Data manipulation/visualization con R

Vincenzo Nardelli - vincenzo.nardelli01@icatt.it

Luiss Business School 21/06/2019

Dopo aver visto le funzioni di subset in una dimensione (vettori) e due dimensioni (data.frame) con le funzioni base di R, utilizziamo alcuni pacchetti esterni per semplificare il lavoro in caso di aggregazioni più complesse.

```
install.packages("dplyr")
install.packages("ggplot2")
```

Per il momento attiviamo il pacchetto dplyr, utile per aggregare i dataframe e continuare l'analisi esplorativa.

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.5.2
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

Le funzioni principali di dplyr sono:

- group_by()
- select()
- filter()
- summarize()
- mutate()
- arrange()

Tramite il comando %>% chiamato pipe è possibile concatenare più comandi. Ad esempio il codice:

```
data %>%
group_by(country) %>%
  summarise(n = n())
```

```
## # A tibble: 38 x 2
##    country             n
##    <fct>           <int>
##  1 Australia        1259
##  2 Austria           401
##  3 Bahrain            19
##  4 Belgium          2069
##  5 Brazil             32
##  6 Canada            151
##  7 Channel Islands   758
##  8 Cyprus            622
##  9 Czech Republic     30
## 10 Denmark           389
## # ... with 28 more rows
```

equivale a

```
summarise(group_by(data, country), n = n())
```

```
## # A tibble: 38 x 2
##    country             n
##    <fct>           <int>
##  1 Australia        1259
##  2 Austria           401
##  3 Bahrain            19
##  4 Belgium          2069
##  5 Brazil             32
##  6 Canada            151
##  7 Channel Islands   758
##  8 Cyprus            622
##  9 Czech Republic     30
## 10 Denmark           389
## # ... with 28 more rows
```

Alcuni esempi di manipolazione di dati:

```
group_by(data, country) %>%
  summarise(n = n()) %>%
  arrange(n)
```

```
## # A tibble: 38 x 2
##    country                 n
##    <fct>               <int>
##  1 Saudi Arabia           10
##  2 Bahrain                19
##  3 Czech Republic         30
##  4 Brazil                 32
##  5 Lithuania              35
##  6 Lebanon                45
##  7 RSA                    58
##  8 European Community     61
##  9 United Arab Emirates   68
## 10 Malta                 127
## # ... with 28 more rows
```

```
data %>%
  filter(status == 'shipped') %>%
  group_by(country) %>%
  summarise(n = n()) %>%
  arrange(desc(n))
```

```
## # A tibble: 38 x 2
##    country             n
##    <fct>           <int>
##  1 United Kingdom 487622
##  2 Germany          9042
##  3 France           8408
##  4 EIRE             7894
##  5 Spain            2485
##  6 Netherlands      2363
##  7 Belgium          2031
##  8 Switzerland      1967
```

```
##  9 Portugal         1501
## 10 Australia        1185
## # ... with 28 more rows
```

```
data %>%
  group_by(country, status) %>%
  summarise(n = n())
```

```
## # A tibble: 68 x 3
## # Groups:   country [38]
##    country   status        n
##    <fct>     <fct>     <int>
##  1 Australia cancelled    74
##  2 Australia shipped    1185
##  3 Austria   cancelled     3
##  4 Austria   shipped     398
##  5 Bahrain   cancelled     1
##  6 Bahrain   shipped      18
##  7 Belgium   cancelled    38
##  8 Belgium   shipped    2031
##  9 Brazil    shipped      32
## 10 Canada    shipped     151
## # ... with 58 more rows
```

```
data <- data %>%
  mutate(price = unitprice*quantity)
```

```
data <- data %>%
  mutate(status_dummy = ifelse(status == "shipped", 1, 0))
```

```
data %>%
  group_by(country) %>%
  summarize(shipped = sum(status_dummy),
            total = n())
```

```
## # A tibble: 38 x 3
##    country         shipped total
##    <fct>             <dbl> <int>
##  1 Australia          1185  1259
##  2 Austria             398   401
##  3 Bahrain              18    19
##  4 Belgium            2031  2069
##  5 Brazil               32    32
##  6 Canada              151   151
##  7 Channel Islands     748   758
##  8 Cyprus              614   622
##  9 Czech Republic       25    30
## 10 Denmark             380   389
## # ... with 28 more rows
```

```
data %>%
  filter(status == "shipped") %>%
  group_by(day, hour) %>%
  summarize(mean = mean(price), sd = sd(price))
```

```
## # A tibble: 398 x 4
## # Groups:   day [31]
```

```
##      day  hour  mean    sd
##    <int> <int> <dbl> <dbl>
##  1     1     7  32.7  17.9
##  2     1     8  21.7  19.4
##  3     1     9  32.7  59.9
##  4     1    10  28.8  51.7
##  5     1    11  16.0  32.5
##  6     1    12  16.2  46.9
##  7     1    13  20.7  43.9
##  8     1    14  16.7  37.9
##  9     1    15  19.4  38.2
## 10     1    16  19.2  57.9
## # ... with 388 more rows
```

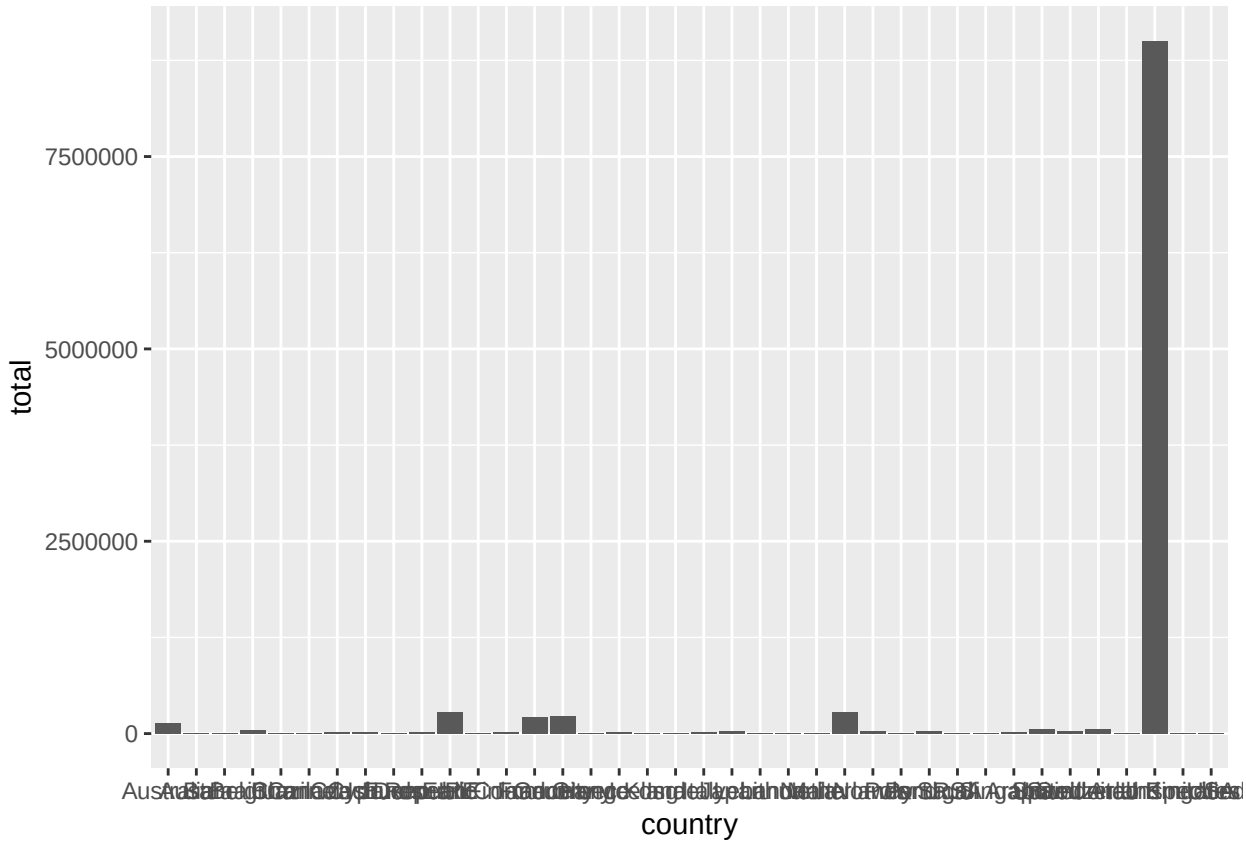
```
data %>%
  group_by(stockid) %>%
  summarize(shipped = sum(status_dummy),
            total = n()) %>%
  mutate(ratio = shipped/total) %>%
  arrange(ratio)
```

```
## # A tibble: 4,070 x 4
##    stockid shipped total ratio
##    <fct>     <dbl> <int> <dbl>
##  1 20957         0     1     0
##  2 35832         0     1     0
##  3 37503         0     1     0
##  4 79320         0     1     0
##  5 84839         0     1     0
##  6 85023C        0     1     0
##  7 85042         0     2     0
##  8 85063         0     2     0
##  9 85065         0     1     0
## 10 CRUK          0    16     0
## # ... with 4,060 more rows
```

Invece con il pacchetto ggplot2 è possibile creare visualizzazioni in modo molto semplice.

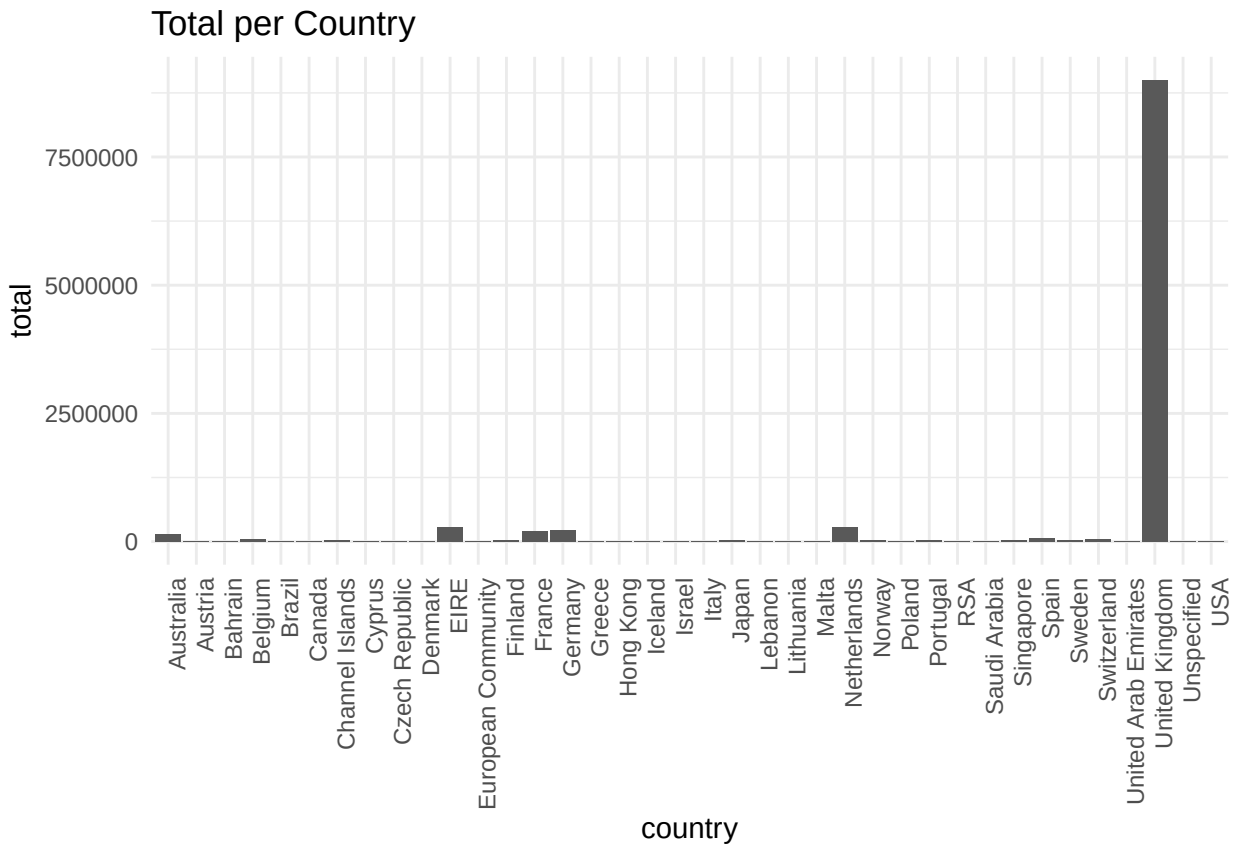
```
library(ggplot2)
```

```
total_sales <- data %>%
  group_by(country) %>%
  filter(status == "shipped") %>%
  summarize(total = sum(price))

ggplot(data=total_sales) +
  geom_col(aes(x=country, y=total))
```



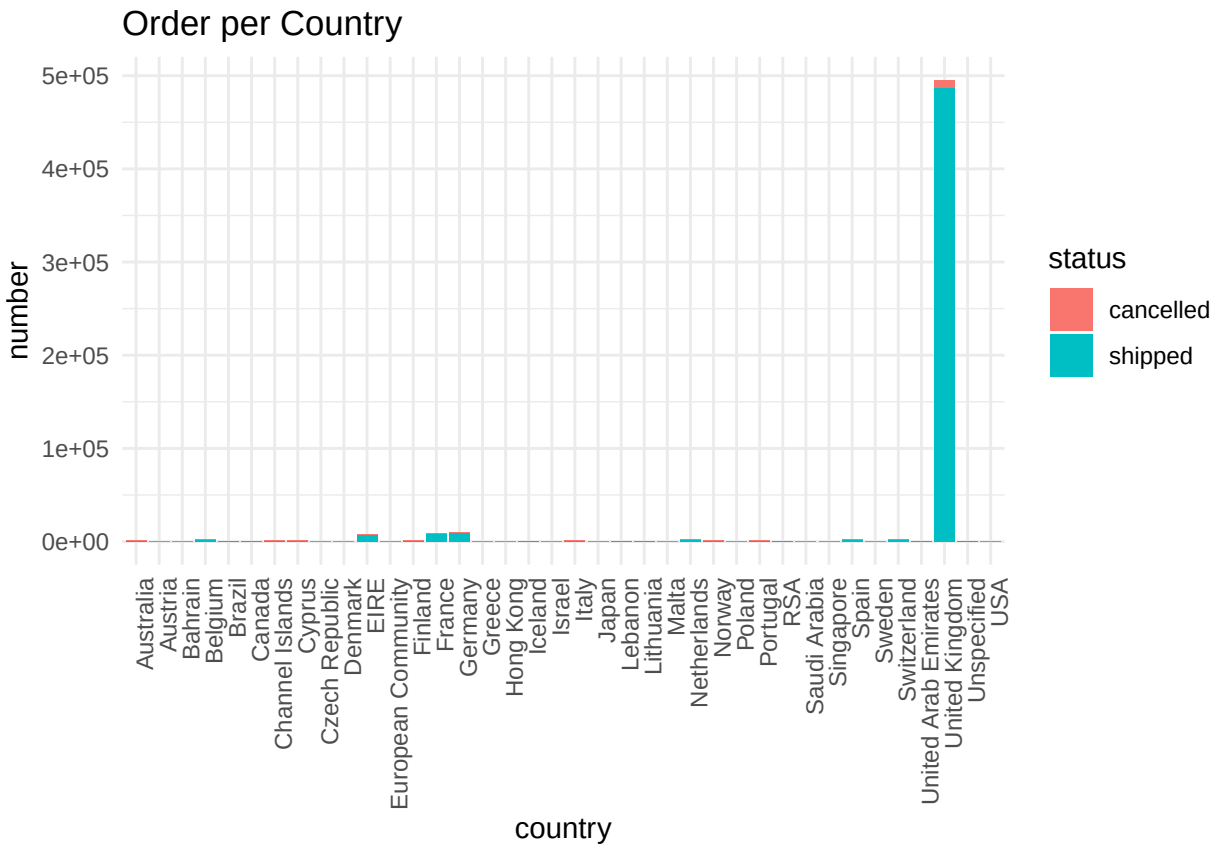
Cambiamo il tema, aggiungiamo il titolo e ruotiamo le scritte dell'asse x per una maggiore leggibilità.

```
ggplot(data=total_sales) +
  geom_col(aes(x=country, y=total)) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  ggtitle("Total per Country")
```

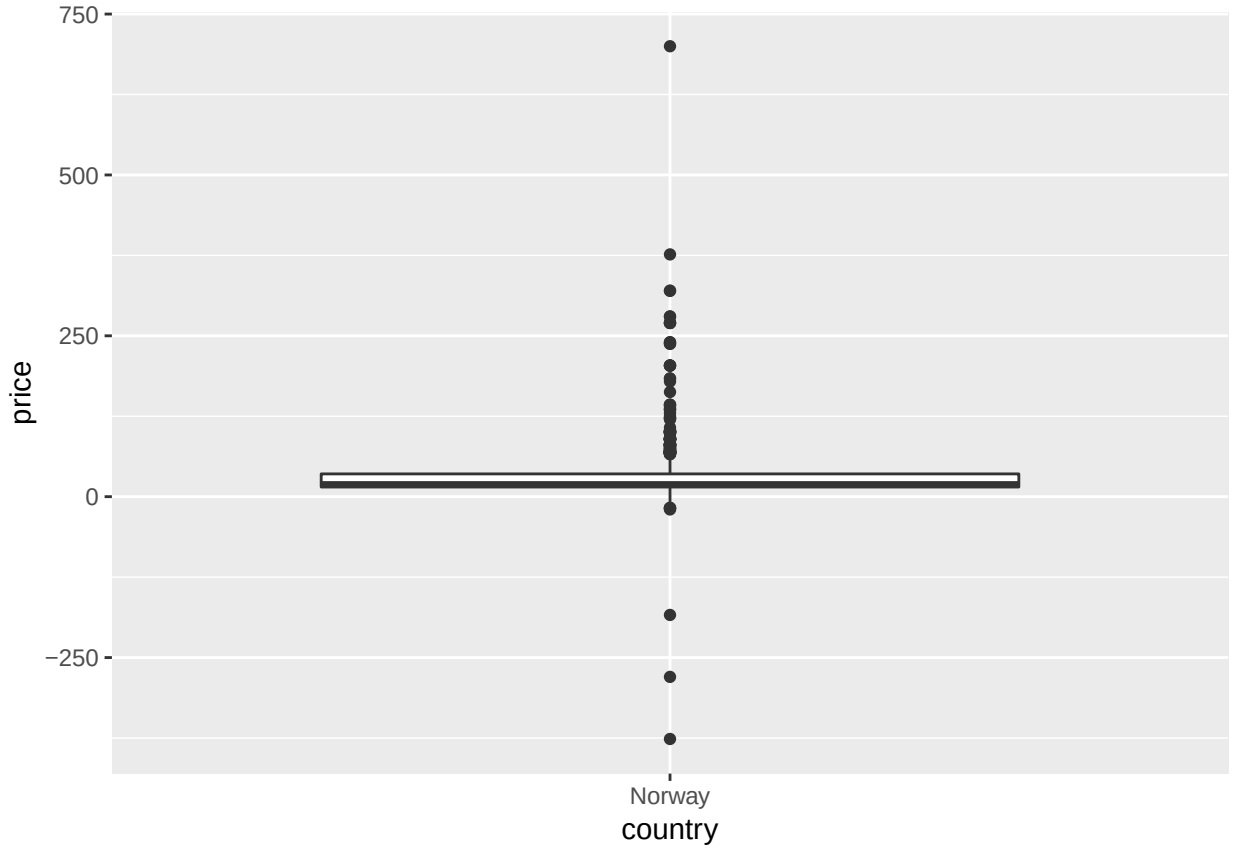


È possibile assegnare diversi colori ad una variabile categorica (factor).

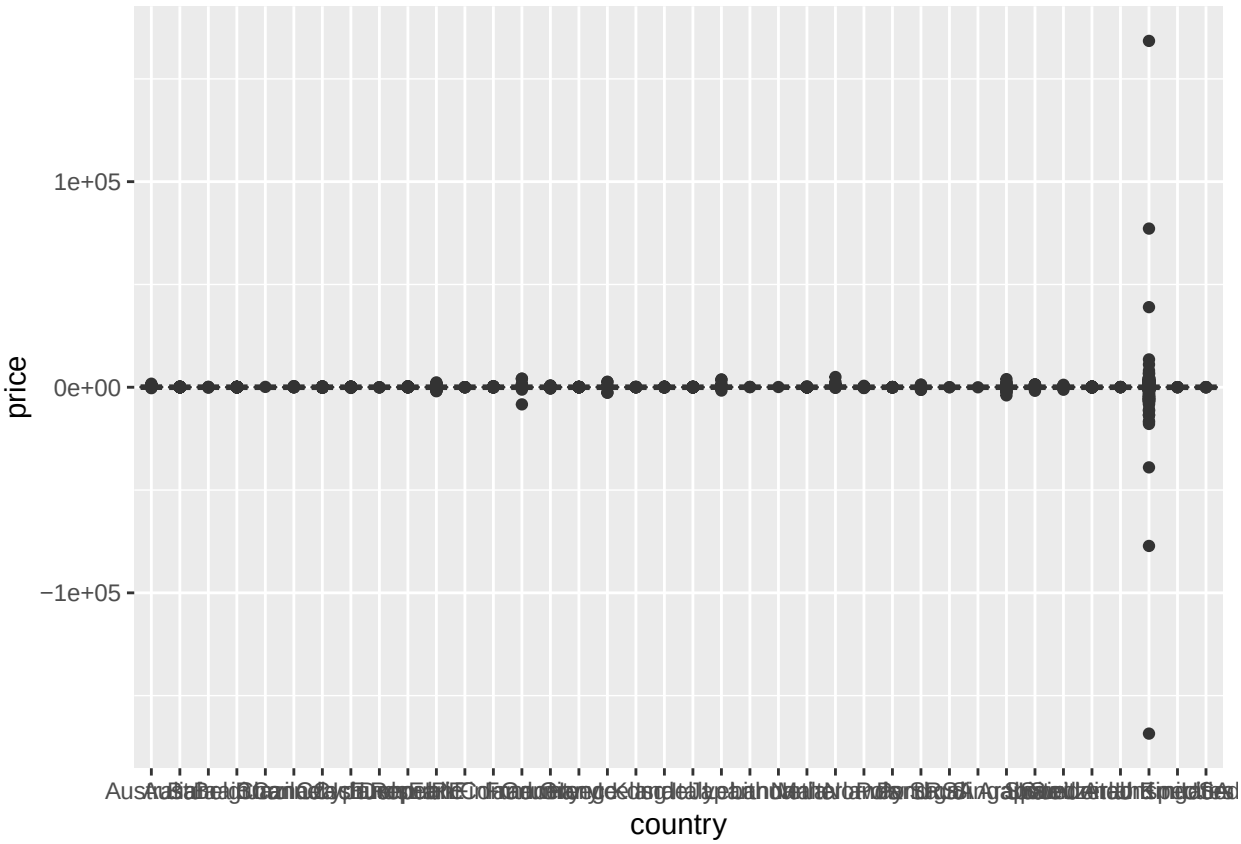
```
total_sales <- data %>%
  group_by(country, status) %>%
  summarize(number = n())

ggplot(data=total_sales) +
  geom_col(aes(x=country, y=number, fill=status)) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  ggtitle("Order per Country")
```




```
data_nor <- data %>%
  filter(country == "Norway")
ggplot(data = data_nor) +
  geom_boxplot(aes(x=country, y=price))
```

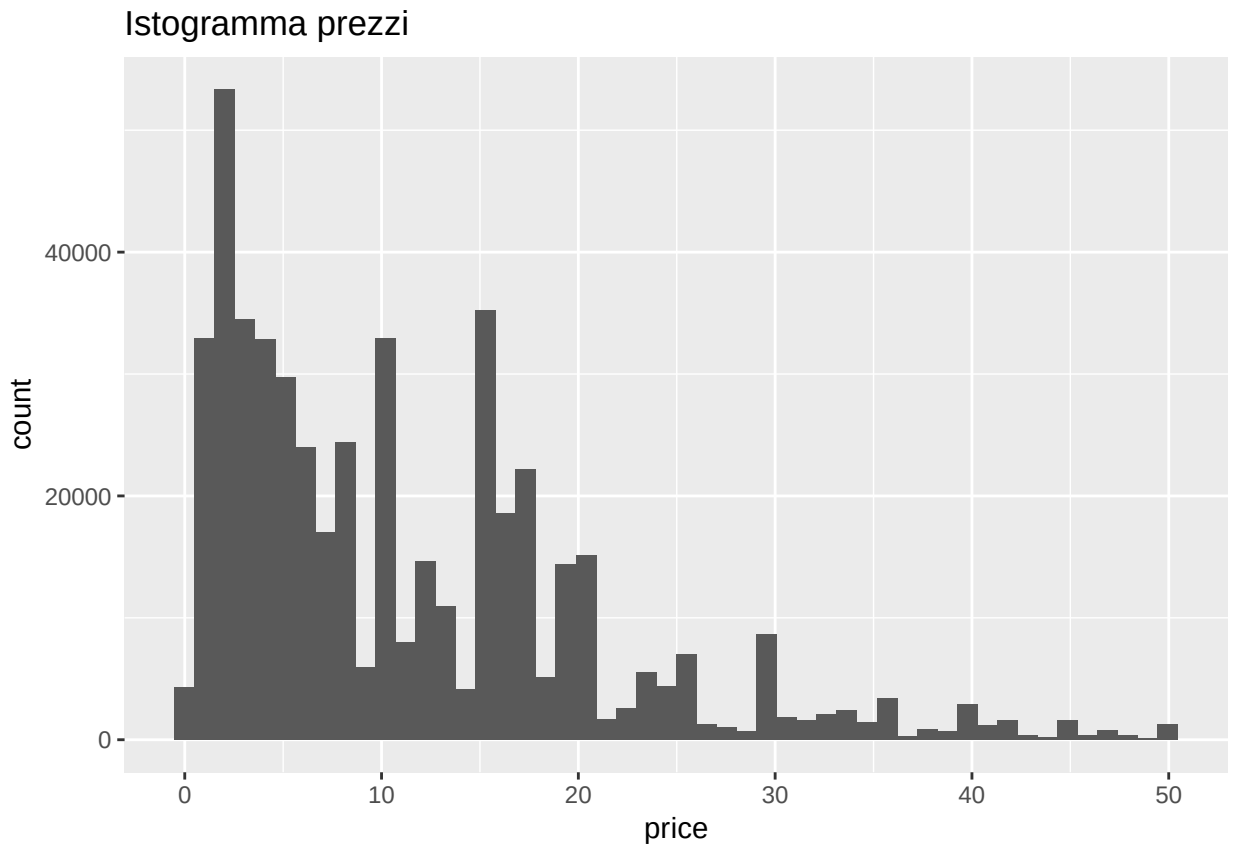



```
ggplot(data = data) +
  geom_boxplot(aes(x=country, y=price))
```

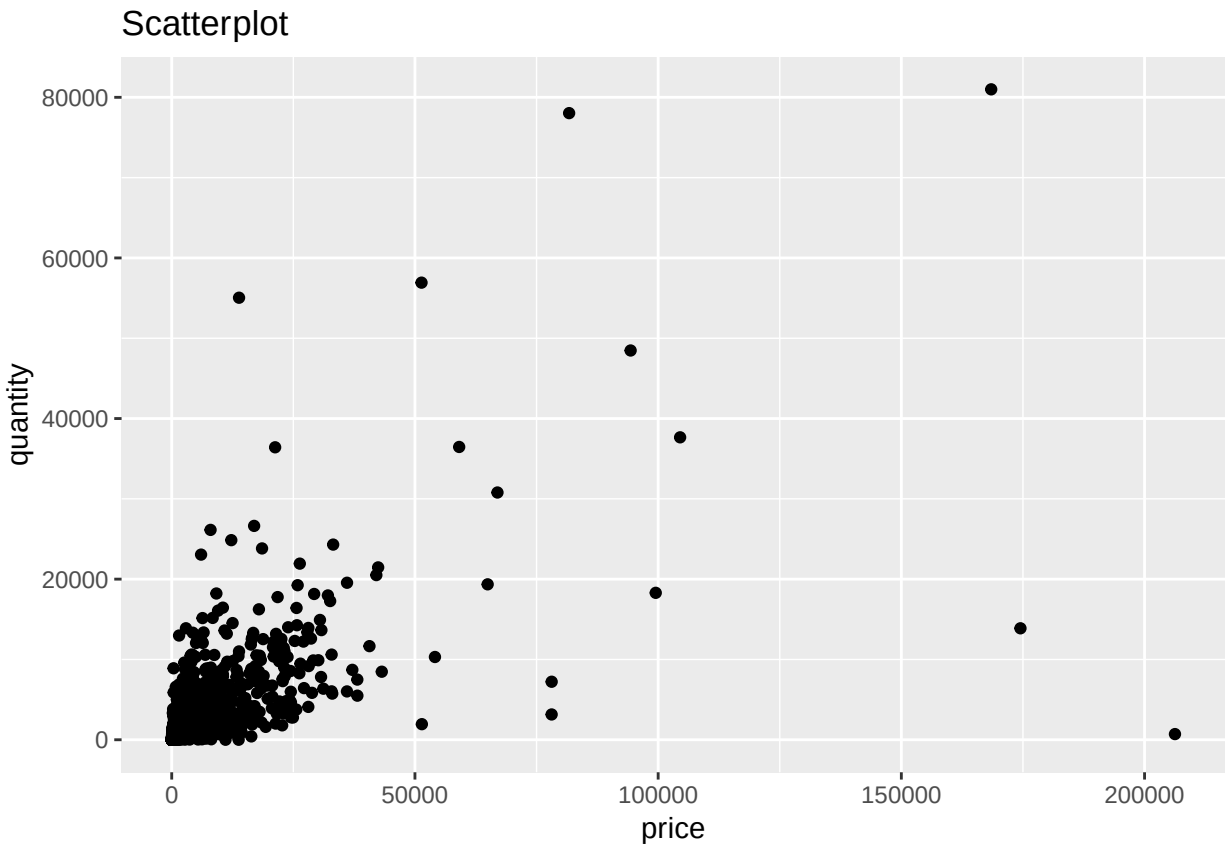


È possibile concatenare un grafico ggplot ad dplyr in questo modo

```
data %>%
  filter(price < 50, price > 0) %>%
  ggplot() +
  geom_histogram(aes(x=price), bins=50) +
  ggtitle("Istogramma prezzi")
```




```
data %>%
  filter(price > 0) %>%
  group_by(stockid) %>%
  summarize(price = sum(price), quantity=sum(quantity)) %>%
  ggplot() +
  geom_point(aes(x=price, y=quantity)) +
  ggtitle("Scatterplot")
```


Scatterplot
80000
60000
40000
20000
0
quantity
0
50000
100000
150000
200000
price