Assessment Part 2: Reshaping Data

Use the following libraries for these questions:

```
library(tidyverse)
library(dslabs)
```

Question 9

1/1 point (graded)

Examine the built-in dataset `co2`. This dataset comes with base R, not `dslabs` - just type `co2` to access the dataset.

Is `co2` tidy? Why or why not?

- `co2` is tidy data: it has one year for each row.
- `co2` is tidy data: each column is a different month.
- `co2` is not tidy: there are multiple observations per column.
- **(selected)** `co2` is not tidy: to be tidy we would have to wrangle it to have three columns (year, month and value), and then each co2 observation would have a row.

✔

Explanation

These data are not tidy because month is a variable and should be stored as a column instead of across multiple columns in the header. There are also multiple observations per row, and each observation should be a different row.

Submit You have used 2 of 2 attempts

Answers are displayed within the problem

Question 10

0/1 point (graded)
Run the following command to define the `co2_wide` object:

```
co2_wide <- data.frame(matrix(co2, ncol = 12, byrow = TRUE)) %>%
      setNames(1:12) %>%
    mutate(year = as.character(1959:1997))
```

Use the `gather` function to make this dataset tidy. Call the column with the CO2 measurements `co2` and call the month column `month`. Name the resulting object `co2_tidy`.

Which code would return the correct tidy format?

- `co2_tidy <- gather(co2_wide,month,co2,year)`
- `co2_tidy <- gather(co2_wide,co2,month,-year)`
- `co2_tidy <- gather(co2_wide,co2,month,year)`
- `co2_tidy <- gather(co2_wide,month,co2,-year)` ✔

✖

Answer
Incorrect:
Try again. The resulting table has the `month` and `co2` column names switched.

Explanation
`gather` takes 4 arguments: (1) the data, (2) the name of the key column, (3) the name of the value column, and optionally (4) the names of any columns not to

gather. We want to collect the months as the key and CO2 as the value while keeping the year column as-is.

Submit You have used 2 of 2 attempts

Answers are displayed within the problem

Question 11

0/1 point (graded)

Use `co2_tidy` to plot CO2 versus month with a different curve for each year:

```
co2_tidy %>% ggplot(aes(as.numeric(month), co2, color = year)) + geom_line()
```

What can be concluded from this plot?

- CO2 concentrations increased monotonically (never decreased) from 1959 to 1997.
- CO2 concentrations are highest around May and the yearly average increased from 1959 to 1997. ✔
- CO2 concentrations are highest around October and the yearly average increased from 1959 to 1997.
- Yearly average CO2 concentrations have remained constant over time.
- CO2 concentrations do not have a seasonal trend.

✖

Answer

Incorrect:

Try again. If average CO2 remained constant, the curves would all overlap or have a random pattern year-to-year, but this is not the case.

Explanation

In every year, the highest CO2 level is around May. The line legend shows that earlier years had lower CO2 levels and that CO2 levels generally increase relative to the previous year. The most recent years have the highest CO2.

Submit You have used 2 of 2 attempts

Answers are displayed within the problem

Question 12

0/1 point (graded)

Load the built-in `admissions` dataset, which contains college admission information for men and women across six majors, and remove the `applicants` percentage column:

```
library(dslabs)
data(admissions)
dat <- admissions %>% select(-applicants)
```

Your goal is to get the data in the shape that has one row for each major, like this:

```
major  men   women
A       62    82
B       63    68
C       37    34
D       33    35
E       28    24
F        6     7
```

Which command could help you to wrangle the data into the desired format?

- ○ `dat_tidy <- spread(dat, major, admitted)`
- ● `dat_tidy <- spread(dat, gender, major)`
- ○ `dat_tidy <- spread(dat, gender, admitted)` ✔
- ○ `dat_tidy <- spread(dat, admitted, gender)`

✖

Answer
Incorrect:

Try again. This table has columns `admitted, men, women` instead of `gender, men, women`.

Explanation
`spread` takes 3 arguments: (1) the data frame, (2) the key to spread across columns, and (3) the value to put in individual cells of the table.

Submit You have used 2 of 2 attempts

Answers are displayed within the problem

Question 13

1/1 point (graded)
Now use the `admissions` dataset to create the object `tmp`, which has columns `major`, `gender`, `key` and `value`:

```
tmp <- gather(admissions, key, value, admitted:applicants)
tmp
```

Combine the key and gender and create a new column called `column_name` to get a variable with the following values: `admitted_men`, `admitted_women`, `applicants_men` and `applicants_women`. Save the new data as `tmp2`.

Which command could help you to wrangle the data into the desired format?

- `tmp2 <- spread(tmp, column_name, key, gender)`
- `tmp2 <- gather(tmp, column_name, c(gender,key))`
- `tmp2 <- unite(tmp, column_name, c(gender, key))`
- `tmp2 <- spread(tmp, column_name, c(key,gender))`
- `tmp2 <- unite(tmp, column_name, c(key, gender))` ✔

Explanation

`unite` takes 3 arguments: (1) the data frame, (2) the name of the new column to create, and (3) a vector of the columns to unite with an underscore, in order.

Submit You have used 2 of 2 attempts

Answers are displayed within the problem

Question 14

0/1 point (graded)

Which function can reshape `tmp2` to a table with six rows and five columns named `major`, `admitted_men`, `admitted_women`, `applicants_men` and `applicants_women`?

- ○ `gather`
- ○ `spread` ✔
- ○ `separate`
- ◉ `unite`

✖

Answer

Incorrect:

Try again. `unite` combines values from multiple columns with an underscore in between, and those values do not become column names.

Explanation

```
spread(tmp2, column_name, value)
```

Submit You have used 2 of 2 attempts

Answers are displayed within the problem