# 1. A new look at an old research study

In 1986, a group of urologists in London published a research paper in *The British Medical Journal* that compared the effectiveness of two different methods to remove kidney stones. Treatment A was open surgery (invasive), and treatment B was percutaneous nephrolithotomy (less invasive). When they looked at the results from 700 patients, treatment B had a higher success rate. However, when they only looked at the subgroup of patients different kidney stone sizes, treatment A had a better success rate. What is going on here? This known statistical phenomenon is called Simpon's paradox. Simpon's paradox occurs when trends appear in subgroups but disappear or reverse when subgroups are combined.

In this notebook, we are going to explore Simpon's paradox using multiple regression and other statistical tools. Let's dive in now!

In [135]:
```
# Load the readr and dplyr packages
library(dplyr)
library(readr)
# Read datasets kidney_stone_data.csv into data
data <- read_csv('datasets//kidney_stone_data.csv')

# Take a look at the first few rows of the dataset
head(data)
```

```
Parsed with column specification:
cols(
  treatment = col_character(),
  stone_size = col_character(),
  success = col_double()
)
```

| treatment | stone_size | success |
|---|---|---|
| B | large | 1 |
| A | large | 1 |
| A | large | 0 |
| A | large | 1 |
| A | large | 1 |
| B | large | 1 |

## 2. Recreate the Treatment X Success summary table

The data contains three columns: `treatment` (A or B), `stone_size` (large or small) and `success` (0 = Failure or 1 = Success). To start, we want to know which treatment had a higher success rate regardless of stone size. Let's create a table with the number of successes and frequency of success by each treatment using the tidyverse syntax.

In [137]:
```
# Calculate the number and frequency of success and failure of each treatmer
data %>%
  group_by(treatment, success) %>%
  summarise(N = n()) %>%
  mutate(Freq = round(N/sum(N), 3))
```

| treatment | success | N | Freq |
|---|---|---|---|
| A | 0 | 77 | 0.220 |
| A | 1 | 273 | 0.780 |
| B | 0 | 61 | 0.174 |
| B | 1 | 289 | 0.826 |

## 3. Bringing stone size into the picture

From the treatment and success rate descriptive table, we saw that treatment B performed better on average compared to treatment A (82% vs. 78% success rate). Now, let's consider stone size and see what happens. We are going to stratify the data into small vs. large stone subcategories and compute the same success count and rate by treatment like we did in the previous task.

The final table will be treatment X stone size X success.

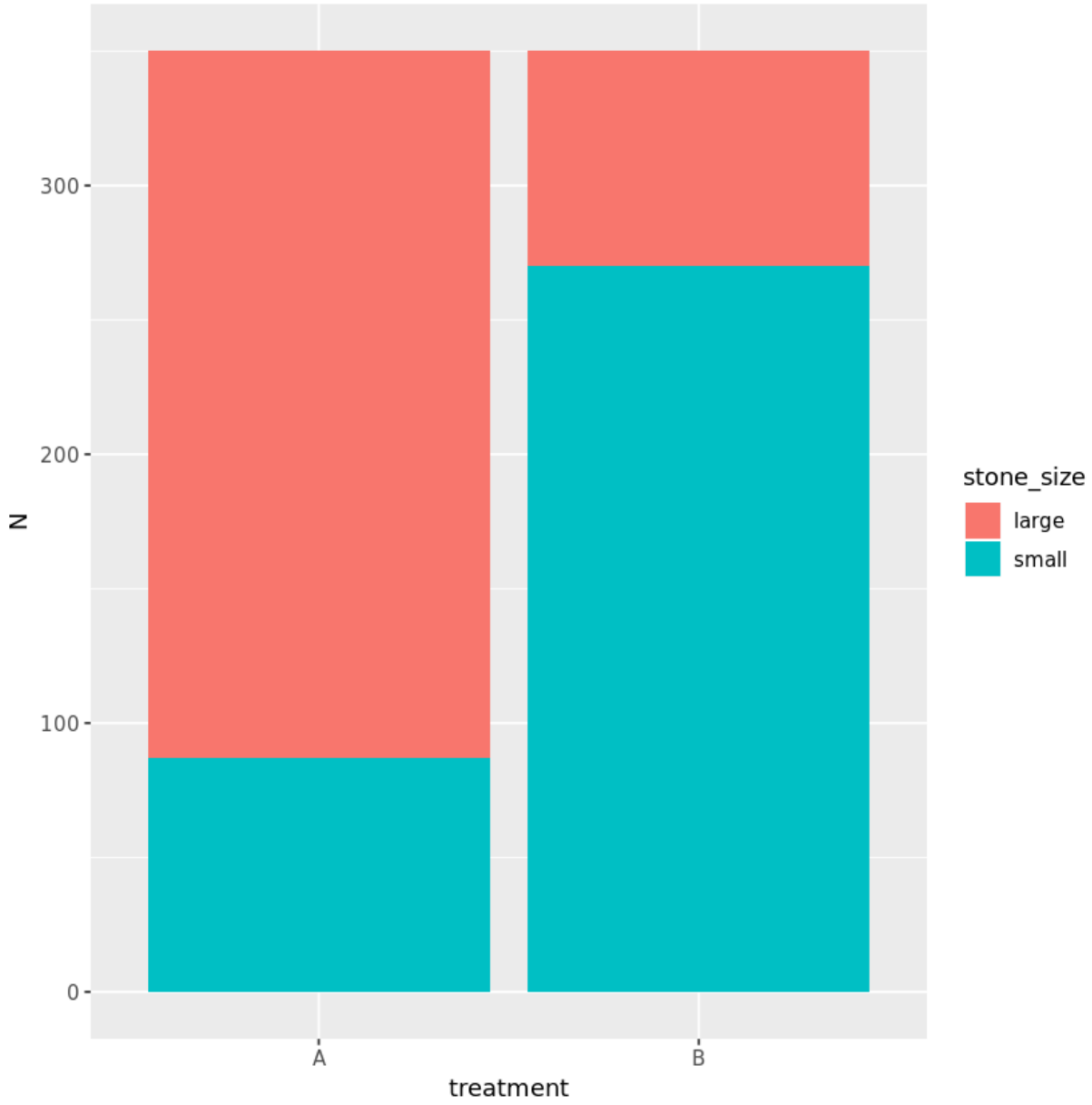
In [139]:
```
# Calculate number and frequency of success and failure by stone size for ea
sum_data <-
  data %>%
  group_by(treatment, stone_size, success) %>%
  summarise(N = n()) %>%
  mutate(Freq = round(N/sum(N),3))

# Print out the data frame we just created
print(sum_data)
```

```
# A tibble: 8 x 5
# Groups:   treatment, stone_size [4]
  treatment stone_size success     N  Freq
  <chr>     <chr>        <dbl> <int> <dbl>
1 A         large            0    71 0.27
2 A         large            1   192 0.73
3 A         small            0     6 0.069
4 A         small            1    81 0.931
5 B         large            0    25 0.312
6 B         large            1    55 0.688
7 B         small            0    36 0.133
8 B         small            1   234 0.867
```

## 4. When in doubt, rely on a plot

What is going on here? When stratified by stone size, treatment A had better results for both large and small stones compared to treatment B (i.e., 73% and 93% v.s. 69% and 87%). Sometimes a plot is a more efficient way to communicate hidden numerical information in the data. In this task, we are going to apply a plotting technique to reveal the hidden information.

In [141]:
```
# Load ggplot2
library(ggplot2)
# Create a bar plot to show stone size count within each treatment
sum_data %>%
  ggplot(aes(x = treatment, y = N)) +
  geom_bar(aes(fill = stone_size), stat='identity')
```




## 5. Identify and confirm the lurking variable

From the bar plot, we noticed an unbalanced distribution of kidney stone sizes in the two treatment options. Large kidney stone cases tended to be in treatment A, while small kidney stone cases tended to be in treatment B. Can we confirm this hypothesis with statistical testing?

Let's analyze the association between stone size (i.e., case severity) and treatment assignment using a statistical test called **Chi-squared**. The **Chi-squared** test is appropriate to test associations between two categorical variables. This test result, together with the common knowledge that a

more severe case would be more likely to fail regardless of treatment, will shed light on the root cause of the paradox.

In [143]:
```
# Load the broom package
library(broom)
# Run a Chi-squared test
trt_ss <- chisq.test(data$treatment, data$stone_size)

# Print out the result in tidy format
print(trt_ss)
```

```
        Pearson's Chi-squared test with Yates' continuity correction

data:  data$treatment and data$stone_size
X-squared = 189.36, df = 1, p-value < 2.2e-16
```

## 6. Remove the confounding effect

After the above exercises, we are confident that stone size/case severity is indeed the lurking variable (aka, confounding variable) in this study of kidney stone treatment and success rate. The good news is that there are ways to get rid of the effect of the lurking variable.

Let's practice using multiple logistic regression to remove the unwanted effect of stone size, and then tidy the output with a function from the `broom` package.

In [145]:
```
# Run a multiple logistic regression
m <- glm(data = data, success ~  treatment + stone_size, family = binomial)

# Print out model coefficient table in tidy format
tidy(m)
```

| term | estimate | std.error | statistic | p.value |
|---|---|---|---|---|
| (Intercept) | 1.0332140 | 0.1344695 | 7.683629 | 1.546436e-14 |
| treatmentB | -0.3572287 | 0.2290792 | -1.559411 | 1.188991e-01 |
| stone_sizesmall | 1.2605654 | 0.2390027 | 5.274272 | 1.332838e-07 |

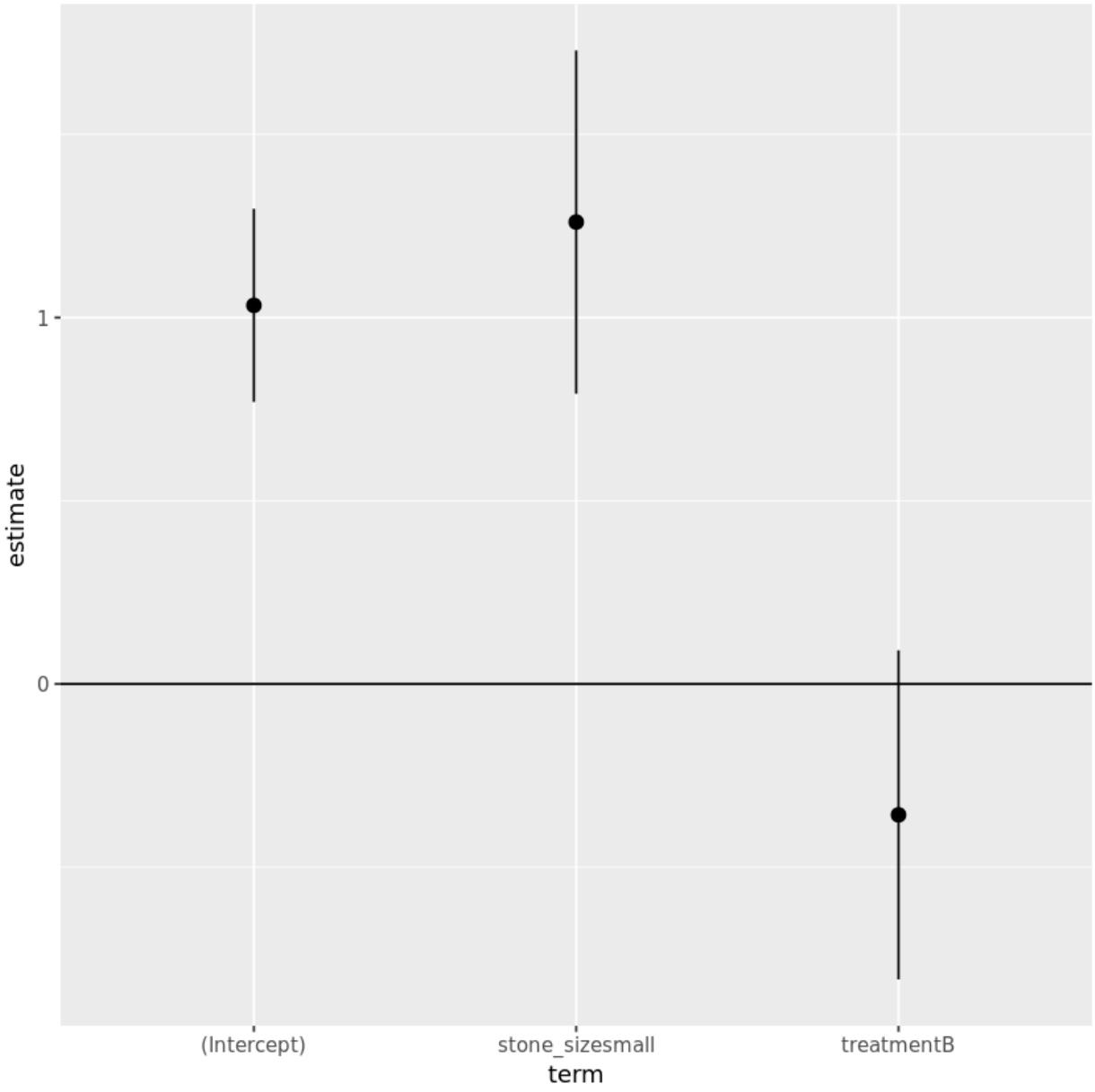
## 7. Visualize model output

We successfully fit a multiple logistic regression and pulled out the model coefficient estimates! Typically (and arbitrarily), P-values below 0.05 indicate statistical significance. Another way to examine whether a significant relationship exists or not is to look at the 95% confidence interval (CI) of the estimate. In our example, we are testing to see:

1. if the effect of a small stone is the same as a big stone, and
2. if treatment A is as effective as treatment B.

If the 95% CI for the coefficient estimates cover **zero**, we cannot conclude that one is different from the other. Otherwise, there is a significant effect.

In [147]:
```
# Save the tidy model output into an object
tidy_m <- tidy(m)

# Plot the coefficient estimates with 95% CI for each term in the model
tidy_m %>%
  ggplot(aes(x = term, y = estimate)) +
  geom_pointrange(aes(ymin = estimate - 1.96 * std.error,
                      ymax = estimate + 1.96 * std.error)) +
  geom_hline(yintercept = 0)
```




## 8. Generate insights

Based on the coefficient estimate plot and the model output table, there is enough information to generate insights about the study. Is treatment A superior to B after taking into account the effect of stone size/severity level?

Everything is in the output table from the regression model. Recall, a coefficient represents the effect size of the specific model term. A positive coefficient means that the term is positively related to the outcome. For categorical predictors, the coefficient is the effect on the outcome relative to the `reference category`. In our study, stone size large and treatment A are the reference categories.

In [149]:
```
# Is small stone more likely to be a success after controlling for treatment
# Options: Yes, No (as string)
small_high_success <- 'Yes'

# Is treatment A significantly better than B?
# Options: Yes, No (as string)
A_B_sig <- 'No'
```