PHW251 Midterm Exam (Fall 2022) Solutions

NAME HERE

DATE HERE

Instructions

This Rmd is your midterm exam. Submit your completed exam by knitting this Rmd to PDF and loading to Gradescope by 11:59 Pacific time on Tuesday, October 25.

When completing this Rmd, please include your answer for each question where it says **ANSWER HERE:**. For questions 11-20, please also provide your code in the indicated code chunks. The code portion will be used to assign partial credit.

When you are done with the exam, please be sure to knit the Rmd directly to PDF. Additionally, keep in mind best practices including ensuring code does not run of the page and not printing entire dataframes within your final PDF. **When knitted, your PDF should be approximately one page per question** (points will be deducted if too long with unnecessary output).

Contents

SECTION 1 - Multiple Choice

[**1 pt each, 10 total**] Each multiple choice question has only one correct answer, unless otherwise specified. Type the letter corresponding to the correct answer(s) after **ANSWER HERE:**. Please do **not** include any of your code/work for this section.

Question 1

Which of the following will return a value of TRUE?

```
x <- 6
y <- 36
```

A. x*x = y
B. x^2>y
C. x != y
D. y<x

ANSWER HERE: C

Question 2

With the lubridate package installed, you have a character vector my_date which contains a date in yyyy-mm-dd format. Which code would correctly yield the last day of the month for my_date?

A. dmy(my_date) + months(1) - days(1)
B. ceiling_date(dmy(my_date), "month") %m-% days(1)
C. as_date(my_date) + months(1) - days(1)
D. ceiling_date(as_date(my_date), "month") %m-% days(1)

ANSWER HERE: D

Question 3

We are interested in subsetting a hypothetical dataframe (called `df`) to only include rows for where city is Los Angeles, Santa Barbara, San Bernardino, or San Diego. Please select the missing code that will make this code operational:

subsetted_df<- filter(df, *MISSING CODE*)

A. city %in% "Los Angeles","Santa Barbara", "San Bernardino", "San Diego"
B. city == c("Los Angeles","Santa Barbara", "San Bernardino", "San Diego")
C. city in c("Los Angeles","Santa Barbara", "San Bernardino", "San Diego")
D. city %in% c("Los Angeles","Santa Barbara", "San Bernardino", "San Diego")

ANSWER HERE: D

Question 4

Which code will create a vector with the following contents?

6, 12, 18, 24, 30

A. seq(6, 30, by=5)
B. 6:30
C. seq(6, 30, length.out=5)
D. multiple(6, 30)

ANSWER HERE: C

Question 5

Which of the code blocks below creates the following plot?

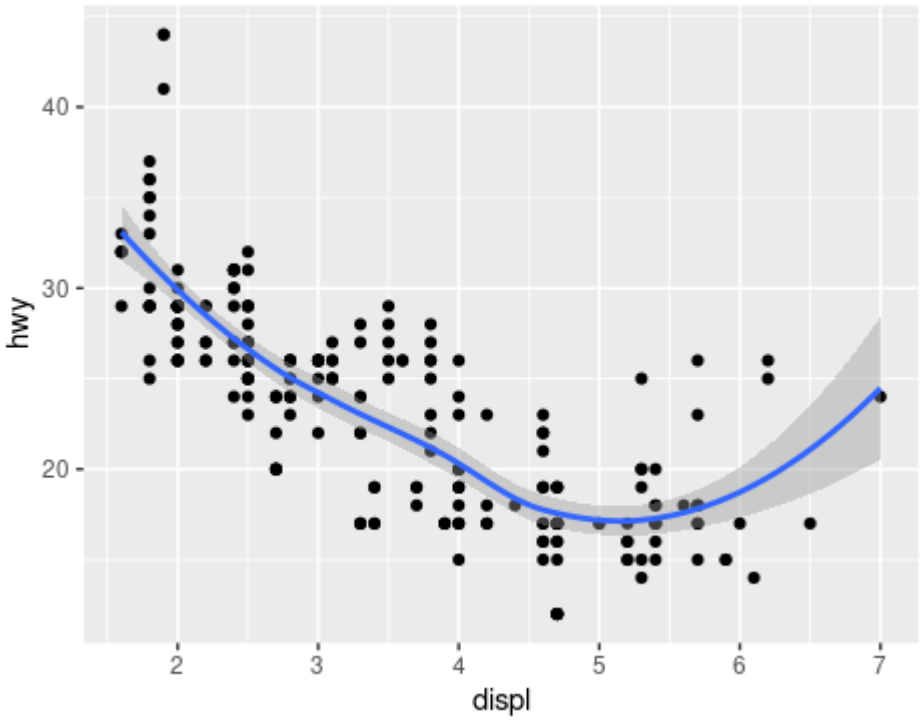

Figure 1: Q5

A. ggplot(data = mpg, mapping = aes(x = hwy, y = displ)), geom_point(), geom_smooth()

B. ggplot(data = mpg, mapping = aes(x = displ, y = hwy)) + geom_point() + geom_smooth()

C. mpg %>% select(displ, hwy) %>% ggplot(mapping = aes(x = hwy, y = displ)) + geom_point() + geom_smooth()

D. ggplot(data = mpg, mapping = aes(x = hwy, y = displ)) %>% geom_point() %>% geom_smooth()

ANSWER HERE: B

Question 6

For the list that is generated by this code:

multi_list <- list("Numbers" = seq(2,22,by=2), "Matrix" = matrix(c(-2,5,8,7,4,22), nrow = 2), "Words"=list("two","four","six", "eight"))

Which of the following will **not** return a single value of 8?

A. multi_list[["Numbers"]][4]
B. multi_list[[2]][1,2]
C. length(multi_list[["Matrix"]])
D. multi_list[[1]][4]

ANSWER HERE: C

Question 7

Data frames and tibbles are two options for storing tabular data in R. What distinguishes data frames and tibbles from one another? Select **all** that apply.

A. Tibbles have less flexibility than data frames for naming columns (i.e. allowing spaces and symbols)
B. Data frames always have row names and tibbles do not
C. Both data frames and tibbles have row names
D. The output will include the data type for each column when printing a tibble, which is not the case when printing data frames

ANSWER HERE: B, D

Question 8

Using the dataframe (`df`) below, which of the following will return the **highlighted row** in the dataframe?

country	year	cases
IND	2021	35
IDN	2021	24
UGA	2022	88
USA	2022	26

Figure 2: Q8

A. df[, 2]
B. unlist(df[2, 0])
C. df[which(df$county==“IDN”),]
D. df[2,]

ANSWER HERE: D

Question 9

Using the df below, which of the following will **not** return this subset data frame?

Original df:

state	year	cases
CA	2020	34
CA	2021	23
AZ	2020	89
AZ	2021	27
TX	2020	32
TX	2021	83

Subset data frame:

state	year	cases
CA	2021	23
AZ	2021	27

Figure 3: Q9

A. `df[which(df$year==2021 & df$cases < 30),]`
B. `df %>% filter(year==2021 | state %in% c("CA","AZ"))`
C. `df %>% filter(year==2021 & cases <30)`
D. `subset(df,year==2021 & cases < 30)`

ANSWER HERE: B

Question 10

There is a need to calculate a temperature in Celcius from Farhenheit ((°Fahrenheit x 5) / 9). If the Celcius temperature is less than or equal to 0°C, the code should print “too cold”, if it is greater than or equal to 30°C it should print “too hot”, and if it is between 0°C and 30°C, it should print “just right”.

Which block of code will print the correct value when

tempC <- ((tempF - 32) * 9)

A. if(tempC >= 30) { print(“too hot”) } else (tempC <= 0){ print(“too cold”) } else { print(“just right”)}

B. if(tempC >= 30) { “too hot” } else if (tempC <= 0){ “too cold” } else { “just right”}

C. if{tempC >= 30} (print(“too hot”)) else if {tempC <= 0{(print(“too cold”)) else (print(“just right”))

D. if(tempC >= 30) { print(“too hot”) } else if (tempC <= 0){ print(“too cold”) } else { print(“just right”)}

ANSWER HERE: D

SECTION 2 - Short Answer

[**15 points total**] All questions below should be answered using R. Unless otherwise specified, you may use any method (base R, tidyverse, or other) to answer these questions.

Please type out your answers in the specified area **ANSWER HERE:** (even if the answer is also available in your code chunk). Code will be used to give partial credit for incorrect answers.

For all questions below, use the "ed_facility_ca.csv" file that is saved on DataHub/GitHub at PHW251_Fall2022/midterm/data/ed_facility_ca.csv. This is a real dataset from the California Health and Human Services Open Data Portal, but has been altered slightly for the purpose of this exam. The dataset contains counts of emergency department encounters at California medical facilities.

The file includes the following columns:

- **Year**
- **OSHPD ID**
- **Facility Name**
- **County Name**
- **ER Service Level Desc:** Level of ER service. Options include - BASIC, COMPREHENSIVE, STANDBY, NOT APPLICABLE
- **Type:** Specifies encounter type. Options include - ED_Visit (Encounter in which patient is treated in the Emergency Department and then released), ED_Admit (Encounter in which the patient is initially treated in the Emergency Department and then admitted to the same hospital for continued inpatient care). Categories are mutually exclusive.
- **Count**

Question 11, Part A

Import the csv data file.

What are the data types of each column when you read the data into R (numeric, factor, logical, character, etc)? **[1 pt]**

ANSWER HERE:

- Year: _____ NUM
- YOSHPD ID: _____ NUM
- Facility Name: _____ CHAR
- County Name: _____ CHAR
- ER Service Level Desc: _____ CHAR
- Type: _____ CHAR
- Count: _____ NUM

```
ed <- read_csv("data/ed_facility_ca.csv")
```

```
## Rows: 6691 Columns: 7
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): Facility Name, County Name, ER Service Level Desc, Type
## dbl (3): Year, OSHPD ID, Count
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
str(ed)
```

```
## spec_tbl_df [6,691 x 7] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Year                 : num [1:6691] 2012 2012 2013 2013 2014 ...
##  $ OSHPD ID             : num [1:6691] 1.06e+08 1.06e+08 1.06e+08 1.06e+08 1.06e+08 ...
##  $ Facility Name        : chr [1:6691] "ALAMEDA HOSPITAL" "ALAMEDA HOSPITAL" "ALAMEDA HOSPITAL" "ALAM
##  $ County Name          : chr [1:6691] "ALAMEDA" "ALAMEDA" "ALAMEDA" "ALAMEDA" ...
##  $ ER Service Level Desc: chr [1:6691] "BASIC" "BASIC" "BASIC" "BASIC" ...
##  $ Type                 : chr [1:6691] "ED_Admit" "ED_Visit" "ED_Admit" "ED_Visit" ...
##  $ Count                : num [1:6691] 2595 13727 2579 13538 2214 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Year = col_double(),
##   ..   `OSHPD ID` = col_double(),
##   ..   `Facility Name` = col_character(),
##   ..   `County Name` = col_character(),
##   ..   `ER Service Level Desc` = col_character(),
##   ..   Type = col_character(),
##   ..   Count = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

Question 12

Notice the column names are not reading in a very user-friendly way. Rename all columns to align with best practices for naming columns (lowercase with underscores in place of spaces).

Paste the new column names **and** the line(s) of code you used to change them below. [**1 pt**]

ANSWER HERE:
"year", "oshpd_id", "facility_name", "county_name","er_service_level_desc","type","count"
rename_with(~ tolower(gsub(" ","_",.x,fixed=TRUE)))

```
ed <- ed %>%
  rename_with(~ tolower(gsub(" ","_",.x,fixed=TRUE)))

# another method using stringr
colnames(ed) <- str_to_lower(gsub(" ", "_", colnames(ed)))

names(ed)
```

```
## [1] "year"                 "oshpd_id"             "facility_name"
## [4] "county_name"          "er_service_level_desc" "type"
## [7] "count"
```

Question 13

Questions 13 through 17 are designed to build off each other.

Using the dataset from question 12, create a new data frame that limits the dataset to only contain rows where the type of service was "basic" and year is between 2015 and 2020.

How many records are in the new subsetted dataset? **[1 pt]**

ANSWER HERE: 3340

```
ed13 <- ed %>%
    filter(er_service_level_desc=="BASIC",year %in% 2015:2020)

count(ed13)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1  3340
```

```
# another filter method to accomplish the same thing
ed13_1 <- ed %>%
    filter(er_service_level_desc=="BASIC" & year >= 2015, year <= 2020)
```

Question 14, Part A

Using the data frame created in question 13, create a new column called `total_encounters` by grouping OSHPD ID and Year and then summing the values in the `count` column to get total encounters. *(Hint: After adding this column your data frame should contain the same number of rows that it had before you added the column.)*

What is the value of `total_encounters` for Alameda Hospital in 2020? **[1 pt]**

ANSWER HERE: 12,115

```
ed14a <- ed13 %>%
  group_by(oshpd_id,year) %>%
  mutate(total_encounters = sum(count)) %>%
  ungroup()

ed14a %>% filter(facility_name=="ALAMEDA HOSPITAL" & year==2020) %>%
  pull(total_encounters)
```

```
## [1] 12115 12115
```

Question 14, Part B

Then create another new column called `pct_encounter_type` that calculates the percent of ED encounters that were visits or admits. Display the percentage as multiplied by 100 and rounded to 1 decimal (for example, 35.1% would be displayed as 35.1).

What is the value of `pct_encounter_type` for ED admits at Sutter Davis Hospital in 2015? **[1 pt]**

ANSWER HERE: 6.1

```
ed14b <- ed14a %>%
  mutate(pct_encounter_type = round(100*count/total_encounters,1))

ed14b %>% filter(facility_name=="SUTTER DAVIS HOSPITAL" & year==2015 & type=="ED_Admit") %>%
  pull(pct_encounter_type)
```

```
## [1] 6.1
```

Question 15, Part A

Using the data frame created in question 14, first create a subset table that only includes rows for ED admits. Then use the arrange function to order the data frame to display rows first by lowest to highest year and then by highest to lowest value of `pct_encounter_type`. Show a single line of code that can be used for this arrange step. **[1 pt]**

ANSWER HERE: arrange(year, desc(pct_encounter_type))

```
ed15a <- ed14b %>%
  filter(type=="ED_Admit") %>%
  arrange(year, desc(pct_encounter_type))
```

Question 15, Part B

What code would you use to obtain only the facility names for the first 5 rows of the dataset created in question 15A (facilities with highest values in the `pct_encounter_type` column for the first year in the data frame)? **[1 pt]**

ANSWER HERE: df[1:5,3]

```
ed15b <- ed15a %>%
  select(facility_name) %>%
  head(5)

ed15b
```

```
## # A tibble: 5 x 1
##   facility_name
##   <chr>
## 1 LOS ANGELES COMMUNITY HOSPITAL AT BELLFLOWER
## 2 OROVILLE HOSPITAL
## 3 COLLEGE MEDICAL CENTER
## 4 NORWALK COMMUNITY HOSPITAL
## 5 MISSION COMMUNITY HOSPITAL - PANORAMA CAMPUS
```

```
ed15b_1 <- ed15a %>%
  select("facility_name") %>%
  slice_head(n=5)

# ed15b_1

# Different ways of getting correct answer
ed15b_2 <- ed15a[1:5,3]

ed15b_3 <- head(ed15a[,3], 5)

ed15b_4 <- head(ed15a, 5)[,("facility_name")]

ed15b_5 <- ed15a[1:5,"facility_name"]
```

Question 16, Part A

Using the dataset created in question 15A, find the average (mean) value of percent of encounter types (`pct_encounter_type`) that were admits among all facilities from 2015-2020. Use this mean value to create another new column called `above_below_avg` that categorizes facilities with `pct_encounter_type` equal to or above average as "above"; otherwise, categorize as "below".

What is the average (mean) percentage of encounters that are ED admits for all facilities from 2015-2020? *[1 pt]*

ANSWER HERE: 13.78

```
#one way
ed16a <- ed15a %>%
  mutate(avg = mean(pct_encounter_type),
         above_below_avg = if_else(pct_encounter_type>=avg,"above","below"))

#another way
summary(ed15a$pct_encounter_type) #summary includes the mean (13.78)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.60    9.10   12.70   13.78   17.00  100.00
```

```
mean(ed15a$pct_encounter_type) #or calculate the average specifically
```

```
## [1] 13.78066
```

```
ed16a_1 <- ed15a %>%
  mutate(above_below_avg = if_else(pct_encounter_type>=13.78,"above","below"))
```

Question 16, Part B

Using the data frame created in question 16A, What is the value of `above_below_avg` for ER admits at West Anaheim Medical Center for 2016? **[1 pt]**

ANSWER HERE: above

```
ed16a %>% filter(year==2016 & facility_name=="WEST ANAHEIM MEDICAL CENTER") %>%
    pull(above_below_avg) #above
```

```
## [1] "above"
```

Question 17, Part A

Restrict the dataset created in 16A to only include records for San Diego, Orange County, and Los Angeles facilities for the year 2020.

How many records remain? **[1 pt]**

ANSWER HERE: 104

```
ed17 <- ed16a %>%
  filter(county_name %in% c("SAN DIEGO","LOS ANGELES","ORANGE") & year == 2020)

count(ed17)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1   104
```

Question 17, Part B

Using the data frame created in question 17A, what hospital has the highest percent of encounters that are admits? [**1 pt**]

ANSWER HERE: COLLEGE MEDICAL CENTER

```
ed17 %>% arrange(desc(pct_encounter_type)) %>% head(1) %>% pull(facility_name)
```

```
## [1] "COLLEGE MEDICAL CENTER"
```

Question 18

Question 18 & 19 utilize the dataframe you imported in questions 11 & 12. To start, create a new subset dataframe that only includes records for encounters that were ED visits (not admits) in the year 2020. Additionally, only include the following columns: Facility Name, County, ER Service Level visits, Type, and Count.

Create a new column called `county_visit_total` that contains the total number of ED visits for each county. *(Hint: after this step, this data frame should contain 1 row per county.)*

Re-order the table to display the county with the highest number of ED visits at the top of the table. What county has the 10th highest total number of ED visits in 2020? **[1 pt]**

ANSWER HERE: Kern

```
ed18a <- ed %>%
  filter(year==2020 & type=="ED_Visit") %>%
  select(facility_name, county_name, er_service_level_desc, type, count)

ed18b <- ed18a %>%
  group_by(county_name,type) %>%
  summarise(county_visit_total = sum(count)) %>%
  ungroup() %>%
  arrange(desc(county_visit_total))
```

```
## `summarise()` has grouped output by 'county_name'. You can override using the
## `.groups` argument.
```

```
head(ed18b,10)
```

```
## # A tibble: 10 x 3
##    county_name    type     county_visit_total
##    <chr>          <chr>                 <dbl>
##  1 LOS ANGELES    ED_Visit            2418500
##  2 SAN DIEGO      ED_Visit             725342
##  3 ORANGE         ED_Visit             670188
##  4 SAN BERNARDINO ED_Visit             654004
##  5 RIVERSIDE      ED_Visit             634860
##  6 SACRAMENTO     ED_Visit             444967
##  7 ALAMEDA        ED_Visit             433225
##  8 SANTA CLARA    ED_Visit             395171
##  9 SAN JOAQUIN    ED_Visit             291980
## 10 KERN           ED_Visit             272825
```

Question 19

Building on to the data frame created in question 18, create a new column called `visit_category` indicating the categorical level of ED visits utilization (High, Medium, Low, Very Low) in each county. The categories should be defined as:

- "High": > 178649 ED visits
- "Medium": > 66521 ED visits
- " Low": > 22026 ED visits
- "Very Low": <= 22026 ED visits

Create a final table that summarizes the number of counties in each category *(Hint: this table should only have 4 rows)*.

How many counties are in the "low" coverage category? **[1 pt]**

ANSWER HERE: 14

```
ed19 <- ed18b %>%
  mutate(visit_category = case_when(
    county_visit_total > 178649 ~ "High",
    county_visit_total > 66521 ~ "Medium",
    county_visit_total > 22026 ~ "Low",
    TRUE ~ "Very low"
  )) %>%
  group_by(visit_category) %>%
  count()

ed19_2 <- ed18b %>%
  mutate(visit_category = case_when(
    county_visit_total <= 22026 ~ "Very low",
    county_visit_total <= 66521 ~ "Low",
    county_visit_total <= 178649 ~ "Medium",
    TRUE ~ "High"
  )) %>%
  group_by(visit_category) %>%
  count()


ed19
```

```
## # A tibble: 4 x 2
## # Groups:   visit_category [4]
##   visit_category     n
##   <chr>          <int>
## 1 High              14
## 2 Low               14
## 3 Medium            13
## 4 Very low          14
```

Question 20, Part A

Question 20 utilizes the dataset as it was first imported in question 11 & 12.

Create a subset dataset with only records for 2016 and basic ER service level. Keep only the following columns: Facility Name, County Name, Type, and Count. Pivot the dataset to create columns for each of the ED encounter types; these columns should each contain the counts of encounters for each type (admit and visit).

How many records are in the dataset after the pivot? **[1 pt]**

ANSWER HERE: 278

```
ed20a_1 <- ed %>%
  filter(year==2016 & er_service_level_desc=="BASIC") %>%
  select(facility_name, county_name, type, count) %>%
  pivot_wider(names_from="type", values_from="count")

#another option
ed20a_2 <- ed %>%
  filter(year==2016 & er_service_level_desc=="BASIC") %>%
  select(facility_name, county_name, type, count) %>%
  pivot_wider(names_from=type, values_from=count)

count(ed20a_1)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1   278
```

```
count(ed20a_2)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1   278
```

Question 20, Part B

Include the line of code used to perform the pivot. Make sure to include the function name as well as the arguments used. *[1 pt]*

ANSWER HERE: Both are correct: pivot_wider(names_from=type, values_from=count) pivot_wider(names_from="typ values_from="count")

```
q20b_1 <- ed %>% pivot_wider(names_from=type, values_from=count)
```

```
## Warning: Values from `count` are not uniquely identified; output will contain list-cols.
## * Use `values_fn = list` to suppress this warning.
## * Use `values_fn = {summary_fun}` to summarise duplicates.
## * Use the following dplyr code to identify duplicates.
##   {data} %>%
##     dplyr::group_by(year, oshpd_id, facility_name, county_name, er_service_level_desc, type) %>%
##     dplyr::summarise(n = dplyr::n(), .groups = "drop") %>%
##     dplyr::filter(n > 1L)
```

```
q20b_2 <- ed %>% pivot_wider(names_from="type", values_from="count")
```

```
## Warning: Values from `count` are not uniquely identified; output will contain list-cols.
## * Use `values_fn = list` to suppress this warning.
## * Use `values_fn = {summary_fun}` to summarise duplicates.
## * Use the following dplyr code to identify duplicates.
##   {data} %>%
##     dplyr::group_by(year, oshpd_id, facility_name, county_name, er_service_level_desc, type) %>%
##     dplyr::summarise(n = dplyr::n(), .groups = "drop") %>%
##     dplyr::filter(n > 1L)
```

EXTRA CREDIT

[2 points total]

Complete questions #18 and #19 using only one dplyr call. In other words, start with the dataset imported in question 11, perform the necessary subsetting, grouping, and summarizing with the end goal of producing a table that displays the number of counties in each visit count category.

Please include sufficient code for the teaching team to be able to run your code, if needed. This means either including the import statement for the csv before the dplyr call, or including the import statement as part of your dplyr call.

Hint: Including more than one `group_by()` in a single call may also require the use of `ungroup()`.

Paste the single dplyr call below. **[1 pt]**

```
ed_bonus_q1 <- ed %>%
  filter(year==2020 & type=="ED_Visit") %>%
  select(facility_name, county_name, er_service_level_desc, type, count) %>%
  group_by(county_name) %>%
  summarise(county_visit_total = sum(count)) %>%
  ungroup() %>%
  arrange(desc(county_visit_total)) %>%
  mutate(visit_category = case_when(
    county_visit_total > 178649 ~ "High",
    county_visit_total > 66521 ~ "Medium",
    county_visit_total > 22026 ~ "Low",
    TRUE ~ "Very low"
  )) %>%
  group_by(visit_category) %>%
  count()

ed_bonus_q1
```

```
## # A tibble: 4 x 2
## # Groups:   visit_category [4]
##   visit_category     n
##   <chr>          <int>
## 1 High              14
## 2 Low               14
## 3 Medium            13
## 4 Very low          14
```

```
identical(ed_bonus_q1, ed19) #show they are the same
```

```
## [1] TRUE
```

Table 1: ED Visit Utilization by Visit Category

Visit Category	Total Counties
High	14
Medium	13
Low	14
Very low	14

Include code that uses the `kable` package to print the final table for question 19 in a print-friendly format (easy to read with meaningful column names and rows in descending order from High to Very Low). **[1 pt]**

```
library(kableExtra)
```

```
##
## Attaching package: 'kableExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     group_rows
```

```
ed_bonus_q2 <- ed19 %>%
  arrange(c("High", "Medium", "Low", "Very low"))
ed_bonus_q2
```

```
## # A tibble: 4 x 2
## # Groups:   visit_category [4]
##   visit_category     n
##   <chr>          <int>
## 1 High              14
## 2 Medium            13
## 3 Low               14
## 4 Very low          14
```

```
kable(ed_bonus_q2,
      caption = "ED Visit Utilization by Visit Category",
      col.names = c("Visit Category", "Total Counties"))
```