5300    16.2900
## 2016-11-07 600000.SH 浦发银行      16.4000    16.3900    16.4500    16.3800
## 2016-11-08 600000.SH 浦发银行      16.4200    16.4500    16.5400    16.4400
## 2016-11-09 600000.SH 浦发银行      16.4700    16.4700    16.5300    16.3600
## 2016-11-10 600000.SH 浦发银行      16.4600    16.5500    16.6400    16.4800
```

```
## 2016-11-11 600000.SH 浦发银行      16.5700    16.5600    16.6300    16.4500
## 2016-11-14 600000.SH 浦发银行      16.5900    16.5300    16.7000    16.5200
## 2016-11-15 600000.SH 浦发银行      16.6000    16.5900    16.6100    16.5000
## 2016-11-16 600000.SH 浦发银行      16.5900    16.5900    16.6300    16.5400
## 2016-11-17 600000.SH 浦发银行      16.6000    16.5800    16.6300    16.5200
## 2016-11-18 600000.SH 浦发银行      16.6000    16.5900    16.6400    16.5500
## 2016-11-21 600000.SH 浦发银行      16.6000    16.5900    16.7300    16.5400
## 2016-11-22 600000.SH 浦发银行      16.6400    16.6700    16.7200    16.6300
## 2016-11-23 600000.SH 浦发银行      16.6800    16.6800    17.2300    16.6600
## 2016-11-24 600000.SH 浦发银行      16.9300    16.9300    17.1200    16.8600
## 2016-11-25 600000.SH 浦发银行      16.8900    16.8900    17.1000    16.8100
## 2016-11-28 600000.SH 浦发银行      17.0900    17.1500    17.4900    17.1500
## 2016-11-29 600000.SH 浦发银行      17.2100    17.2100    17.3500    17.1100
## 2016-11-30 600000.SH 浦发银行      17.2600    17.2800    17.4500    17.1000
## 2016-12-01 600000.SH 浦发银行      17.1600    17.1500    17.2700    17.0500
## 2016-12-02 600000.SH 浦发银行      17.1500    17.1000    17.1800    16.8000
## 2016-12-05 600000.SH 浦发银行      17.1000    16.9500    17.3200    16.9100
## 2016-12-06 600000.SH 浦发银行      17.3000    17.2800    17.4200    17.2000
## 2016-12-07 600000.SH 浦发银行      17.2200    17.1700    17.2000    17.0600
## 2016-12-08 600000.SH 浦发银行      17.0700    17.1500    17.2000    17.0400
## 2016-12-09 600000.SH 浦发银行      17.1400    17.1400    17.4600    17.1200
## 2016-12-12 600000.SH 浦发银行      17.3400    17.3400    17.5700    17.1900
## 2016-12-13 600000.SH 浦发银行      17.4900    17.3600    17.4400    17.1600
## 2016-12-14 600000.SH 浦发银行      17.1700    17.1700    17.4200    17.1200
## 2016-12-15 600000.SH 浦发银行      17.1800    17.1300    17.1300    16.7000
## 2016-12-16 600000.SH 浦发银行      16.7000    16.7000    16.8300    16.6300
## 2016-12-19 600000.SH 浦发银行      16.6600    16.6500    16.7000    16.5100
## 2016-12-20 600000.SH 浦发银行      16.5200    16.5200    16.5600    16.1200
## 2016-12-21 600000.SH 浦发银行      16.2800    16.2600    16.3700    16.2500
## 2016-12-22 600000.SH 浦发银行      16.3200    16.3200    16.3300    16.1400
## 2016-12-23 600000.SH 浦发银行      16.1900    16.1700    16.2200    16.0700
## 2016-12-26 600000.SH 浦发银行      16.1900    16.1300    16.2800    16.0100
## 2016-12-27 600000.SH 浦发银行      16.2500    16.2500    16.3200    16.1300
## 2016-12-28 600000.SH 浦发银行      16.1400    16.1400    16.1900    16.0300
## 2016-12-29 600000.SH 浦发银行      16.0900    16.0900    16.1500    15.9900
## 2016-12-30 600000.SH 浦发银行      16.0700    16.0700    16.2300    16.0400
## 2017-01-03 600000.SH 浦发银行      16.2100    16.2100    16.4400    16.1700
## 2017-01-04 600000.SH 浦发银行      16.3000    16.2900    16.3500    16.1800
## 2017-01-05 600000.SH 浦发银行      16.3300    16.3000    16.3800    16.2400
```

```
## 2017-01-06 600000.SH 浦发银行      16.3000    16.3000    16.3000    16.1300
## 2017-01-09 600000.SH 浦发银行      16.1800    16.2400    16.2900    16.1300
## 2017-01-10 600000.SH 浦发银行      16.2000    16.1800    16.2400    16.1400
## 2017-01-11 600000.SH 浦发银行      16.1900    16.2400    16.2400    16.1500
## 2017-01-12 600000.SH 浦发银行      16.1600    16.1800    16.2000    16.1100
## 2017-01-13 600000.SH 浦发银行      16.1200    16.1000    16.2900    16.1000
## 2017-01-16 600000.SH 浦发银行      16.2700    16.2300    16.6000    16.1000
## 2017-01-17 600000.SH 浦发银行      16.5600    16.4600    16.5400    16.3700
## 2017-01-18 600000.SH 浦发银行      16.4000    16.4200    16.5500    16.3600
## 2017-01-19 600000.SH 浦发银行      16.4800    16.4300    16.6400    16.4300
## 2017-01-20 600000.SH 浦发银行      16.5400    16.5800    16.6600    16.5000
## 2017-01-23 600000.SH 浦发银行      16.6000    16.6600    16.6900    16.5100
## 2017-01-24 600000.SH 浦发银行      16.5700    16.5800    16.7000    16.5800
## 2017-01-25 600000.SH 浦发银行      16.6900    16.6900    16.7400    16.6100
## 2017-01-26 600000.SH 浦发银行      16.6900    16.6900    16.8400    16.6100
## 2017-02-03 600000.SH 浦发银行      16.7400    16.8200    16.8500    16.6200
## 2017-02-06 600000.SH 浦发银行      16.6300    16.7500    16.7800    16.6600
## 2017-02-07 600000.SH 浦发银行      16.6600    16.6500    16.7400    16.6300
## 2017-02-08 600000.SH 浦发银行      16.6700    16.6300    16.6900    16.5400
## 2017-02-09 600000.SH 浦发银行      16.6700    16.6800    16.7700    16.5900
## 2017-02-10 600000.SH 浦发银行      16.7200    16.7600    16.8400    16.7000
## 2017-02-13 600000.SH 浦发银行      16.7800    16.8800    16.9000    16.7800
## 2017-02-14 600000.SH 浦发银行      16.8500    16.8700    16.8800    16.7400
## 2017-02-15 600000.SH 浦发银行      16.7500    16.8200    16.9300    16.7500
## 2017-02-16 600000.SH 浦发银行      16.8400    16.8800    16.9100    16.7600
## 2017-02-17 600000.SH 浦发银行      16.7800    16.7800    16.8200    16.5900
## 2017-02-20 600000.SH 浦发银行      16.6400    16.6200    16.9800    16.6200
## 2017-02-21 600000.SH 浦发银行      16.9100    16.8800    17.0800    16.8100
## 2017-02-22 600000.SH 浦发银行      16.8800    16.8800    16.9000    16.7300
## 2017-02-23 600000.SH 浦发银行      16.7500    16.7800    16.8300    16.6400
## 2017-02-24 600000.SH 浦发银行      16.6900    16.6700    16.7500    16.6600
## 2017-02-27 600000.SH 浦发银行      16.7100    16.6900    16.7200    16.5300
## 2017-02-28 600000.SH 浦发银行      16.5900    16.5800    16.6700    16.5300
## 2017-03-01 600000.SH 浦发银行      16.5900    16.5800    16.6200    16.5200
## 2017-03-02 600000.SH 浦发银行      16.5600    16.6200    16.6200    16.3900
## 2017-03-03 600000.SH 浦发银行      16.4100    16.4200    16.4300    16.3200
## 2017-03-06 600000.SH 浦发银行      16.3500    16.3700    16.5300    16.3500
## 2017-03-07 600000.SH 浦发银行      16.4100    16.3800    16.4500    16.3700
## 2017-03-08 600000.SH 浦发银行      16.4300    16.4000    16.4400    16.3500
```

```
## 2017-03-09 600000.SH 浦发银行      16.3900    16.3700    16.4000    16.2200
## 2017-03-10 600000.SH 浦发银行      16.2200    16.2300    16.2800    16.1700
## 2017-03-13 600000.SH 浦发银行      16.2300    16.2300    16.3400    16.1600
## 2017-03-14 600000.SH 浦发银行      16.3400    16.3400    16.3500    16.2400
## 2017-03-15 600000.SH 浦发银行      16.2600    16.2400    16.2800    16.1700
## 2017-03-16 600000.SH 浦发银行      16.2400    16.2700    16.3500    16.2500
## 2017-03-17 600000.SH 浦发银行      16.2900    16.3000    16.3200    16.1400
## 2017-03-20 600000.SH 浦发银行      16.2000    16.2400    16.2500    16.1300
## 2017-03-21 600000.SH 浦发银行      16.1400    16.1500    16.1600    15.9800
## 2017-03-22 600000.SH 浦发银行      16.0000    15.9800    15.9800    15.7000
## 2017-03-23 600000.SH 浦发银行      15.7800    15.8000    15.9200    15.7900
## 2017-03-24 600000.SH 浦发银行      15.8800    15.8500    16.0000    15.8300
## 2017-03-27 600000.SH 浦发银行      15.9600    15.9700    16.1300    15.9000
## 2017-03-28 600000.SH 浦发银行      16.0400    16.1100    16.1300    15.9700
## 2017-03-29 600000.SH 浦发银行      16.0100    16.0100    16.1100    15.8000
## 2017-03-30 600000.SH 浦发银行      15.8700    15.8800    15.9000    15.7200
## 2017-03-31 600000.SH 浦发银行      15.8100    15.7800    16.0500    15.7700
## 2017-04-05 600000.SH 浦发银行      16.0100    16.0500    16.2000    15.8900
## 2017-04-06 600000.SH 浦发银行      16.1600    16.0900    16.1800    16.0300
## 2017-04-07 600000.SH 浦发银行      16.1000    16.0600    16.1600    16.0200
## 2017-04-10 600000.SH 浦发银行      16.1000    16.1200    16.1200    15.9400
## 2017-04-11 600000.SH 浦发银行      15.9700    15.9700    15.9900    15.8100
## 2017-04-12 600000.SH 浦发银行      15.8900    15.8800    15.9600    15.7500
## 2017-04-13 600000.SH 浦发银行      15.8900    15.8800    15.8900    15.7300
## 2017-04-14 600000.SH 浦发银行      15.7500    15.7600    15.7800    15.4200
## 2017-04-17 600000.SH 浦发银行      15.5100    15.5300    15.6000    15.3000
## 2017-04-18 600000.SH 浦发银行      15.5600    15.5000    15.5200    15.3000
## 2017-04-19 600000.SH 浦发银行      15.3000    15.2500    15.2700    15.1100
## 2017-04-20 600000.SH 浦发银行      15.1100    15.1200    15.1400    14.9000
## 2017-04-21 600000.SH 浦发银行      14.9100    14.9200    15.0700    14.8500
## 2017-04-24 600000.SH 浦发银行      15.0500    15.0500    15.1100    14.9100
## 2017-04-25 600000.SH 浦发银行      15.0000    15.0200    15.1000    14.9900
## 2017-04-26 600000.SH 浦发银行      15.0500    15.0600    15.1100    15.0000
## 2017-04-27 600000.SH 浦发银行      15.0500    15.0500    15.2500    15.0300
## 2017-04-28 600000.SH 浦发银行      15.2100    15.1500    15.2200    15.0800
## 2017-05-02 600000.SH 浦发银行      15.2100    15.2100    15.2200    15.1300
## 2017-05-03 600000.SH 浦发银行      15.1600    15.1600    15.1600    15.0500
## 2017-05-04 600000.SH 浦发银行      15.0800    15.0700    15.0700    14.9000
## 2017-05-05 600000.SH 浦发银行      14.9800    14.9500    14.9800    14.5200
```

```
## 2017-05-08 600000.SH 浦发银行      14.9200    14.7800    14.9000    14.5100
## 2017-05-09 600000.SH 浦发银行      14.8600    14.6900    14.8400    14.6600
##            收盘价(元) 成交量(股) 成交金额(元) 涨跌(元) 涨跌幅(%) 均价(元)
## 2016-01-04    15.7205   42240610    754425783  -0.4151   -2.5725  17.8602
## 2016-01-05    15.8618   58054793   1034181474   0.1413    0.8989  17.8139
## 2016-01-06    15.9855   46772653    838667398   0.1236    0.7795  17.9307
## 2016-01-07    15.4644   11350479    199502702  -0.5211   -3.2597  17.5766
## 2016-01-08    15.4467   71918296   1262105060  -0.0177   -0.1142  17.5492
## 2016-01-11    15.0581   90177135   1550155933  -0.3886   -2.5157  17.1901
## 2016-01-12    15.4114   55374454    964061502   0.3533    2.3460  17.4099
## 2016-01-13    15.3584   47869312    843717365  -0.0530   -0.3438  17.6254
## 2016-01-14    15.8265   54838833    966117848   0.4681    3.0477  17.6174
## 2016-01-15    15.5262   46723139    836146426  -0.3003   -1.8973  17.8958
## 2016-01-18    15.6145   32729006    583291559   0.0883    0.5688  17.8219
## 2016-01-19    15.6940   29807159    527753175   0.0795    0.5090  17.7056
## 2016-01-20    15.2348   35968636    623546924  -0.4593   -2.9263  17.3359
## 2016-01-21    14.6430   34197115    582845476  -0.5917   -3.8841  17.0437
## 2016-01-22    15.0316   42007318    708179241   0.3886    2.6538  16.8585
## 2016-01-25    15.0405   23558971    400445071   0.0088    0.0588  16.9976
## 2016-01-26    14.7137   38279766    643611762  -0.3268   -2.1726  16.8134
## 2016-01-27    14.7314   44291307    742640743   0.0177    0.1200  16.7672
## 2016-01-28    14.6430   38902180    643984171  -0.0883   -0.5995  16.5539
## 2016-01-29    14.9257   47429000    799493381   0.2826    1.9300  16.8566
## 2016-02-01    14.5724   29139596    485275132  -0.3533   -2.3669  16.6535
## 2016-02-02    14.8020   25380981    425396889   0.2296    1.5758  16.7605
## 2016-02-03    14.6695   27779183    459255616  -0.1325   -0.8950  16.5324
## 2016-02-04    15.2789   47162860    799283799   0.6094    4.1541  16.9473
## 2016-02-05    16.2328   78413454   1401156041   0.9538    6.2428  17.8688
## 2016-02-15    16.2946   67296317   1221398339   0.0618    0.3808  18.1496
## 2016-02-16    16.2946         NA           NA   0.0000    0.0000        NA
## 2016-02-17    16.2946         NA           NA   0.0000    0.0000        NA
## 2016-02-18    16.2946         NA           NA   0.0000    0.0000        NA
## 2016-02-19    16.2946         NA           NA   0.0000    0.0000        NA
## 2016-02-22    16.2946         NA           NA   0.0000    0.0000        NA
## 2016-02-23    16.2946         NA           NA   0.0000    0.0000        NA
## 2016-02-24    16.2946         NA           NA   0.0000    0.0000        NA
## 2016-02-25    16.2946         NA           NA   0.0000    0.0000        NA
## 2016-02-26    16.2946         NA           NA   0.0000    0.0000        NA
## 2016-02-29    16.2946         NA           NA   0.0000    0.0000        NA
```

```
## 2016-03-01    16.2946         NA           NA   0.0000    0.0000       NA
## 2016-03-02    16.2946         NA           NA   0.0000    0.0000       NA
## 2016-03-03    16.2946         NA           NA   0.0000    0.0000       NA
## 2016-03-04    16.2946         NA           NA   0.0000    0.0000       NA
## 2016-03-07    16.2946         NA           NA   0.0000    0.0000       NA
## 2016-03-08    16.2946         NA           NA   0.0000    0.0000       NA
## 2016-03-09    16.2946         NA           NA   0.0000    0.0000       NA
## 2016-03-10    16.2946         NA           NA   0.0000    0.0000       NA
## 2016-03-11    14.9875   98240322   1657158636  -1.3071   -8.0217  16.8684
## 2016-03-14    14.8903   56922239    973317374  -0.0971   -0.6482  17.0991
## 2016-03-15    15.0316   37540932    638223050   0.1413    0.9490  17.0007
## 2016-03-16    15.9236  108404378   1924212601   0.8920    5.9342  17.7503
## 2016-03-17    15.6322   68635812   1219236346  -0.2914   -1.8303  17.7639
## 2016-03-18    15.7558   55987968    996332597   0.1236    0.7910  17.7955
## 2016-03-21    15.9501   42001532    756966797   0.1943    1.2332  18.0224
## 2016-03-22    16.0296   43760811    800814893   0.0795    0.4983  18.2998
## 2016-03-23    16.0120   27830795    506023215  -0.0177   -0.1102  18.1821
## 2016-03-24    15.6499   27448272    488313660  -0.3621   -2.2614  17.7903
## 2016-03-25    15.6675   13619185    241971869   0.0177    0.1129  17.7670
## 2016-03-28    15.4114   24521696    433449200  -0.2561   -1.6347  17.6762
## 2016-03-29    15.4114   19151152    335806206   0.0000    0.0000  17.5345
## 2016-03-30    15.7823   35562148    634175207   0.3709    2.4069  17.8329
## 2016-03-31    15.8353   23762112    425380429   0.0530    0.3358  17.9016
## 2016-04-01    16.0561   35790541    643979468   0.2208    1.3943  17.9930
## 2016-04-05    16.0473   42622297    768010128  -0.0088   -0.0550  18.0190
## 2016-04-06    15.8618   23428724    421234846  -0.1855   -1.1558  17.9794
## 2016-04-07    15.6234   29047713    516129852  -0.2385   -1.5033  17.7683
## 2016-04-08    15.5439   17132244    301704521  -0.0795   -0.5088  17.6103
## 2016-04-11    15.6057   24117635    426862387   0.0618    0.3977  17.6992
## 2016-04-12    15.5615   14100759    248002295  -0.0442   -0.2830  17.5879
## 2016-04-13    15.6675   35189716    627667715   0.1060    0.6810  17.8367
## 2016-04-14    15.7205   15796208    281977244   0.0530    0.3382  17.8509
## 2016-04-15    15.8000   34773526    623836222   0.0795    0.5056  17.9400
## 2016-04-18    15.7293   28084133    501790676  -0.0707   -0.4472  17.8674
## 2016-04-19    15.8088   27246371    487095912   0.0795    0.5053  17.8775
## 2016-04-20    15.8618   46614611    825437834   0.0530    0.3352  17.7077
## 2016-04-21    16.0031   48212915    872382605   0.1413    0.8909  18.0944
## 2016-04-22    16.1091   28178162    513194655   0.1060    0.6623  18.2125
## 2016-04-25    15.7912   24294360    435467860  -0.3179   -1.9737  17.9246
```

```
## 2016-04-26    15.8265   15739400    281501388   0.0353    0.2237  17.8851
## 2016-04-27    15.7735   13838875    247622905  -0.0530   -0.3348  17.8933
## 2016-04-28    15.8088   15728417    282201353   0.0353    0.2240  17.9421
## 2016-04-29    15.7470   23065933    412460987  -0.0618   -0.3911  17.8818
## 2016-05-03    16.0031   27776723    501830132   0.2561    1.6265  18.0666
## 2016-05-04    15.9678   23118959    418592994  -0.0353   -0.2208  18.1060
## 2016-05-05    15.9501   14936632    269576075  -0.0177   -0.1106  18.0480
## 2016-05-06    15.6587   23406763    418454827  -0.2914   -1.8272  17.8775
## 2016-05-09    15.4644   21947591    386423417  -0.1943   -1.2408  17.6066
## 2016-05-10    15.4379   16846648    294675478  -0.0265   -0.1713  17.4916
## 2016-05-11    15.4114   11941164    208655785  -0.0265   -0.1716  17.4737
## 2016-05-12    15.3761   14668954    254410730  -0.0353   -0.2292  17.3435
## 2016-05-13    15.2259   11013829    190929402  -0.1501   -0.9765  17.3354
## 2016-05-16    15.2701   11045624    190872655   0.0442    0.2900  17.2804
## 2016-05-17    15.2348    9765153    168387277  -0.0353   -0.2313  17.2437
## 2016-05-18    15.5086   30962453    536935470   0.2738    1.7971  17.3415
## 2016-05-19    15.3231   10380826    180686332  -0.1855   -1.1959  17.4058
## 2016-05-20    15.4114   11140621    194223009   0.0883    0.5764  17.4338
## 2016-05-23    15.4997   14080659    247441303   0.0883    0.5731  17.5731
## 2016-05-24    15.4997   10725938    187905438   0.0000    0.0000  17.5188
## 2016-05-25    15.6057   13766238    243549455   0.1060    0.6838  17.6918
## 2016-05-26    15.6322   11065062    196266938   0.0265    0.1698  17.7375
## 2016-05-27    15.6675   11621242    205801175   0.0353    0.2260  17.7091
## 2016-05-30    15.8530   17538184    313324795   0.1855    1.1838  17.8653
## 2016-05-31    16.1533   35631526    650177320   0.3003    1.8942  18.2472
## 2016-06-01    16.0650   18312424    333790682  -0.0883   -0.5467  18.2276
## 2016-06-02    15.9148   17786525    321768768  -0.1501   -0.9346  18.0906
## 2016-06-03    15.9060   18003512    323746001  -0.0088   -0.0555  17.9824
## 2016-06-06    15.9148   17856247    321865763   0.0088    0.0555  18.0254
## 2016-06-07    15.9148   16533983    298145352   0.0000    0.0000  18.0323
## 2016-06-08    15.8883   23758300    427741583  -0.0265   -0.1665  18.0039
## 2016-06-13    15.7029   30051583    536281150  -0.1855   -1.1673  17.8454
## 2016-06-14    15.7117   18773357    333449719   0.0088    0.0562  17.7619
## 2016-06-15    15.6675   26332177    464557383  -0.0442   -0.2811  17.6422
## 2016-06-16    15.7205   30871127    549171213   0.0530    0.3382  17.7892
## 2016-06-17    15.6940   17412233    311113072  -0.0265   -0.1685  17.8675
## 2016-06-20    15.7647    9681495    172507202   0.0707    0.4502  17.8182
## 2016-06-21    15.7029   15852873    282663680  -0.0618   -0.3922  17.8304
## 2016-06-22    15.8000   14227501    253800359   0.0971    0.6187  17.8387
```

```
## 2016-06-23    15.7200   12882295    203560849  -0.0800   -0.5063  15.8016
## 2016-06-24    15.2800   30575522    470756348  -0.4400   -2.7990  15.3965
## 2016-06-27    15.3100   19163499    292488228   0.0300    0.1963  15.2628
## 2016-06-28    15.2800   12559826    191346962  -0.0300   -0.1960  15.2348
## 2016-06-29    15.5000   21253351    326320213   0.2200    1.4398  15.3538
## 2016-06-30    15.5700   20735886    323223547   0.0700    0.4516  15.5876
## 2016-07-01    15.9200   25106479    394802033   0.3500    2.2479  15.7251
## 2016-07-04    15.9300   21411650    341120776   0.0100    0.0628  15.9316
## 2016-07-05    15.7900   16171615    255727223  -0.1400   -0.8788  15.8133
## 2016-07-06    15.6400   22976510    360070168  -0.1500   -0.9500  15.6712
## 2016-07-07    15.5700   24498042    380536334  -0.0700   -0.4476  15.5333
## 2016-07-08    15.4500   14901331    230568191  -0.1200   -0.7707  15.4730
## 2016-07-11    15.4300   22852491    354003568  -0.0200   -0.1294  15.4908
## 2016-07-12    15.6400   25545130    397267013   0.2100    1.3610  15.5516
## 2016-07-13    15.7900   36238056    573546775   0.1500    0.9591  15.8272
## 2016-07-14    15.7000   16883421    265364888  -0.0900   -0.5700  15.7175
## 2016-07-15    15.7100   15998100    251152944   0.0100    0.0637  15.6989
## 2016-07-18    15.7700   27766982    438169692   0.0600    0.3819  15.7802
## 2016-07-19    15.7300   17179816    269908287  -0.0400   -0.2536  15.7108
## 2016-07-20    15.7500   14866526    233284416   0.0200    0.1271  15.6919
## 2016-07-21    15.8300   22020445    348156592   0.0800    0.5079  15.8106
## 2016-07-22    15.6300   16711700    262066573  -0.2000   -1.2634  15.6816
## 2016-07-25    15.6900   11441970    179598105   0.0600    0.3839  15.6964
## 2016-07-26    15.7400   20462397    321598665   0.0500    0.3187  15.7166
## 2016-07-27    15.8700   40697330    637937308   0.1300    0.8259  15.6752
## 2016-07-28    15.6800   17451372    274345301  -0.1900   -1.1972  15.7206
## 2016-07-29    15.7000   14348301    224747562   0.0200    0.1276  15.6637
## 2016-08-01    15.8000   22909943    361191350   0.1000    0.6369  15.7657
## 2016-08-02    15.7500   10877815    171311340  -0.0500   -0.3165  15.7487
## 2016-08-03    15.6700   10583990    165699514  -0.0800   -0.5079  15.6557
## 2016-08-04    15.5900   15610498    243122631  -0.0800   -0.5105  15.5743
## 2016-08-05    15.7300   17648517    277447198   0.1400    0.8980  15.7207
## 2016-08-08    15.7400   13129367    206333087   0.0100    0.0636  15.7154
## 2016-08-09    15.7900   15480879    243949588   0.0500    0.3177  15.7581
## 2016-08-10    15.8300   18499535    292534452   0.0400    0.2533  15.8131
## 2016-08-11    16.0700   52689421    852413099   0.2400    1.5161  16.1781
## 2016-08-12    16.3600   46991632    763634354   0.2900    1.8046  16.2504
## 2016-08-15    16.9400   77333862   1299618491   0.5800    3.5452  16.8053
## 2016-08-16    16.7000   49384036    829743468  -0.2400   -1.4168  16.8019
```

```
## 2016-08-17    16.6400   36446597    606581272  -0.0600   -0.3593  16.6430
## 2016-08-18    16.4400   37110139    612294413  -0.2000   -1.2019  16.4994
## 2016-08-19    16.4200   19927300    327521631  -0.0200   -0.1217  16.4358
## 2016-08-22    16.4100   19768630    324280093  -0.0100   -0.0609  16.4038
## 2016-08-23    16.4000   31754370    521794934  -0.0100   -0.0609  16.4322
## 2016-08-24    16.3900   17085424    280402637  -0.0100   -0.0610  16.4118
## 2016-08-25    16.3400   24749915    403472216  -0.0500   -0.3051  16.3020
## 2016-08-26    16.3400   14688056    240035889   0.0000    0.0000  16.3423
## 2016-08-29    16.2300   30202023    489414008  -0.1100   -0.6732  16.2047
## 2016-08-30    16.4000   36396993    597418078   0.1700    1.0474  16.4139
## 2016-08-31    16.4800   22003438    362526025   0.0800    0.4878  16.4759
## 2016-09-01    16.4100   21272047    349934548  -0.0700   -0.4248  16.4504
## 2016-09-02    16.4800   23147001    380418559   0.0700    0.4266  16.4349
## 2016-09-05    16.4900   25677035    423399290   0.0100    0.0607  16.4894
## 2016-09-06    16.4100   20282647    333206432  -0.0800   -0.4851  16.4282
## 2016-09-07    16.4800   17975558    295519896   0.0700    0.4266  16.4401
## 2016-09-08    16.6000   24406900    403801797   0.1200    0.7282  16.5446
## 2016-09-09    16.5600   18739471    310623908  -0.0400   -0.2410  16.5759
## 2016-09-12    16.4000   36509177    597895474  -0.1600   -0.9662  16.3766
## 2016-09-13    16.4500   14234378    233969071   0.0500    0.3049  16.4369
## 2016-09-14    16.4000   23981413    393967822  -0.0500   -0.3040  16.4280
## 2016-09-19    16.4600   14465955    238091122   0.0600    0.3659  16.4587
## 2016-09-20    16.4600   17396811    286078658   0.0000    0.0000  16.4443
## 2016-09-21    16.4500    9416102    154918234  -0.0100   -0.0608  16.4525
## 2016-09-22    16.5300   11527832    190442429   0.0800    0.4863  16.5202
## 2016-09-23    16.4800   10200460    168270146  -0.0500   -0.3025  16.4963
## 2016-09-26    16.4800   20769186    342748706   0.0000    0.0000  16.5028
## 2016-09-27    16.4800   17025470    280852907   0.0000    0.0000  16.4960
## 2016-09-28    16.4800   12841990    211844619   0.0000    0.0000  16.4962
## 2016-09-29    16.5000   10997408    181620111   0.0200    0.1214  16.5148
## 2016-09-30    16.4900   12190268    201085204  -0.0100   -0.0606  16.4956
## 2016-10-10    16.5500   18666854    308450225   0.0600    0.3639  16.5240
## 2016-10-11    16.5700   12690366    209785537   0.0200    0.1208  16.5311
## 2016-10-12    16.5100   10643884    175594201  -0.0600   -0.3621  16.4972
## 2016-10-13    16.4300   14504356    238973130  -0.0800   -0.4846  16.4760
## 2016-10-14    16.4600   10660114    175020225   0.0300    0.1826  16.4182
## 2016-10-17    16.2700   16424904    268130860  -0.1900   -1.1543  16.3247
## 2016-10-18    16.3400   18251833    298064113   0.0700    0.4302  16.3306
## 2016-10-19    16.2700   11143167    181631901  -0.0700   -0.4284  16.2998
```

```
## 2016-10-20    16.2900    8985794    146255319    0.0200    0.1229  16.2763
## 2016-10-21    16.3000   10884798    177155062    0.0100    0.0614  16.2755
## 2016-10-24    16.4700   24757241    406980667    0.1700    1.0429  16.4389
## 2016-10-25    16.4200   13309321    218721453   -0.0500   -0.3036  16.4337
## 2016-10-26    16.3200   12436805    203174010   -0.1000   -0.6090  16.3365
## 2016-10-27    16.1900   19517454    316513318   -0.1300   -0.7966  16.2169
## 2016-10-28    16.2900   15067204    244739683    0.1000    0.6177  16.2432
## 2016-10-31    16.2700   14673298    237635070   -0.0200   -0.1228  16.1951
## 2016-11-01    16.3000   13028259    212179107    0.0300    0.1844  16.2861
## 2016-11-02    16.2000   25051160    406667269   -0.1000   -0.6135  16.2335
## 2016-11-03    16.3500   25864110    421412550    0.1500    0.9259  16.2933
## 2016-11-04    16.4000   21516475    353348691    0.0500    0.3058  16.4222
## 2016-11-07    16.4200   21351658    350381736    0.0200    0.1220  16.4100
## 2016-11-08    16.4700   17307155    285419770    0.0500    0.3045  16.4914
## 2016-11-09    16.4600   39826582    655055488   -0.0100   -0.0607  16.4477
## 2016-11-10    16.5700   20484554    339602785    0.1100    0.6683  16.5785
## 2016-11-11    16.5900   21539188    356740636    0.0200    0.1207  16.5624
## 2016-11-14    16.6000   18124496    301208006    0.0100    0.0603  16.6188
## 2016-11-15    16.5900   15714142    260648299   -0.0100   -0.0602  16.5869
## 2016-11-16    16.6000   12395504    205688793    0.0100    0.0603  16.5938
## 2016-11-17    16.6000   22718556    376599274    0.0000    0.0000  16.5767
## 2016-11-18    16.6000   29274383    485488349    0.0000    0.0000  16.5841
## 2016-11-21    16.6400   21432453    356612260    0.0400    0.2410  16.6389
## 2016-11-22    16.6800   21080968    351666631    0.0400    0.2404  16.6817
## 2016-11-23    16.9300   45036427    765689745    0.2500    1.4988  17.0016
## 2016-11-24    16.8900   21043140    356783606   -0.0400   -0.2363  16.9549
## 2016-11-25    17.0900   23335230    396231886    0.2000    1.1841  16.9800
## 2016-11-28    17.2100   33835281    584558233    0.1200    0.7022  17.2766
## 2016-11-29    17.2600   29530098    509203656    0.0500    0.2905  17.2435
## 2016-11-30    17.1600   20135026    346784527   -0.1000   -0.5794  17.2229
## 2016-12-01    17.1500   18449219    316391192   -0.0100   -0.0583  17.1493
## 2016-12-02    17.1000   33895875    576329869   -0.0500   -0.2915  17.0030
## 2016-12-05    17.3000   38201368    655302810    0.2000    1.1696  17.1539
## 2016-12-06    17.2200   20948075    362989434   -0.0800   -0.4624  17.3281
## 2016-12-07    17.0700   15550748    265825550   -0.1500   -0.8711  17.0941
## 2016-12-08    17.1400   13234569    226759229    0.0700    0.4101  17.1339
## 2016-12-09    17.3400   28835648    500164630    0.2000    1.1669  17.3454
## 2016-12-12    17.4900   48359554    841905183    0.1500    0.8651  17.4093
## 2016-12-13    17.1700   15602869    270112439   -0.3200   -1.8296  17.3117
```

```
## 2016-12-14    17.1800   17860552    309185673    0.0100     0.0582  17.3111
## 2016-12-15    16.7000   30809613    517398943   -0.4800    -2.7939  16.7934
## 2016-12-16    16.6600   14531366    242842108   -0.0400    -0.2395  16.7116
## 2016-12-19    16.5200    8483493    140430889   -0.1400    -0.8403  16.5534
## 2016-12-20    16.2800   22525817    366213562   -0.2400    -1.4528  16.2575
## 2016-12-21    16.3200   12638809    206022869    0.0400     0.2457  16.3008
## 2016-12-22    16.1900   11499190    186428629   -0.1300    -0.7966  16.2123
## 2016-12-23    16.1900   11187906    180365980    0.0000     0.0000  16.1215
## 2016-12-26    16.2500   14482456    233999583    0.0600     0.3706  16.1575
## 2016-12-27    16.1400   10665824    172900199   -0.1100    -0.6769  16.2107
## 2016-12-28    16.0900   14441526    232406679   -0.0500    -0.3098  16.0929
## 2016-12-29    16.0700   11851308    190400022   -0.0200    -0.1243  16.0657
## 2016-12-30    16.2100   12262167    197653917    0.1400     0.8712  16.1190
## 2017-01-03    16.3000   16237125    265043268    0.0900     0.5552  16.3233
## 2017-01-04    16.3300   29658734    482612222    0.0300     0.1840  16.2722
## 2017-01-05    16.3000   26437646    431449126   -0.0300    -0.1837  16.3195
## 2017-01-06    16.1800   17195598    278864536   -0.1200    -0.7362  16.2172
## 2017-01-09    16.2000   14908745    241579598    0.0200     0.1236  16.2039
## 2017-01-10    16.1900    7996756    129458363   -0.0100    -0.0617  16.1889
## 2017-01-11    16.1600    9193332    148793816   -0.0300    -0.1853  16.1850
## 2017-01-12    16.1200    8296150    134057682   -0.0400    -0.2475  16.1590
## 2017-01-13    16.2700   19034143    308468975    0.1500     0.9305  16.2061
## 2017-01-16    16.5600   53304724    876414508    0.2900     1.7824  16.4416
## 2017-01-17    16.4000   12555292    206230538   -0.1600    -0.9662  16.4258
## 2017-01-18    16.4800   11478663    189191354    0.0800     0.4878  16.4820
## 2017-01-19    16.5400   12180687    201675871    0.0600     0.3641  16.5570
## 2017-01-20    16.6000   14288268    236958622    0.0600     0.3628  16.5841
## 2017-01-23    16.5700   14616540    242945923   -0.0300    -0.1807  16.6213
## 2017-01-24    16.6900   14985241    249503391    0.1200     0.7242  16.6499
## 2017-01-25    16.6900   11284869    188172530    0.0000     0.0000  16.6748
## 2017-01-26    16.7400    8602907    144343272    0.0500     0.2996  16.7784
## 2017-02-03    16.6300    8174289    136532939   -0.1100    -0.6571  16.7027
## 2017-02-06    16.6600   13455850    225037394    0.0300     0.1804  16.7241
## 2017-02-07    16.6700   14759284    246037892    0.0100     0.0600  16.6700
## 2017-02-08    16.6700   11238867    186815177    0.0000     0.0000  16.6222
## 2017-02-09    16.7200   11393034    190339364    0.0500     0.2999  16.7066
## 2017-02-10    16.7800   13985262    234489132    0.0600     0.3589  16.7669
## 2017-02-13    16.8500   19992872    336932200    0.0700     0.4172  16.8526
## 2017-02-14    16.7500   12987235    217924152   -0.1000    -0.5935  16.7799
```

```
## 2017-02-15    16.8400   25688032    433573962    0.0900    0.5373  16.8784
## 2017-02-16    16.7800   16327832    274241728   -0.0600   -0.3563  16.7960
## 2017-02-17    16.6400   13863642    231623672   -0.1400   -0.8343  16.7073
## 2017-02-20    16.9100   29949984    503858485    0.2700    1.6226  16.8233
## 2017-02-21    16.8800   17509118    296489045   -0.0300   -0.1774  16.9334
## 2017-02-22    16.7500   17032277    285870629   -0.1300   -0.7701  16.7841
## 2017-02-23    16.6900   15011148    250909254   -0.0600   -0.3582  16.7149
## 2017-02-24    16.7100   11594971    193610637    0.0200    0.1198  16.6978
## 2017-02-27    16.5900   13732273    228302043   -0.1200   -0.7181  16.6252
## 2017-02-28    16.5900   12097943    200661656    0.0000    0.0000  16.5864
## 2017-03-01    16.5600   16226984    268840218   -0.0300   -0.1808  16.5675
## 2017-03-02    16.4100   18996632    312626391   -0.1500   -0.9058  16.4569
## 2017-03-03    16.3500   12429467    203315662   -0.0600   -0.3656  16.3576
## 2017-03-06    16.4100   15703751    258165904    0.0600    0.3670  16.4398
## 2017-03-07    16.4300    9319842    152969168    0.0200    0.1219  16.4133
## 2017-03-08    16.3900   10109647    165715073   -0.0400   -0.2435  16.3918
## 2017-03-09    16.2200   17366025    282644706   -0.1700   -1.0372  16.2757
## 2017-03-10    16.2300   16396375    265663459    0.0100    0.0617  16.2026
## 2017-03-13    16.3400   17950147    291999358    0.1100    0.6778  16.2672
## 2017-03-14    16.2600   16988979    276642688   -0.0800   -0.4896  16.2837
## 2017-03-15    16.2400   18900324    306175990   -0.0200   -0.1230  16.1995
## 2017-03-16    16.2900   19036545    310417986    0.0500    0.3079  16.3064
## 2017-03-17    16.2000   21560354    350179470   -0.0900   -0.5525  16.2418
## 2017-03-20    16.1400   15017806    242775930   -0.0600   -0.3704  16.1659
## 2017-03-21    16.0000   31124617    499206643   -0.1400   -0.8674  16.0390
## 2017-03-22    15.7800   43811429    691677895   -0.2200   -1.3750  15.7876
## 2017-03-23    15.8800   22225660    352635979    0.1000    0.6337  15.8662
## 2017-03-24    15.9600   19902006    316984958    0.0800    0.5038  15.9273
## 2017-03-27    16.0400   18997369    304990151    0.0800    0.5013  16.0543
## 2017-03-28    16.0100   13212969    211750766   -0.0300   -0.1870  16.0260
## 2017-03-29    15.8700   23446792    372794465   -0.1400   -0.8745  15.8996
## 2017-03-30    15.8100   23645144    373665115   -0.0600   -0.3781  15.8030
## 2017-03-31    16.0100   24187120    385793421    0.2000    1.2650  15.9504
## 2017-04-05    16.1600   31036271    499121565    0.1500    0.9369  16.0819
## 2017-04-06    16.1000   22335442    359361840   -0.0600   -0.3713  16.0893
## 2017-04-07    16.1000   20126948    323821647    0.0000    0.0000  16.0890
## 2017-04-10    15.9700   16154024    258662668   -0.1300   -0.8075  16.0123
## 2017-04-11    15.8900   18611375    295381671   -0.0800   -0.5009  15.8710
## 2017-04-12    15.8900   22210541    352357729    0.0000    0.0000  15.8644
```

```
## 2017-04-13    15.7500    23426650    369388222  -0.1400   -0.8811  15.7679
## 2017-04-14    15.5100    40156553    625510632  -0.2400   -1.5238  15.5768
## 2017-04-17    15.5600    36739448    566536789   0.0500    0.3224  15.4204
## 2017-04-18    15.3000    22889980    351624622  -0.2600   -1.6710  15.3615
## 2017-04-19    15.1100    24455407    371021507  -0.1900   -1.2418  15.1713
## 2017-04-20    14.9100    41061406    614436383  -0.2000   -1.3236  14.9638
## 2017-04-21    15.0500    22688358    340453104   0.1400    0.9390  15.0056
## 2017-04-24    15.0000    17627803    264649933  -0.0500   -0.3322  15.0132
## 2017-04-25    15.0500    12975919    195296862   0.0500    0.3333  15.0507
## 2017-04-26    15.0500    14939871    225022668   0.0000    0.0000  15.0619
## 2017-04-27    15.2100    22887645    345791526   0.1600    1.0631  15.1082
## 2017-04-28    15.2100    15718509    238419161   0.0000    0.0000  15.1681
## 2017-05-02    15.1600    12607509    191225527  -0.0500   -0.3287  15.1676
## 2017-05-03    15.0800    14247943    215130847  -0.0800   -0.5277  15.0991
## 2017-05-04    14.9800    19477788    291839737  -0.1000   -0.6631  14.9832
## 2017-05-05    14.9200    40194577    592160198  -0.0600   -0.4005  14.7323
## 2017-05-08    14.8600    43568576    638781010  -0.0600   -0.4021  14.6615
## 2017-05-09    14.7600    19225492    283864640  -0.1000   -0.6729  14.7650
##             换手率(%) A股流通市值(元)    总市值(元) A股流通股本(股) 市盈率
## 2016-01-04    0.2264    332031791187 332031791187     18653471415 6.5614
## 2016-01-05    0.3112    335016346613 335016346613     18653471415 6.6204
## 2016-01-06    0.2507    337627832612 337627832612     18653471415 6.6720
## 2016-01-07    0.0608    326622284477 326622284477     18653471415 6.4545
## 2016-01-08    0.3855    326249215048 326249215048     18653471415 6.4471
## 2016-01-11    0.4834    318041687626 318041687626     18653471415 6.2849
## 2016-01-12    0.2969    325503076192 325503076192     18653471415 6.4324
## 2016-01-13    0.2566    324383867907 324383867907     18653471415 6.4102
## 2016-01-14    0.2940    334270207757 334270207757     18653471415 6.6056
## 2016-01-15    0.2505    327928027476 327928027476     18653471415 6.4803
## 2016-01-18    0.1755    329793374617 329793374617     18653471415 6.5171
## 2016-01-19    0.1598    331472187045 331472187045     18653471415 6.5503
## 2016-01-20    0.1928    321772381909 321772381909     18653471415 6.3586
## 2016-01-21    0.1833    309274556061 309274556061     18653471415 6.1117
## 2016-01-22    0.2252    317482083483 317482083483     18653471415 6.2739
## 2016-01-25    0.1263    317668618197 317668618197     18653471415 6.2775
## 2016-01-26    0.2052    310766833774 310766833774     18653471415 6.1412
## 2016-01-27    0.2374    311139903202 311139903202     18653471415 6.1485
## 2016-01-28    0.2086    309274556061 309274556061     18653471415 6.1117
## 2016-01-29    0.2543    315243666914 315243666914     18653471415 6.2296
```

```
## 2016-02-01    0.1562    307782278348 307782278348     18653471415 6.0822
## 2016-02-02    0.1361    312632180915 312632180915     18653471415 6.1780
## 2016-02-03    0.1489    309834160203 309834160203     18653471415 6.1227
## 2016-02-04    0.2528    322705055480 322705055480     18653471415 6.3771
## 2016-02-05    0.4204    342850804608 342850804608     18653471415 6.7752
## 2016-02-15    0.3608    344156547607 344156547607     18653471415 6.8010
## 2016-02-16        NA    344156547607 344156547607     18653471415 6.8010
## 2016-02-17        NA    344156547607 344156547607     18653471415 6.8010
## 2016-02-18        NA    344156547607 344156547607     18653471415 6.8010
## 2016-02-19        NA    344156547607 344156547607     18653471415 6.8010
## 2016-02-22        NA    344156547607 344156547607     18653471415 6.8010
## 2016-02-23        NA    344156547607 344156547607     18653471415 6.8010
## 2016-02-24        NA    344156547607 344156547607     18653471415 6.8010
## 2016-02-25        NA    344156547607 344156547607     18653471415 6.8010
## 2016-02-26        NA    344156547607 344156547607     18653471415 6.8010
## 2016-02-29        NA    344156547607 344156547607     18653471415 6.8010
## 2016-03-01        NA    344156547607 344156547607     18653471415 6.8010
## 2016-03-02        NA    344156547607 344156547607     18653471415 6.8010
## 2016-03-03        NA    344156547607 344156547607     18653471415 6.8010
## 2016-03-04        NA    344156547607 344156547607     18653471415 6.8010
## 2016-03-07        NA    344156547607 344156547607     18653471415 6.8010
## 2016-03-08        NA    344156547607 344156547607     18653471415 6.8010
## 2016-03-09        NA    344156547607 344156547607     18653471415 6.8010
## 2016-03-10        NA    344156547607 344156547607     18653471415 6.8010
## 2016-03-11    0.5267    316549409913 316549409913     18653471415 6.2554
## 2016-03-14    0.3052    314497528057 314497528057     18653471415 6.2149
## 2016-03-15    0.2013    317482083483 317482083483     18653471415 6.2739
## 2016-03-16    0.5811    336322089612 336322089612     18653471415 6.6462
## 2016-03-17    0.3680    330166444046 330166444046     18653471415 6.5245
## 2016-03-18    0.3001    332777930044 350609194366     18653471415 6.9285
## 2016-03-21    0.2252    336881693755 354932850351     18653471415 7.0139
## 2016-03-22    0.2346    338560506182 356701618708     18653471415 7.0489
## 2016-03-23    0.1492    338187436754 356308559073     18653471415 7.0411
## 2016-03-24    0.1471    330539513474 348250836557     18653471415 6.8819
## 2016-03-25    0.0730    330912582902 348643896192     18653471415 6.8897
## 2016-03-28    0.1315    325503076192 342944531485     18653471415 6.7770
## 2016-03-29    0.1027    325503076192 342944531485     18653471415 6.7770
## 2016-03-30    0.1906    333337534186 351198783819     18653471415 6.9401
## 2016-03-31    0.1274    334456742471 352377962724     18653471415 6.9634
```

```
## 2016-04-01    0.1919    339120110325 357291208160     18653471415 7.0605
## 2016-04-05    0.2285    338933575611 357094678343     18653471415 7.0566
## 2016-04-06    0.1256    335016346613 352967552176     18653471415 6.9751
## 2016-04-07    0.1557    329979909331 347661247104     18653471415 6.8702
## 2016-04-08    0.0918    328301096904 345892478747     18653471415 6.8353
## 2016-04-11    0.1293    329606839903 347268187469     18653471415 6.8625
## 2016-04-12    0.0756    328674166332 346285538382     18653471415 6.8430
## 2016-04-13    0.1886    330912582902 348643896192     18653471415 6.8897
## 2016-04-14    0.0847    332031791187 349823075097     18653471415 6.9130
## 2016-04-15    0.1864    333710603614 351591843454     18653471415 6.9479
## 2016-04-18    0.1506    332218325901 350019604914     18653471415 6.9168
## 2016-04-19    0.1461    333897138329 351788373271     18653471415 6.9518
## 2016-04-20    0.2499    335016346613 352967552176     18653471415 6.9751
## 2016-04-21    0.2585    338000902040 356112029256     18653471415 7.0372
## 2016-04-22    0.1511    340239318610 358470387065     18653471415 7.0838
## 2016-04-25    0.1302    333524068900 351395313636     18653471415 6.9440
## 2016-04-26    0.0844    334270207757 352181432906     18653471415 6.9596
## 2016-04-27    0.0742    333150999472 351002254001     18653471415 6.9363
## 2016-04-28    0.0843    333897138329 351788373271     18653471415 6.9518
## 2016-04-29    0.1237    332591395329 350412664549     18653471415 6.9246
## 2016-05-03    0.1489    338000902040 356112029256     18653471415 7.0372
## 2016-05-04    0.1239    337254763183 355325909986     18653471415 7.0217
## 2016-05-05    0.0801    336881693755 354932850351     18653471415 7.0139
## 2016-05-06    0.1255    330726048188 348447366374     18653471415 6.8858
## 2016-05-09    0.1177    326622284477 344123710390     18653471415 6.8003
## 2016-05-10    0.0903    326062680334 343534120938     18653471415 6.7887
## 2016-05-11    0.0640    325503076192 342944531485     18653471415 6.7770
## 2016-05-12    0.0786    324756937335 342158412215     18653471415 6.7615
## 2016-05-13    0.0590    321585847195 338817405318     18653471415 6.6955
## 2016-05-16    0.0592    322518520765 339800054406     18653471415 6.7149
## 2016-05-17    0.0524    321772381909 339013935136     18653471415 6.6994
## 2016-05-18    0.1660    327554958047 345106359477     18653471415 6.8197
## 2016-05-19    0.0557    323637729050 340979233310     18653471415 6.7382
## 2016-05-20    0.0597    325503076192 342944531485     18653471415 6.7770
## 2016-05-23    0.0755    327368423333 344909829660     18653471415 6.8159
## 2016-05-24    0.0575    327368423333 344909829660     18653471415 6.8159
## 2016-05-25    0.0738    329606839903 347268187469     18653471415 6.8625
## 2016-05-26    0.0593    330166444046 347857776922     18653471415 6.8741
## 2016-05-27    0.0623    330912582902 348643896192     18653471415 6.8897
```

```
## 2016-05-30    0.0940    334829811899 352771022359     18653471415 6.9712
## 2016-05-31    0.1910    341171992180 359453036153     18653471415 7.1033
## 2016-06-01    0.0982    339306645039 357487737978     18653471415 7.0644
## 2016-06-02    0.0954    336135554898 354146731081     18653471415 6.9984
## 2016-06-03    0.0965    335949020184 353950201263     18653471415 6.9945
## 2016-06-06    0.0957    336135554898 354146731081     18653471415 6.9984
## 2016-06-07    0.0886    336135554898 354146731081     18653471415 6.9984
## 2016-06-08    0.1274    335575950756 353557141629     18653471415 6.9867
## 2016-06-13    0.1611    331658721759 349430015462     18653471415 6.9052
## 2016-06-14    0.1006    331845256473 349626545279     18653471415 6.9091
## 2016-06-15    0.1412    330912582902 348643896192     18653471415 6.8897
## 2016-06-16    0.1655    332031791187 349823075097     18653471415 6.9130
## 2016-06-17    0.0933    331472187045 349233485644     18653471415 6.9013
## 2016-06-20    0.0519    332964464758 350805724184     18653471415 6.9324
## 2016-06-21    0.0850    331658721759 349430015462     18653471415 6.9052
## 2016-06-22    0.0763    333710603614 351591843454     18653471415 6.9479
## 2016-06-23    0.0628    322555827716 339839360374     20518818557 6.7157
## 2016-06-24    0.1490    313527547551 330327317208     20518818557 6.5277
## 2016-06-27    0.0934    314143112108 330975865606     20518818557 6.5405
## 2016-06-28    0.0612    313527547551 330327317208     20518818557 6.5277
## 2016-06-29    0.1036    318041687634 335083338791     20518818557 6.6217
## 2016-06-30    0.1011    319478004932 336596618386     20518818557 6.6516
## 2016-07-01    0.1224    326659591427 344163016358     20518818557 6.8011
## 2016-07-04    0.1044    326864779613 344379199157     20518818557 6.8054
## 2016-07-05    0.0788    323992145015 341352639968     20518818557 6.7456
## 2016-07-06    0.1120    320914322231 338109897980     20518818557 6.6815
## 2016-07-07    0.1194    319478004932 336596618386     20518818557 6.6516
## 2016-07-08    0.0726    317015746706 334002424795     20518818557 6.6003
## 2016-07-11    0.1114    316605370335 333570059196     20518818557 6.5918
## 2016-07-12    0.1245    320914322231 338109897980     20518818557 6.6815
## 2016-07-13    0.1766    323992145015 341352639968     20518818557 6.7456
## 2016-07-14    0.0823    322145451345 339406994775     20518818557 6.7071
## 2016-07-15    0.0780    322350639530 339623177575     20518818557 6.7114
## 2016-07-18    0.1353    323581768644 340920274370     20518818557 6.7370
## 2016-07-19    0.0837    322761015902 340055543173     20518818557 6.7199
## 2016-07-20    0.0725    323171392273 340487908772     20518818557 6.7285
## 2016-07-21    0.1073    324812897757 342217371165     20518818557 6.7627
## 2016-07-22    0.0814    320709134046 337893715181     20518818557 6.6772
## 2016-07-25    0.0558    321940263159 339190811976     20518818557 6.7028
```

```
## 2016-07-26    0.0997    322966204087 340271725972     20518818557 6.7242
## 2016-07-27    0.1983    325633650500 343082102362     20518818557 6.7797
## 2016-07-28    0.0851    321735074974 338974629177     20518818557 6.6986
## 2016-07-29    0.0699    322145451345 339406994775     20518818557 6.7071
## 2016-08-01    0.1117    324197333201 341568822768     20518818557 6.7498
## 2016-08-02    0.0530    323171392273 340487908772     20518818557 6.7285
## 2016-08-03    0.0516    321529886788 338758446378     20518818557 6.6943
## 2016-08-04    0.0761    319888381304 337028983984     20518818557 6.6601
## 2016-08-05    0.0860    322761015902 340055543173     20518818557 6.7199
## 2016-08-08    0.0640    322966204087 340271725972     20518818557 6.7242
## 2016-08-09    0.0754    323992145015 341352639968     20518818557 6.7456
## 2016-08-10    0.0902    324812897757 342217371165     20518818557 6.7627
## 2016-08-11    0.2568    329737414211 347405758347     20518818557 6.8652
## 2016-08-12    0.2290    335687871593 353675059524     20518818557 6.9891
## 2016-08-15    0.3769    347588786356 366213661879     20518818557 7.2369
## 2016-08-16    0.2407    342664269902 361025274697     20518818557 7.1343
## 2016-08-17    0.1776    341433140788 359728177902     20518818557 7.1087
## 2016-08-18    0.1809    337329377077 355404521918     20518818557 7.0232
## 2016-08-19    0.0971    336919000706 354972156319     20518818557 7.0147
## 2016-08-22    0.0963    336713812520 354755973520     20518818557 7.0104
## 2016-08-23    0.1548    336508624335 354539790721     20518818557 7.0062
## 2016-08-24    0.0833    336303436149 354323607922     20518818557 7.0019
## 2016-08-25    0.1206    335277495221 353242693925     20518818557 6.9805
## 2016-08-26    0.0716    335277495221 353242693925     20518818557 6.9805
## 2016-08-29    0.1472    333020425180 350864683134     20518818557 6.9335
## 2016-08-30    0.1774    336508624335 354539790721     20518818557 7.0062
## 2016-08-31    0.1072    338150129819 356269253115     20518818557 7.0403
## 2016-09-01    0.1037    336713812520 354755973520     20518818557 7.0104
## 2016-09-02    0.1128    338150129819 356269253115     20518818557 7.0403
## 2016-09-05    0.1251    338355318005 356485435914     20518818557 7.0446
## 2016-09-06    0.0988    336713812520 354755973520     20518818557 7.0104
## 2016-09-07    0.0876    338150129819 356269253115     20518818557 7.0403
## 2016-09-08    0.1189    340612388046 358863446705     20518818557 7.0916
## 2016-09-09    0.0913    339791635304 357998715508     20518818557 7.0745
## 2016-09-12    0.1779    336508624335 354539790721     20518818557 7.0062
## 2016-09-13    0.0694    337534565263 355620704717     20518818557 7.0275
## 2016-09-14    0.1169    336508624335 354539790721     20518818557 7.0062
## 2016-09-19    0.0705    337739753448 355836887516     20518818557 7.0318
## 2016-09-20    0.0848    337739753448 355836887516     20518818557 7.0318
```

```
## 2016-09-21    0.0459    337534565263 355620704717     20518818557 7.0275
## 2016-09-22    0.0562    339176070747 357350167111     20518818557 7.0617
## 2016-09-23    0.0497    338150129819 356269253115     20518818557 7.0403
## 2016-09-26    0.1012    338150129819 356269253115     20518818557 7.0403
## 2016-09-27    0.0830    338150129819 356269253115     20518818557 7.0403
## 2016-09-28    0.0626    338150129819 356269253115     20518818557 7.0403
## 2016-09-29    0.0536    338560506191 356701618713     20518818557 7.0489
## 2016-09-30    0.0594    338355318005 356485435914     20518818557 7.0446
## 2016-10-10    0.0910    339586447118 357782532709     20518818557 7.0702
## 2016-10-11    0.0618    339996823489 358214898308     20518818557 7.0788
## 2016-10-12    0.0519    338765694376 356917801512     20518818557 7.0532
## 2016-10-13    0.0707    337124188892 355188339118     20518818557 7.0190
## 2016-10-14    0.0520    337739753448 355836887516     20518818557 7.0318
## 2016-10-17    0.0800    333841177922 351729414331     20518818557 6.9506
## 2016-10-18    0.0890    335277495221 353242693925     20518818557 6.9805
## 2016-10-19    0.0543    333841177922 351729414331     20518818557 6.9506
## 2016-10-20    0.0438    334251554294 352161779929     20518818557 6.9592
## 2016-10-21    0.0530    334456742479 352377962729     20518818557 6.9634
## 2016-10-24    0.1207    337944941634 356053070315     20518818557 7.0361
## 2016-10-25    0.0649    336919000706 354972156319     20518818557 7.0147
## 2016-10-26    0.0606    334867118850 352810328327     20518818557 6.9720
## 2016-10-27    0.0951    332199672438 349999951937     20518818557 6.9164
## 2016-10-28    0.0734    334251554294 352161779929     20518818557 6.9592
## 2016-10-31    0.0715    333841177922 351729414331     20518818557 6.9506
## 2016-11-01    0.0635    334456742479 352377962729     20518818557 6.9634
## 2016-11-02    0.1221    332404860623 350216134736     20518818557 6.9207
## 2016-11-03    0.1261    335482683407 353458876725     20518818557 6.9848
## 2016-11-04    0.1049    336508624335 354539790721     20518818557 7.0062
## 2016-11-07    0.1041    336919000706 354972156319     20518818557 7.0147
## 2016-11-08    0.0843    337944941634 356053070315     20518818557 7.0361
## 2016-11-09    0.1941    337739753448 355836887516     20518818557 7.0318
## 2016-11-10    0.0998    339996823489 358214898308     20518818557 7.0788
## 2016-11-11    0.1050    340407199861 358647263906     20518818557 7.0873
## 2016-11-14    0.0883    340612388046 358863446705     20518818557 7.0916
## 2016-11-15    0.0766    340407199861 358647263906     20518818557 7.0873
## 2016-11-16    0.0604    340612388046 358863446705     20518818557 7.0916
## 2016-11-17    0.1107    340612388046 358863446705     20518818557 7.0916
## 2016-11-18    0.1427    340612388046 358863446705     20518818557 7.0916
## 2016-11-21    0.1045    341433140788 359728177902     20518818557 7.1087
```

```
## 2016-11-22    0.1027    342253893531 360592909099     20518818557 7.1258
## 2016-11-23    0.2195    347383598170 365997479079     20518818557 7.2326
## 2016-11-24    0.1026    346562845428 365132747883     20518818557 7.2155
## 2016-11-25    0.1137    350666609139 369456403867     20518818557 7.3009
## 2016-11-28    0.1649    353128867366 372050597458     20518818557 7.3522
## 2016-11-29    0.1439    354154808294 373131511454     20518818557 7.3736
## 2016-11-30    0.0981    352102926438 370969683462     20518818557 7.3308
## 2016-12-01    0.0899    351897738253 370753500662     20518818557 7.3266
## 2016-12-02    0.1652    350871797325 369672586666     20518818557 7.3052
## 2016-12-05    0.1862    354975561036 373996242651     20518818557 7.3906
## 2016-12-06    0.1021    353334055552 372266780257     20518818557 7.3565
## 2016-12-07    0.0758    350256232768 369024038269     20518818557 7.2924
## 2016-12-08    0.0645    351692550067 370537317863     20518818557 7.3223
## 2016-12-09    0.1405    355796313778 374860973847     20518818557 7.4077
## 2016-12-12    0.2357    358874136562 378103715836     20518818557 7.4718
## 2016-12-13    0.0760    352308114624 371185866261     20518818557 7.3351
## 2016-12-14    0.0870    352513302809 371402049060     20518818557 7.3394
## 2016-12-15    0.1502    342664269902 361025274697     20518818557 7.1343
## 2016-12-16    0.0708    341843517160 360160543501     20518818557 7.1172
## 2016-12-19    0.0413    338970882562 357133984311     20518818557 7.0574
## 2016-12-20    0.1098    334046366108 351945597130     20518818557 6.9549
## 2016-12-21    0.0616    334867118850 352810328327     20518818557 6.9720
## 2016-12-22    0.0560    332199672438 349999951937     20518818557 6.9164
## 2016-12-23    0.0545    332199672438 349999951937     20518818557 6.9164
## 2016-12-26    0.0706    333430801551 351297048733     20518818557 6.9421
## 2016-12-27    0.0520    331173731510 348919037941     20518818557 6.8951
## 2016-12-28    0.0704    330147790582 347838123945     20518818557 6.8737
## 2016-12-29    0.0578    329737414211 347405758347     20518818557 6.8652
## 2016-12-30    0.0598    332610048809 350432317536     20518818557 6.9250
## 2017-01-03    0.0791    334456742479 352377962729     20518818557 6.6362
## 2017-01-04    0.1445    335072307036 353026511126     20518818557 6.6485
## 2017-01-05    0.1288    334456742479 352377962729     20518818557 6.6362
## 2017-01-06    0.0838    331994484252 349783769138     20518818557 6.5874
## 2017-01-09    0.0727    332404860623 350216134736     20518818557 6.5955
## 2017-01-10    0.0390    332199672438 349999951937     20518818557 6.5915
## 2017-01-11    0.0448    331584107881 349351403540     20518818557 6.5792
## 2017-01-12    0.0404    330763355139 348486672343     20518818557 6.5630
## 2017-01-13    0.0928    333841177922 351729414331     20518818557 6.6240
## 2017-01-16    0.2598    339791635304 357998715508     20518818557 6.7421
```

```
## 2017-01-17    0.0612    336508624335 354539790721     20518818557 6.6770
## 2017-01-18    0.0559    338150129819 356269253115     20518818557 6.7095
## 2017-01-19    0.0594    339381258933 357566349910     20518818557 6.7340
## 2017-01-20    0.0696    340612388046 358863446705     20518818557 6.7584
## 2017-01-23    0.0712    339996823489 358214898308     20518818557 6.7462
## 2017-01-24    0.0730    342459081716 360809091898     20518818557 6.7950
## 2017-01-25    0.0550    342459081716 360809091898     20518818557 6.7950
## 2017-01-26    0.0419    343485022644 361890005894     20518818557 6.8154
## 2017-02-03    0.0398    341227952603 359511995103     20518818557 6.7706
## 2017-02-06    0.0656    341843517160 360160543501     20518818557 6.7828
## 2017-02-07    0.0719    342048705345 360376726300     20518818557 6.7869
## 2017-02-08    0.0548    342048705345 360376726300     20518818557 6.7869
## 2017-02-09    0.0555    343074646273 361457640296     20518818557 6.8072
## 2017-02-10    0.0682    344305775386 362754737091     20518818557 6.8317
## 2017-02-13    0.0974    345742092685 364268016686     20518818557 6.8602
## 2017-02-14    0.0633    343690210830 362106188694     20518818557 6.8195
## 2017-02-15    0.1252    345536904500 364051833886     20518818557 6.8561
## 2017-02-16    0.0796    344305775386 362754737091     20518818557 6.8317
## 2017-02-17    0.0676    341433140788 359728177902     20518818557 6.7747
## 2017-02-20    0.1460    346973221799 365565113481     20518818557 6.8846
## 2017-02-21    0.0853    346357657242 364916565083     20518818557 6.8724
## 2017-02-22    0.0830    343690210830 362106188694     20518818557 6.8195
## 2017-02-23    0.0732    342459081716 360809091898     20518818557 6.7950
## 2017-02-24    0.0565    342869458087 361241457497     20518818557 6.8032
## 2017-02-27    0.0669    340407199861 358647263906     20518818557 6.7543
## 2017-02-28    0.0590    340407199861 358647263906     20518818557 6.7543
## 2017-03-01    0.0791    339791635304 357998715508     20518818557 6.7421
## 2017-03-02    0.0926    336713812520 354755973520     20518818557 6.6810
## 2017-03-03    0.0606    335482683407 353458876725     20518818557 6.6566
## 2017-03-06    0.0765    336713812520 354755973520     20518818557 6.6810
## 2017-03-07    0.0454    337124188892 355188339118     20518818557 6.6892
## 2017-03-08    0.0493    336303436149 354323607922     20518818557 6.6729
## 2017-03-09    0.0846    332815236995 350648500335     20518818557 6.6037
## 2017-03-10    0.0799    333020425180 350864683134     20518818557 6.6077
## 2017-03-13    0.0875    335277495221 353242693925     20518818557 6.6525
## 2017-03-14    0.0828    333635989737 351513231532     20518818557 6.6200
## 2017-03-15    0.0921    333225613366 351080865933     20518818557 6.6118
## 2017-03-16    0.0928    334251554294 352161779929     20518818557 6.6322
## 2017-03-17    0.1051    332404860623 350216134736     20518818557 6.5955
```

```
## 2017-03-20    0.0695    348919037941 348919037941     21618279922 6.5711
## 2017-03-21    0.1440    345892478752 345892478752     21618279922 6.5141
## 2017-03-22    0.2027    341136457169 341136457169     21618279922 6.4245
## 2017-03-23    0.1028    343298285161 343298285161     21618279922 6.4652
## 2017-03-24    0.0921    345027747555 345027747555     21618279922 6.4978
## 2017-03-27    0.0879    346757209949 346757209949     21618279922 6.5304
## 2017-03-28    0.0611    346108661551 346108661551     21618279922 6.5182
## 2017-03-29    0.1085    343082102362 343082102362     21618279922 6.4612
## 2017-03-30    0.1094    341785005567 341785005567     21618279922 6.4368
## 2017-03-31    0.1119    346108661551 346108661551     21618279922 6.5182
## 2017-04-05    0.1436    349351403540 349351403540     21618279922 6.5792
## 2017-04-06    0.1033    348054306744 348054306744     21618279922 6.5548
## 2017-04-07    0.0931    348054306744 348054306744     21618279922 6.5548
## 2017-04-10    0.0747    345243930354 345243930354     21618279922 6.5019
## 2017-04-11    0.0861    343514467961 343514467961     21618279922 6.4693
## 2017-04-12    0.1027    343514467961 343514467961     21618279922 6.4693
## 2017-04-13    0.1084    340487908772 340487908772     21618279922 6.4123
## 2017-04-14    0.1858    335299521590 335299521590     21618279922 6.3146
## 2017-04-17    0.1699    336380435586 336380435586     21618279922 6.3350
## 2017-04-18    0.1059    330759682807 330759682807     21618279922 6.2291
## 2017-04-19    0.1131    326652209621 326652209621     21618279922 6.1518
## 2017-04-20    0.1899    322328553637 322328553637     21618279922 6.0703
## 2017-04-21    0.1049    325355112826 325355112826     21618279922 6.1273
## 2017-04-24    0.0815    324274198830 324274198830     21618279922 6.1070
## 2017-04-25    0.0600    325355112826 325355112826     21618279922 6.1273
## 2017-04-26    0.0691    325355112826 325355112826     21618279922 6.1273
## 2017-04-27    0.1059    328814037614 328814037614     21618279922 6.1925
## 2017-04-28    0.0727    328814037614 328814037614     21618279922 6.1925
## 2017-05-02    0.0583    327733123618 327733123618     21618279922 6.1721
## 2017-05-03    0.0659    326003661224 326003661224     21618279922 6.1395
## 2017-05-04    0.0901    323841833232 323841833232     21618279922 6.0988
## 2017-05-05    0.1859    322544736436 322544736436     21618279922 6.0744
## 2017-05-08    0.2015    321247639641 321247639641     21618279922 6.0500
## 2017-05-09    0.0889    319085811649 319085811649     21618279922 6.0093
```

**题目 70（数据计算）：对收盘价做步长为 5 的滑动平均**

**难度：⋆⋆⋆**

**代码及运行结果：**

```r
library(slider)

df %>%
  mutate(avg_5 = slide_dbl(`收盘价（元）`, mean, na.rm = TRUE,
                           .before = 2, .after = 2)) %>%
  select(日期, `收盘价（元）`, avg_5)
```

```
## # A tibble: 327 x 3
##    日期       `收盘价(元)` avg_5
##    <date>            <dbl> <dbl>
## 1 2016-01-04         15.7  15.9
## 2 2016-01-05         15.9  15.8
## 3 2016-01-06         16.0  15.7
## 4 2016-01-07         15.5  15.6
## 5 2016-01-08         15.4  15.5
## 6 2016-01-11         15.1  15.3
## # ... with 321 more rows
```

**题目 71（数据计算）：对收盘价做步长为 5 的滑动求和**

难度：★★★

代码及运行结果：

```r
df %>%
  mutate(sum_5 = slide_dbl(`收盘价（元）`, sum, na.rm = TRUE,
                           .before = 2, .after = 2)) %>%
  select(日期, `收盘价（元）`, sum_5)
```

```
## # A tibble: 327 x 3
##    日期       `收盘价(元)` sum_5
##    <date>            <dbl> <dbl>
## 1 2016-01-04         15.7  47.6
## 2 2016-01-05         15.9  63.0
## 3 2016-01-06         16.0  78.5
## 4 2016-01-07         15.5  77.8
## 5 2016-01-08         15.4  77.4
## 6 2016-01-11         15.1  76.7
## # ... with 321 more rows
```

**题目 72（数据可视化）：将收盘价及其 5 日均线、20 日均线绘制在同一个图上**

难度：★★★★

代码及运行结果：

```
df %>%
  mutate(avg_5 = slide_dbl(`收盘价 (元)`, mean, na.rm = TRUE,
                           .before = 2, .after = 2),
  avg_20 = slide_dbl(`收盘价 (元)`, mean, na.rm = TRUE,
                     .before = 10, .after = 9)) %>%
  pivot_longer(c(`收盘价 (元)`, avg_5, avg_20),
               names_to = "type", values_to = "price") %>%
  ggplot(aes(日期, price, color = type)) +
    geom_line()
```


**题目 73**（数据重采样）：按周为采样规则，计算一周收盘价最大值

难度：★★★★

代码及运行结果：

```
library(tsibble)
weekmax = df %>%
  group_by(weeks = yearweek(日期)) %>%    # 年-周
  slice_max(`收盘价 (元)`)   # 默认 n = 1
```

```
weekmax
```

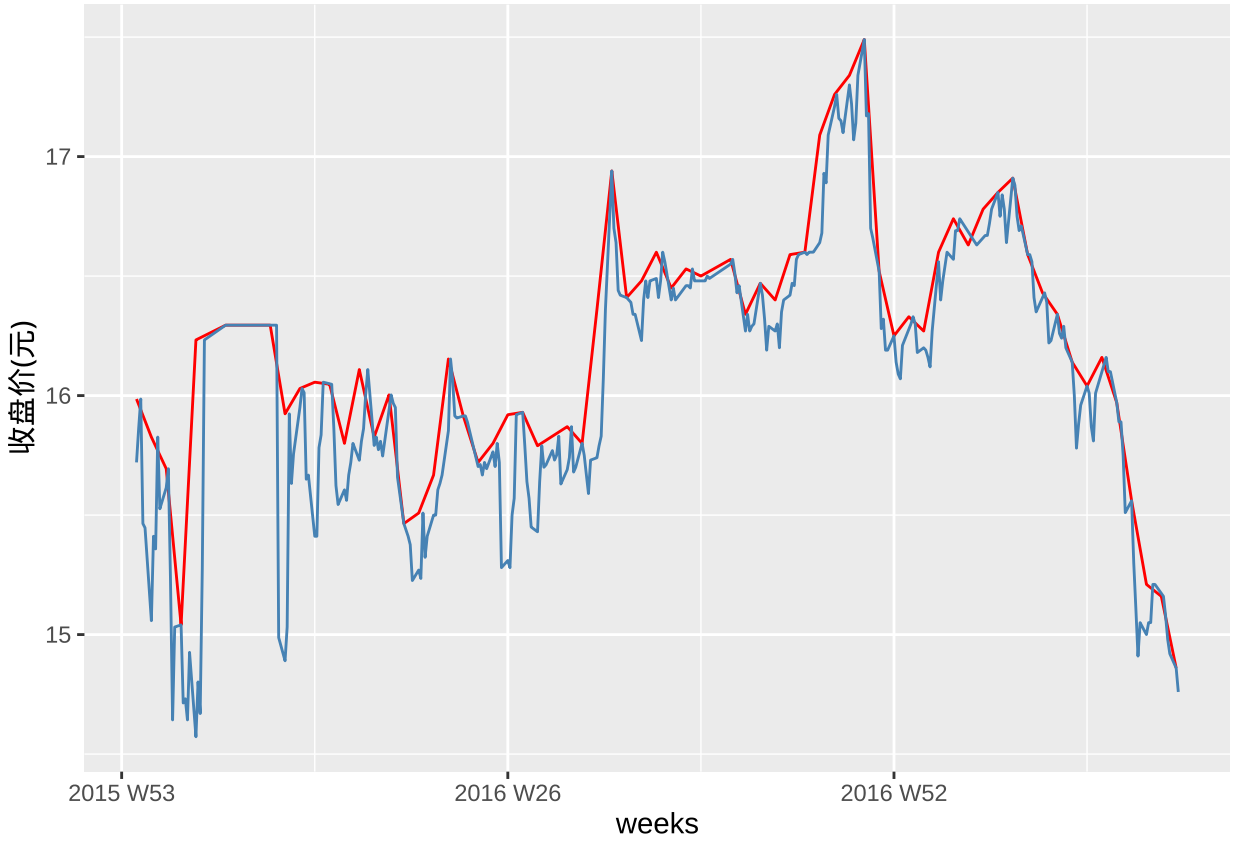
```
## # A tibble: 91 x 19
## # Groups:   weeks [69]
##   代码  简称  日期       前收~1 开盘价~2 最高~3 最低价~4 收盘~5 成交量~6  成交~7
##   <chr> <chr> <date>      <dbl>    <dbl>  <dbl>    <dbl>  <dbl>    <dbl>   <dbl>
## 1 6000~ 浦发~ 2016-01-06   15.9     15.8   16.0     15.6   16.0 46772653  8.39e8
## 2 6000~ 浦发~ 2016-01-14   15.4     15.0   15.9     14.9   15.8 54838833  9.66e8
## 3 6000~ 浦发~ 2016-01-19   15.6     15.7   15.8     15.5   15.7 29807159  5.28e8
## 4 6000~ 浦发~ 2016-01-25   15.0     15.0   15.2     14.8   15.0 23558971  4.00e8
## 5 6000~ 浦发~ 2016-02-05   15.3     15.2   16.2     15.1   16.2 78413454  1.40e9
## 6 6000~ 浦发~ 2016-02-15   16.2     15.9   16.3     15.7   16.3 67296317  1.22e9
## # ... with 85 more rows, 9 more variables: `涨跌(元)` <dbl>, `涨跌幅(%)` <dbl>,
## #   `均价(元)` <dbl>, `换手率(%)` <dbl>, `A股流通市值(元)` <dbl>,
## #   `总市值(元)` <dbl>, `A股流通股本(股)` <dbl>, 市盈率 <dbl>, weeks <week>,
## #   and abbreviated variable names 1: `前收盘价(元)`, 2: `开盘价(元)`,
## #   3: `最高价(元)`, 4: `最低价(元)`, 5: `收盘价(元)`, 6: `成交量(股)`,
## #   7: `成交金额(元)`
```

**题目 74（数据可视化）：绘制重采样数据与原始数据**

**难度：** ⋆⋆⋆

**代码及运行结果：**

```
weekmax %>%
  ggplot(aes(weeks, `收盘价 (元)`)) +
  geom_line(color = "red") +
  geom_line(data = df, aes(日期, `收盘价 (元)`), color = "steelblue")
```




**题目 75（数据操作）：将数据往后移动 5 天**

**难度：** ⋆⋆⋆

**代码及运行结果：**

```
df %>%
  mutate(across(4:18, ~ lag(.x, 5)))
```

```
## # A tibble: 327 x 18
##   代码  简称  日期       前收~1 开盘价~2 最高~3 最低价~4 收盘~5 成交量~6  成交~7
##   <chr> <chr> <date>      <dbl>    <dbl>  <dbl>    <dbl>  <dbl>    <dbl>   <dbl>
## 1 6000~ 浦发~ 2016-01-04   NA       NA     NA       NA     NA          NA NA
## 2 6000~ 浦发~ 2016-01-05   NA       NA     NA       NA     NA          NA NA
## 3 6000~ 浦发~ 2016-01-06   NA       NA     NA       NA     NA          NA NA
## 4 6000~ 浦发~ 2016-01-07   NA       NA     NA       NA     NA          NA NA
## 5 6000~ 浦发~ 2016-01-08   NA       NA     NA       NA     NA          NA NA
## 6 6000~ 浦发~ 2016-01-11   16.1     16.1   16.1     15.5   15.7 42240610  7.54e8
## # ... with 321 more rows, 8 more variables: `涨跌(元)` <dbl>,
## #   `涨跌幅(%)` <dbl>, `均价(元)` <dbl>, `换手率(%)` <dbl>,
## #   `A股流通市值(元)` <dbl>, `总市值(元)` <dbl>, `A股流通股本(股)` <dbl>,
```

```
## #   市盈率 <dbl>, and abbreviated variable names 1: `前收盘价(元)`,
## #   2: `开盘价(元)`, 3: `最高价(元)`, 4: `最低价(元)`, 5: `收盘价(元)`,
## #   6: `成交量(股)`, 7: `成交金额(元)`
```

注：这是批量做后移，单个变量做后移用 mutate(var = lag(var, 5)) 即可。

**题目 76（数据操作）：将数据往前移动 5 天**

**难度：★★★**

**代码及运行结果：**

```
df %>%
  mutate(across(4:18, ~ lead(.x, 5)))
```

```
## # A tibble: 327 x 18
##   代码  简称  日期       前收~1 开盘价~2 最高~3 最低价~4 收盘~5 成交量~6  成交~7
##   <chr> <chr> <date>      <dbl>    <dbl>  <dbl>    <dbl>  <dbl>    <dbl>   <dbl>
## 1 6000~ 浦发~ 2016-01-04   15.4     15.2   15.4     15.0   15.1 90177135  1.55e9
## 2 6000~ 浦发~ 2016-01-05   15.1     15.2   15.5     15.1   15.4 55374454  9.64e8
## 3 6000~ 浦发~ 2016-01-06   15.4     15.5   15.8     15.3   15.4 47869312  8.44e8
## 4 6000~ 浦发~ 2016-01-07   15.4     15.0   15.9     14.9   15.8 54838833  9.66e8
## 5 6000~ 浦发~ 2016-01-08   15.8     15.7   16.0     15.5   15.5 46723139  8.36e8
## 6 6000~ 浦发~ 2016-01-11   15.5     15.4   15.9     15.3   15.6 32729006  5.83e8
## # ... with 321 more rows, 8 more variables: `涨跌(元)` <dbl>,
## #   `涨跌幅(%)` <dbl>, `均价(元)` <dbl>, `换手率(%)` <dbl>,
## #   `A股流通市值(元)` <dbl>, `总市值(元)` <dbl>, `A股流通股本(股)` <dbl>,
## #   市盈率 <dbl>, and abbreviated variable names 1: `前收盘价(元)`,
## #   2: `开盘价(元)`, 3: `最高价(元)`, 4: `最低价(元)`, 5: `收盘价(元)`,
## #   6: `成交量(股)`, 7: `成交金额(元)`
```

**题目 77（数据操作）：计算开盘价的累积平均**

**难度：★★★**

**代码及运行结果：**

```
rlt = df %>%
  mutate(累积平均 = cummean(`开盘价 (元)`)) %>%
  select(日期, `开盘价 (元)`, 累积平均)
rlt
```

```
## # A tibble: 327 x 3
##   日期       `开盘价(元)` 累积平均
##   <date>            <dbl>    <dbl>
```

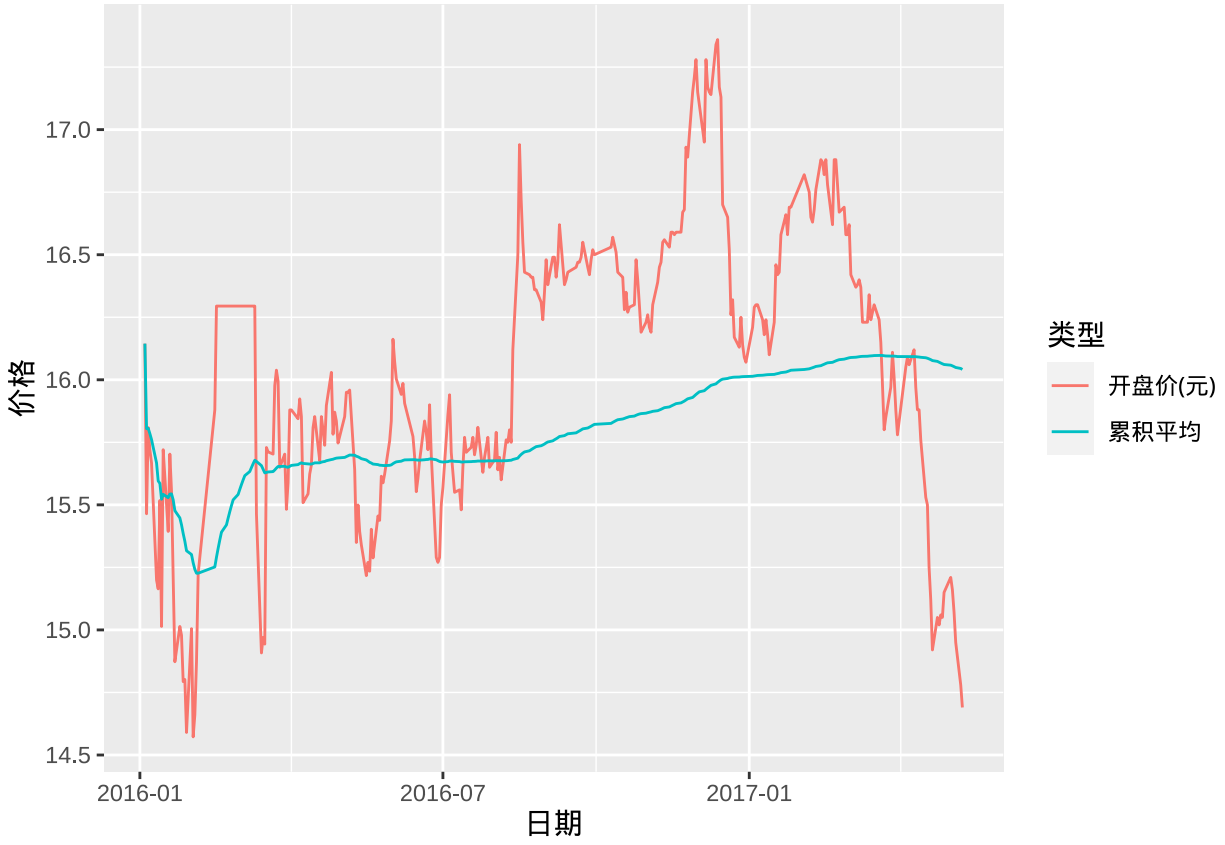
```
## 1 2016-01-04          16.1     16.1
## 2 2016-01-05          15.5     15.8
## 3 2016-01-06          15.8     15.8
## 4 2016-01-07          15.7     15.8
## 5 2016-01-08          15.7     15.8
## 6 2016-01-11          15.2     15.7
## # ... with 321 more rows
```

题目 **78**（数据计算）：绘制开盘价的累积平均与原始数据的折线图

难度：⋆⋆⋆

代码及运行结果：

```
rlt %>%
  pivot_longer(-日期, names_to = " 类型", values_to = " 价格") %>%
  ggplot(aes(日期, 价格, color = 类型)) +
    geom_line()
```




题目 **79**（数据计算）：计算布林指标

难度：⋆⋆⋆⋆

代码及运行结果：

```
boll = df %>%
  mutate(avg_20 = slide_dbl(`收盘价 (元)`, mean, na.rm = TRUE,
                            .before = 10, .after = 9),
         sd_20 = slide_dbl(`收盘价 (元)`, sd, na.rm = TRUE,
                           .before = 10, .after = 9),
         up = avg_20 + 2 * sd_20,
         down = avg_20 - 2 * sd_20) %>%
  select(日期, `收盘价 (元)`, avg_20, up, down)

boll %>%
  slice_sample(n = 10)
```

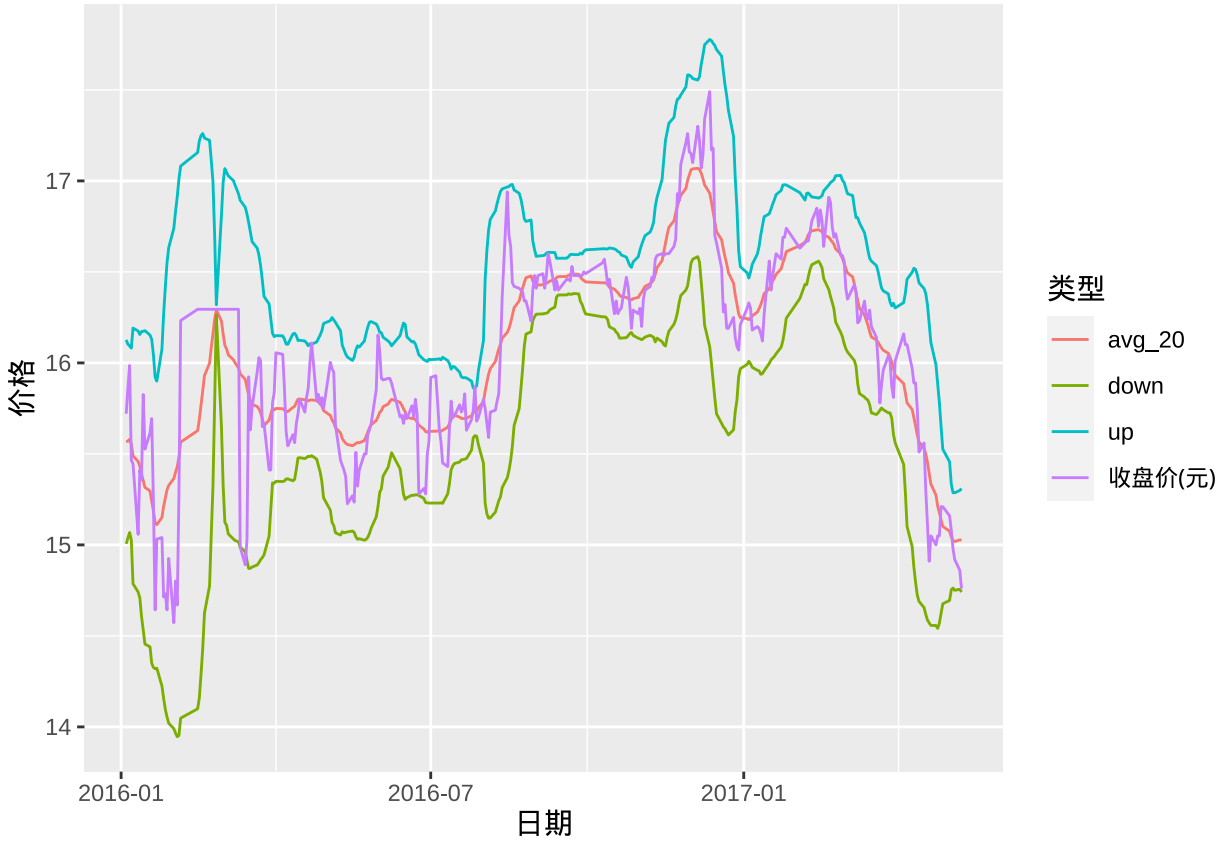
```
## # A tibble: 10 x 5
##    日期       `收盘价(元)` avg_20    up  down
##    <date>            <dbl>  <dbl> <dbl> <dbl>
## 1 2016-07-29         15.7   15.8  16.0  15.6
## 2 2016-04-22         16.1   15.8  16.1  15.5
## 3 2016-09-13         16.4   16.5  16.6  16.4
## 4 2016-12-29         16.1   16.3  16.6  15.9
## 5 2016-08-29         16.2   16.5  16.8  16.2
## 6 2016-07-13         15.8   15.7  16.0  15.4
## # ... with 4 more rows
```

**题目 80（数据可视化）：绘制布林曲线**

难度：★★★

代码及运行结果：

```
boll %>%
  pivot_longer(-日期, names_to = " 类型", values_to = " 价格") %>%
  ggplot(aes(日期, 价格, color = 类型)) +
    geom_line()
```




## Part VI 数据生成

**题目 81（加载查看包）：加载并查看 tidyverse 包版本**

难度：⋆

代码及运行结果：

```
# 得是首次加载
library(tidyverse)
```

**题目 82（生成随机数）：生成 20 个 0~100 的随机数，创建数据框**

难度：⋆

代码及运行结果：

```
set.seed(123)       # 保证结果出现
df1 = tibble(nums = sample(100, 20))
df1
```

```
## # A tibble: 20 x 1
##    nums
```

```
##   <int>
## 1    31
## 2    79
## 3    51
## 4    14
## 5    67
## 6    42
## # ... with 14 more rows
```

**题目 83（生成等差数）：生成 20 个 0~100 固定步长的数，创建数据框**

难度：⋆

代码及运行结果：

```
df2 = tibble(nums = seq(0, 99, by = 5))
df2
```

```
## # A tibble: 20 x 1
##    nums
##   <dbl>
## 1     0
## 2     5
## 3    10
## 4    15
## 5    20
## 6    25
## # ... with 14 more rows
```

**题目 84（生成指定分布随机数）：生成 20 个标准正态分布的随机数，创建数据框**

难度：⋆

代码及运行结果：

```
set.seed(123)
df3 = tibble(nums = rnorm(20, 0, 1))
df3
```

```
## # A tibble: 20 x 1
##      nums
##     <dbl>
## 1 -0.560
## 2 -0.230
## 3  1.56
```

```
## 4  0.0705
## 5  0.129
## 6  1.72
## # ... with 14 more rows
```

题目 **85**（合并数据）：将 **df1, df2, df3** 按行合并为新数据框

难度：⋆

代码及运行结果：

```
bind_rows(df1, df2, df3)
```

```
## # A tibble: 60 x 1
##    nums
##   <dbl>
## 1    31
## 2    79
## 3    51
## 4    14
## 5    67
## 6    42
## # ... with 54 more rows
```

题目 **86**（合并数据）：将 **df1, df2, df3** 按列合并为新数据框

难度：⋆

代码及运行结果：

```
df = bind_cols(df1, df2, df3)
df
```

```
## # A tibble: 20 x 3
##   nums...1 nums...2 nums...3
##      <int>    <dbl>    <dbl>
## 1       31        0  -0.560
## 2       79        5  -0.230
## 3       51       10   1.56
## 4       14       15   0.0705
## 5       67       20   0.129
## 6       42       25   1.72
## # ... with 14 more rows
```

题目 **87**（查看数据）：查看 **df** 所有数据的最小值、**25%** 分位数、中位数、**75%** 分位数、最大值

难度：⋆⋆

代码及运行结果：

```
unlist(df) %>%
  summary()
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -1.9666  0.4886 25.5000 33.3305 61.2500 97.0000
```

题目 **88**（修改列名）：修改列名为 **col1, col2, col3**

难度：⋆

代码及运行结果：

```
df = df %>%
  set_names(str_c("col", 1:3))
df
```

```
## # A tibble: 20 x 3
##    col1  col2    col3
##   <int> <dbl>   <dbl>
## 1    31     0 -0.560
## 2    79     5 -0.230
## 3    51    10  1.56
## 4    14    15  0.0705
## 5    67    20  0.129
## 6    42    25  1.72
## # ... with 14 more rows
```

注：若只修改个别列名，用 `rename(newname = oldname)`.

题目 **89**（数据操作）：提取在第 **1** 列中而不在第 **2** 列中的数

难度：⋆⋆

代码及运行结果：

```
setdiff(df$col1, df$col2)
```

```
##  [1] 31 79 51 14 67 42 43 97 69 57  9 72 26  7 87 36
```

题目 **90**（数据操作）：提取在第 **1** 列和第 **2** 列出现频率最高的三个数字

难度：⋆⋆⋆

代码及运行结果：

```
tibble(nums = c(df$col1, df$col2)) %>%
  count(nums, sort = TRUE) %>%
  slice(1:3)
```

```
## # A tibble: 3 x 2
##    nums     n
##   <dbl> <int>
## 1    25     2
## 2    50     2
## 3    90     2
```

**题目 91（数据操作）：提取第 1 列可以整除 5 的数的位置**

难度：⋆⋆

代码及运行结果：

```
which(df$col1 %% 5 == 0)
```

```
## [1]  7 10 11 18
```

- 选取满足条件的索引，通常用途还是用来选出满足条件的行，不兜圈子做法：

```
df %>%
  filter(col1 %% 5 == 0)
```

```
## # A tibble: 4 x 3
##    col1  col2   col3
##   <int> <dbl>  <dbl>
## 1    50    30  0.461
## 2    25    45 -0.446
## 3    90    50  1.22
## 4    95    85 -1.97
```

**题目 92（数据计算）：计算第 1 列的 1 阶差分**

难度：⋆⋆

代码及运行结果：

```
df %>%
  mutate(diff1 = col1 - lag(col1))
```

```
## # A tibble: 20 x 4
##    col1  col2    col3 diff1
```

```
##   <int> <dbl>   <dbl> <int>
## 1    31     0 -0.560     NA
## 2    79     5 -0.230     48
## 3    51    10  1.56     -28
## 4    14    15  0.0705   -37
## 5    67    20  0.129     53
## 6    42    25  1.72     -25
## # ... with 14 more rows
```

注：若只是要数值，用 `diff(df$col1)` 即可。

**题目 93（数据操作）：将 col1, col2, col3 三列顺序颠倒**

难度：⋆⋆

代码及运行结果：

```
df %>%
  select(rev(names(df)))
```

```
## # A tibble: 20 x 3
##      col3  col2  col1
##     <dbl> <dbl> <int>
## 1 -0.560      0    31
## 2 -0.230      5    79
## 3  1.56      10    51
## 4  0.0705    15    14
## 5  0.129     20    67
## 6  1.72      25    42
## # ... with 14 more rows
```

注：更灵活的调整列序，dplyr 1.0 提供的 `relocate()` 函数。

**题目 94（数据操作）：提取第一列位置在 1,10,15 的数**

难度：⋆

代码及运行结果：

```
df[c(1,10,15), 1]
```

```
## # A tibble: 3 x 1
##    col1
##   <int>
## 1    31
## 2    25
```

```
## 3    72
```

题目 **95**（数据操作）：查找第一列的局部最大值位置

难度：⋆⋆⋆

代码及运行结果：

```
df %>%
  mutate(diff = sign(col1 - lag(col1)) + sign(col1 - lead(col1)))
```

```
## # A tibble: 20 x 4
##    col1  col2    col3  diff
##   <int> <dbl>   <dbl> <dbl>
## 1    31     0 -0.560     NA
## 2    79     5 -0.230      2
## 3    51    10  1.56       0
## 4    14    15  0.0705    -2
## 5    67    20  0.129      2
## 6    42    25  1.72      -2
## # ... with 14 more rows
```

```
which(rlt$diff == 2)
```

```
## integer(0)
```

- 不兜圈子做法：

```
df %>%
  mutate(diff = sign(col1 - lag(col1)) + sign(col1 - lead(col1))) %>%
  filter(diff == 2)
```

```
## # A tibble: 7 x 4
##    col1  col2   col3  diff
##   <int> <dbl>  <dbl> <dbl>
## 1    79     5 -0.230     2
## 2    67    20  0.129     2
## 3    50    30  0.461     2
## 4    97    40 -0.687     2
## 5    90    50  1.22      2
## 6    72    70 -0.556     2
## # ... with 1 more row
```

题目 **96**（数据计算）：按行计算 **df** 每一行的均值

难度：⋆⋆

代码及运行结果：

```
rowMeans(df)    # 或者 apply(df, 1, mean)
```

```
##  [1] 10.146508 27.923274 20.852903  9.690169 29.043096 22.905022 26.820305
##  [8] 25.578313 45.437716 23.184779 47.074694 41.453271 39.133590 24.703561
## [15] 47.148053 34.262304 29.165950 59.344461 59.233785 43.509070
```

```
# 或者
df %>%
  mutate(row_avg = pmap_dbl(., ~ mean(c(...))))
```

```
## # A tibble: 20 x 4
##    col1  col2    col3 row_avg
##   <int> <dbl>   <dbl>   <dbl>
## 1    31     0 -0.560    10.1
## 2    79     5 -0.230    27.9
## 3    51    10  1.56     20.9
## 4    14    15  0.0705    9.69
## 5    67    20  0.129    29.0
## 6    42    25  1.72     22.9
## # ... with 14 more rows
```

题目 **97**（数据计算）：对第二列计算步长为 **3** 的移动平均值

* 难度：**★★★

代码及运行结果：

```
df %>%
  mutate(avg_3 = slide_dbl(col2, mean, .before = 1, .after = 1))
```

```
## # A tibble: 20 x 4
##    col1  col2    col3 avg_3
##   <int> <dbl>   <dbl> <dbl>
## 1    31     0 -0.560    2.5
## 2    79     5 -0.230    5
## 3    51    10  1.56    10
## 4    14    15  0.0705  15
## 5    67    20  0.129   20
## 6    42    25  1.72    25
## # ... with 14 more rows
```

题目 98（数据计算）：按第三列值的大小升序排列

难度：★★

代码及运行结果：

```
df %>%
  arrange(col3)
```

```
## # A tibble: 20 x 3
##    col1  col2   col3
##   <int> <dbl>  <dbl>
## 1    95    85 -1.97
## 2    43    35 -1.27
## 3    97    40 -0.687
## 4    31     0 -0.560
## 5    72    70 -0.556
## 6    36    95 -0.473
## # ... with 14 more rows
```

题目 99（数据操作）：按第一列大于 50 的数修改为”高”

难度：★★

代码及运行结果：

```
df %>%
  mutate(col1 = ifelse(col1 > 50, " 高", col1))
```

```
## # A tibble: 20 x 3
##   col1   col2    col3
##   <chr> <dbl>   <dbl>
## 1 31        0 -0.560
## 2 高        5 -0.230
## 3 高       10  1.56
## 4 14       15  0.0705
## 5 高       20  0.129
## 6 42       25  1.72
## # ... with 14 more rows
```

题目 100（数据计算）：计算第一列与第二列的欧氏距离

难度：★★★

代码及运行结果：

```
(df$col1 - df$col2) ^ 2 |> sum() |> sqrt()
```

```
## [1] 176.054
```

## Part V 高级

**题目 101（数据读取）：从 csv 文件中读取指定数据：读取前 10 行，positionName 和 salary 列**

难度：★★

代码及运行结果：

```
read_csv("data/数据 1_101-120 涉及.csv", n_max = 10,
         col_select = c(positionName, salary))
```

```
## # A tibble: 10 x 2
##   positionName salary
##   <chr>         <dbl>
## 1 数据分析      37500
## 2 数据建模      15000
## 3 数据分析       3500
## 4 数据分析      45000
## 5 数据分析      30000
## 6 数据分析      50000
## # ... with 4 more rows
```

**题目 102（数据读取）：从 csv 文件中读取数据，将薪资大于 10000 的改为” 高”**

难度：★★

代码及运行结果：

```
df = read_csv("data/数据 2_101-120 涉及.csv") %>%
  mutate(薪资水平 = if_else(薪资水平 > 10000, " 高", " 低"))
```

**题目 103（数据操作）：从 df 中对薪资水平每隔 20 行进行抽样**

难度：★★

代码及运行结果：

```
df %>%
  slice(seq(1, n(), by = 20))
```

```
## # A tibble: 58 x 2
##   学历要求 薪资水平
```

```
##   <chr>    <chr>
## 1 本科     高
## 2 本科     高
## 3 本科     高
## 4 本科     高
## 5 本科     高
## 6 本科     高
## # ... with 52 more rows
```

**题目 104（数据操作）：取消使用科学记数法**

难度：★★

代码及运行结果：

```
set.seed(123)
df = tibble(val = runif(10) ^ 10)
# 三位小数
df %>%
  mutate(val = scales::number(val, accuracy = 0.001))
```

```
## # A tibble: 10 x 1
##   val
##   <chr>
## 1 0.000
## 2 0.093
## 3 0.000
## 4 0.288
## 5 0.541
## 6 0.000
## # ... with 4 more rows
```

```
# 科学记数法
df %>%
  mutate(val =  scales::scientific(val, 2))
```

```
## # A tibble: 10 x 1
##   val
##   <chr>
## 1 3.9e-06
## 2 9.3e-02
## 3 1.3e-04
## 4 2.9e-01
```

```
## 5 5.4e-01
## 6 3.9e-14
## # ... with 4 more rows
```

题目 **105**（数据操作）：将上一题的数据转换为百分数

难度：⋆⋆⋆

代码及运行结果：

```
df %>%
  mutate(val = scales::percent(val, 0.01))
```

```
## # A tibble: 10 x 1
##   val
##   <chr>
## 1 0.00%
## 2 9.27%
## 3 0.01%
## 4 28.82%
## 5 54.13%
## 6 0.00%
## # ... with 4 more rows
```

题目 **106**（数据操作）：查找上一题数据中第 **3** 大值的行号

难度：⋆⋆⋆

代码及运行结果：

```
order(df$val, decreasing = TRUE)[3]
```

```
## [1] 4
```

- 不兜圈子做法：

```
df %>%
  arrange(-val) %>%
  slice(3)
```

```
## # A tibble: 1 x 1
##     val
##   <dbl>
## 1 0.288
```

题目 **107**（数据操作）：反转 **df** 的行

难度：⋆⋆

代码及运行结果：

```
df %>%
  slice(rev(1:n()))    # 或者 df[nrow(df):1,]
```

```
## # A tibble: 10 x 1
##         val
##       <dbl>
## 1 3.94e- 4
## 2 2.60e- 3
## 3 3.20e- 1
## 4 1.69e- 3
## 5 3.85e-14
## 6 5.41e- 1
## # ... with 4 more rows
```

题目 **108**（数据连接：全连接）：根据多列匹配合并数据，保留 **df1** 和 **df2** 的观测

难度：⋆⋆

代码及运行结果：

```
df1 = tibble(
  key1 = c("K0","K0","K1","K2"),
  key2 = c("K0","K1","K0","K1"),
  A = str_c('A', 0:3),
  B = str_c('B', 0:3))
df1
```

```
## # A tibble: 4 x 4
##   key1  key2  A     B
##   <chr> <chr> <chr> <chr>
## 1 K0    K0    A0    B0
## 2 K0    K1    A1    B1
## 3 K1    K0    A2    B2
## 4 K2    K1    A3    B3
```

```
df2 = tibble(
  key1 = c("K0","K1","K1","K2"),
  key2 = str_c("K", rep(0,4)),
  C = str_c('C', 0:3),
```

```
  D = str_c('D', 0:3))
df2
```

```
## # A tibble: 4 x 4
##   key1  key2  C     D
##   <chr> <chr> <chr> <chr>
## 1 K0    K0    C0    D0
## 2 K1    K0    C1    D1
## 3 K1    K0    C2    D2
## 4 K2    K0    C3    D3
```

```
df1 %>%
  full_join(df2, by = c("key1", "key2"))
```

```
## # A tibble: 6 x 6
##   key1  key2  A     B     C     D
##   <chr> <chr> <chr> <chr> <chr> <chr>
## 1 K0    K0    A0    B0    C0    D0
## 2 K0    K1    A1    B1    <NA>  <NA>
## 3 K1    K0    A2    B2    C1    D1
## 4 K1    K0    A2    B2    C2    D2
## 5 K2    K1    A3    B3    <NA>  <NA>
## 6 K2    K0    <NA>  <NA>  C3    D3
```

**题目 109（数据连接：左连接）：根据多列匹配合并数据，只保留 df1 的观测**

难度：⋆⋆

代码及运行结果：

```
df1 %>%
  left_join(df2, by = c("key1", "key2"))
```

```
## # A tibble: 5 x 6
##   key1  key2  A     B     C     D
##   <chr> <chr> <chr> <chr> <chr> <chr>
## 1 K0    K0    A0    B0    C0    D0
## 2 K0    K1    A1    B1    <NA>  <NA>
## 3 K1    K0    A2    B2    C1    D1
## 4 K1    K0    A2    B2    C2    D2
## 5 K2    K1    A3    B3    <NA>  <NA>
```

注：dplyr 包还提供了右连接：`right_join()`，内连接：`inner_join()`，以及用于过滤的连接：半连接：`semi_join()`，反连接：`anti_join()`.

**题目 110（数据处理）：再次读取数据 1 并显示所有列**

**难度：⋆⋆**

**代码及运行结果：**

```
df = read_csv("data/数据 1_101-120 涉及.csv")
glimpse(df)
```

```
## Rows: 105
## Columns: 53
## $ positionId           <dbl> 6802721, 5204912, 6877668, 6496141, 6467417, 688~
## $ positionName         <chr> "数据分析", "数据建模", "数据分析", "数据分析", ~
## $ companyId            <dbl> 475770, 50735, 100125, 26564, 29211, 94826, 3487~
## $ companyLogo          <chr> "i/image2/M01/B7/3E/CgoB5lwPfEaAdn8WAABWQ0Jgl5s3~
## $ companySize          <chr> "50-150人", "150-500人", "2000人以上", "500-2000~
## $ industryField        <chr> "移动互联网,电商", "电商", "移动互联网,企业服务"~
## $ financeStage         <chr> "A轮", "B轮", "上市公司", "D轮及以上", "上市公司~
## $ companyLabelList     <chr> "['绩效奖金', '带薪年假', '定期体检', '弹性工作'~
## $ firstType            <chr> "产品|需求|项目类", "开发|测试|运维类", "产品|需~
## $ secondType           <chr> "数据分析", "数据开发", "数据分析", "数据开发", ~
## $ thirdType            <chr> "数据分析", "建模", "数据分析", "数据分析", "数~
## $ skillLables          <chr> "['SQL', '数据库', '数据运营', 'BI']", "['算法',~
## $ positionLables       <chr> "['电商', '社交', 'SQL', '数据库', '数据运营', '~
## $ industryLables       <chr> "['电商', '社交', 'SQL', '数据库', '数据运营', '~
## $ createTime           <chr> "2020/3/16 11:00", "2020/3/16 11:08", "2020/3/16~
## $ formatCreateTime     <chr> "11:00发布", "11:08发布", "10:33发布", "10:10发~
## $ district             <chr> "余杭区", "滨江区", "江干区", "江干区", "余杭区"~
## $ businessZones        <chr> "['仓前']", "['西兴', '长河']", "['四季青', '钱~
## $ salary               <dbl> 37500, 15000, 3500, 45000, 30000, 50000, 30000, ~
## $ workYear             <chr> "1-3年", "3-5年", "1-3年", "3-5年", "3-5年", "1-~
## $ jobNature            <chr> "全职", "全职", "全职", "全职", "全职", "全职", ~
## $ education            <chr> "本科", "本科", "本科", "本科", "大专", "本科", ~
## $ positionAdvantage    <chr> "五险一金、弹性工作、带薪年假、年度体检", "六险~
## $ imState              <chr> "today", "disabled", "today", "threeDays", "disa~
## $ lastLogin            <chr> "2020/3/16 11:00", "2020/3/16 11:08", "2020/3/16~
## $ publisherId          <dbl> 12022406, 5491688, 5322583, 9814560, 6392394, 11~
## $ approve              <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ subwayline           <chr> NA, NA, "4号线", "1号线", NA, NA, NA, "2号线", N~
## $ stationname          <chr> NA, NA, "江锦路", "文泽路", NA, NA, NA, "丰潭路"~
## $ linestaion           <chr> NA, NA, "4号线_城星路;4号线_市民中心;4号线_江锦~
## $ latitude             <dbl> 30.27842, 30.18804, 30.24152, 30.29940, 30.28295~
```

```
## $ longitude             <dbl> 120.0059, 120.2012, 120.2125, 120.3503, 120.0098~
## $ hitags                <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ resumeProcessRate     <dbl> 50, 23, 11, 100, 20, 16, 100, 1, 83, 1, 83, 0, 1~
## $ resumeProcessDay      <dbl> 1, 1, 4, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, ~
## $ score                 <dbl> 233, 176, 80, 68, 66, 66, 65, 47, 24, 18, 17, 17~
## $ newScore              <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ matchScore            <dbl> 15.1018750, 32.5594140, 14.9723570, 12.8741530, ~
## $ matchScoreExplain     <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ query                 <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ explain               <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ isSchoolJob           <dbl> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, ~
## $ adWord                <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ plus                  <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ pcShow                <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ appShow               <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ deliver               <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ gradeDescription      <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ promotionScoreExplain <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ isHotHire             <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ count                 <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ aggregatePositionIds  <chr> "[]", "[]", "[]", "[]", "[]", "[]", "[]", "[]", ~
## $ famousCompany         <lgl> FALSE, FALSE, FALSE, TRUE, TRUE, FALSE, FALSE, T~
```

**题目 111（数据操作）：查找 secondType 与 thirdType 值相等的行号**

难度：★★

代码及运行结果：

```
which(df$secondType == df$thirdType)
```

```
##  [1]   1   3   5   6   7  11  15  24  26  28  29  30  31  34  38  39  40  41  42
## [20]  49  50  53  54  56  58  62  66  67  68  72  74  75  76  80  81  83  86  89
## [39]  90  92  97 101
```

- 不兜圈子：

```
df %>%
  filter(secondType == thirdType)
```

```
## # A tibble: 42 x 53
##   positionId positionN~1 compa~2 compa~3 compa~4 indus~5 finan~6 compa~7 first~8
##        <dbl> <chr>         <dbl> <chr>   <chr>   <chr>   <chr>   <chr>   <chr>
## 1    6802721 数据分析     475770 i/imag~ 50-150~ 移动互~ A轮     ['绩效~ 产品|~
```

```
## 2    6877668 数据分析     100125 image2~ 2000人~ 移动互~ 上市公~ ['节日~ 产品|~
## 3    6467417 数据分析      29211 i/imag~ 2000人~ 物流丨~ 上市公~ ['技能~ 产品|~
## 4    6882347 数据分析      94826 image2~ 50-150~ 移动互~ B轮      ['股票~ 产品|~
## 5    6841659 数据分析     348784 i/imag~ 50-150~ 移动互~ A轮      ['大牛~ 产品|~
## 6    6804629 数据分析师    34132 i/imag~ 150-50~ 数据服~ A轮      ['开放~ 产品|~
## # ... with 36 more rows, 44 more variables: secondType <chr>, thirdType <chr>,
## #   skillLables <chr>, positionLables <chr>, industryLables <chr>,
## #   createTime <chr>, formatCreateTime <chr>, district <chr>,
## #   businessZones <chr>, salary <dbl>, workYear <chr>, jobNature <chr>,
## #   education <chr>, positionAdvantage <chr>, imState <chr>, lastLogin <chr>,
## #   publisherId <dbl>, approve <dbl>, subwayline <chr>, stationname <chr>,
## #   linestaion <chr>, latitude <dbl>, longitude <dbl>, hitags <chr>, ...
```

**题目 112**（数据操作）：查找薪资大于平均薪资的第三个数据

难度：★★★

代码及运行结果：

```
df %>%
  filter(salary > mean(salary)) %>%
  slice(3)
```

```
## # A tibble: 1 x 53
##   positionId positionN~1 compa~2 compa~3 compa~4 indus~5 finan~6 compa~7 first~8
##        <dbl> <chr>         <dbl> <chr>   <chr>   <chr>   <chr>   <chr>   <chr>
## 1    6882347 数据分析      94826 image2~ 50-150~ 移动互~ B轮      ['股票~ 产品|~
## # ... with 44 more variables: secondType <chr>, thirdType <chr>,
## #   skillLables <chr>, positionLables <chr>, industryLables <chr>,
## #   createTime <chr>, formatCreateTime <chr>, district <chr>,
## #   businessZones <chr>, salary <dbl>, workYear <chr>, jobNature <chr>,
## #   education <chr>, positionAdvantage <chr>, imState <chr>, lastLogin <chr>,
## #   publisherId <dbl>, approve <dbl>, subwayline <chr>, stationname <chr>,
## #   linestaion <chr>, latitude <dbl>, longitude <dbl>, hitags <chr>, ...
```

**题目 113**（数据操作）：将上一题数据的 **salary** 列开根号

难度：★★

代码及运行结果：

```
df %>%
  mutate(salary_sqrt = sqrt(salary)) %>%
  select(salary, salary_sqrt)
```

```
## # A tibble: 105 x 2
##   salary salary_sqrt
##    <dbl>       <dbl>
## 1  37500       194.
## 2  15000       122.
## 3   3500        59.2
## 4  45000       212.
## 5  30000       173.
## 6  50000       224.
## # ... with 99 more rows
```

**题目 114（数据操作）：将上一题数据的 linestation 列按 __ 拆分**

难度：⋆⋆⋆

代码及运行结果：

```
df %>%
  separate(linestaion, into = c("line", "station"), sep = "_", remove = FALSE) %>%
  select(linestaion, line, station)
```

```
## # A tibble: 105 x 3
##   linestaion                                  line  station
##   <chr>                                       <chr> <chr>
## 1 <NA>                                        <NA>  <NA>
## 2 <NA>                                        <NA>  <NA>
## 3 4号线_城星路;4号线_市民中心;4号线_江锦路 4号线 城星路;4号线
## 4 1号线_文泽路                                1号线 文泽路
## 5 <NA>                                        <NA>  <NA>
## 6 <NA>                                        <NA>  <NA>
## # ... with 99 more rows
```

注：正常需要先按 “;” 分割，再分别按 “-” 分割。

**题目 115（数据查看）：查看上一题数据一共有多少列**

难度：⋆

代码及运行结果：

```
ncol(df)
```

```
## [1] 53
```

题目 116（数据操作）：提取 industryField 列以” 数据” 开头的行

难度：⋆⋆

代码及运行结果：

```
df %>%
  filter(str_detect(industryField, "^ 数据"))
```

```
## # A tibble: 15 x 53
##   positionId positionN~1 compa~2 compa~3 compa~4 indus~5 finan~6 compa~7 first~8
##        <dbl> <chr>         <dbl> <chr>   <chr>   <chr>   <chr>   <chr>   <chr>
## 1    6458372 数据分析专~   34132 i/imag~ 150-50~ 数据服~ A轮     ['开放~ 产品|~
## 2    6804629 数据分析师    34132 i/imag~ 150-50~ 数据服~ A轮     ['开放~ 产品|~
## 3    6804489 资深数据分~   34132 i/imag~ 150-50~ 数据服~ A轮     ['开放~ 开发|~
## 4    6267370 数据分析专~   31544 image1~ 150-50~ 数据服~ 不需要~ ['专业~ 开发|~
## 5    6804489 资深数据分~   34132 i/imag~ 150-50~ 数据服~ A轮     ['开放~ 开发|~
## 6    6242470 数据分析师    31544 image1~ 150-50~ 数据服~ 不需要~ ['专业~ 产品|~
## # ... with 9 more rows, 44 more variables: secondType <chr>, thirdType <chr>,
## #   skillLables <chr>, positionLables <chr>, industryLables <chr>,
## #   createTime <chr>, formatCreateTime <chr>, district <chr>,
## #   businessZones <chr>, salary <dbl>, workYear <chr>, jobNature <chr>,
## #   education <chr>, positionAdvantage <chr>, imState <chr>, lastLogin <chr>,
## #   publisherId <dbl>, approve <dbl>, subwayline <chr>, stationname <chr>,
## #   linestaion <chr>, latitude <dbl>, longitude <dbl>, hitags <chr>, ...
```

题目 117（数据分组汇总）：以 salary score 和 positionID 做数据透视表

难度：⋆⋆⋆

代码及运行结果：

```
df %>%
  group_by(positionId) %>%
  summarise(salary_avg = mean(salary), score_avg = mean(score))
```

```
## # A tibble: 95 x 3
##   positionId salary_avg score_avg
##        <dbl>      <dbl>     <dbl>
## 1    5203054      30000         4
## 2    5204912      15000       176
## 3    5269002      37500         1
## 4    5453691      30000         4
## 5    5519962      37500        14
```

```
## 6    5520623      30000         6
## # ... with 89 more rows
```

**题目 118（数据分组汇总）：同时对 salary、score 两列进行汇总计算**

难度：★★★

代码及运行结果：

```
df %>%
  summarise(across(c(salary, score),
                   list(sum=sum, mean=mean, min=min),
                   .names = "{.col}_{.fn}"))
```

```
## # A tibble: 1 x 6
##   salary_sum salary_mean salary_min score_sum score_mean score_min
##        <dbl>       <dbl>      <dbl>     <dbl>      <dbl>     <dbl>
## 1    3331000      31724.       3500      1335       12.7         0
```

注：若要分组再这样汇总，前面加上 group_by(var) 即可。

**题目 119（数据分组汇总）：同时对不同列进行不同的汇总计算：对 salary 求平均，对 score 求和**

难度：★★★

代码及运行结果：

```
df %>%
  summarise(salary_avg = mean(salary),
            score_sum = sum(score))
```

```
## # A tibble: 1 x 2
##   salary_avg score_sum
##        <dbl>     <dbl>
## 1     31724.      1335
```

注：若要分组再这样汇总，前面加上 group_by(var) 即可。

**题目 120（数据分组汇总）：计算并提取平均薪资最高的区**

难度：★★★★

代码及运行结果：

```
df %>%
  group_by(district) %>%
  summarise(salary_avg = mean(salary)) %>%
  slice_max(salary_avg)   # 默认 n = 1
```

```
## # A tibble: 1 x 2
##   district salary_avg
##   <chr>         <dbl>
## 1 萧山区        36250
```