Biostat 203B Homework 3

Due Feb 21 @ 11:59PM

Amaan Jogia-Sattar, 206324648

Display machine information for reproducibility:

```
sessionInfo()
```

```
R version 4.4.2 (2024-10-31)
Platform: aarch64-apple-darwin20
Running under: macOS Sequoia 15.3.1

Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/lib/libRlapack.dylib;  l

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

time zone: America/Los_Angeles
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

loaded via a namespace (and not attached):
 [1] compiler_4.4.2    fastmap_1.2.0     cli_3.6.3         tools_4.4.2
 [5] htmltools_0.5.8.1 rstudioapi_0.17.1 yaml_2.3.10       rmarkdown_2.28
 [9] knitr_1.48        jsonlite_1.8.9    xfun_0.48         digest_0.6.37
[13] rlang_1.1.4       evaluate_1.0.1
```

Load necessary libraries (you can add more as needed).

```
library(arrow)
```

```
Attaching package: 'arrow'

The following object is masked from 'package:utils':

    timestamp
```

```
library(gtsummary)
library(memuse)
library(pryr)
```

```
Attaching package: 'pryr'

The following object is masked from 'package:gtsummary':

    where
```

```
library(R.utils)
```

```
Loading required package: R.oo

Loading required package: R.methodsS3

R.methodsS3 v1.8.2 (2022-06-13 22:00:14 UTC) successfully loaded. See ?R.methodsS3 for help.

R.oo v1.27.0 (2024-11-01 18:00:02 UTC) successfully loaded. See ?R.oo for help.

Attaching package: 'R.oo'

The following object is masked from 'package:R.methodsS3':

    throw
```

```
The following objects are masked from 'package:methods':

    getClasses, getMethods

The following objects are masked from 'package:base':

    attach, detach, load, save

R.utils v2.12.3 (2023-11-18 01:00:02 UTC) successfully loaded. See ?R.utils for help.


Attaching package: 'R.utils'

The following object is masked from 'package:arrow':

    timestamp

The following object is masked from 'package:utils':

    timestamp

The following objects are masked from 'package:base':

    cat, commandArgs, getOption, isOpen, nullfile, parse, use, warnings
```

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.1
v purrr     1.0.2

-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x purrr::compose()      masks pryr::compose()
x lubridate::duration() masks arrow::duration()
x tidyr::extract()      masks R.utils::extract()
x dplyr::filter()       masks stats::filter()
```

```
x dplyr::lag()          masks stats::lag()
x purrr::partial()      masks pryr::partial()
x dplyr::where()        masks pryr::where(), gtsummary::where()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becor
```

```
library(lubridate)
```

Display your machine memory.

```
memuse::Sys.meminfo()
```

```
Totalram:  36.000 GiB
Freeram:    7.517 GiB
```

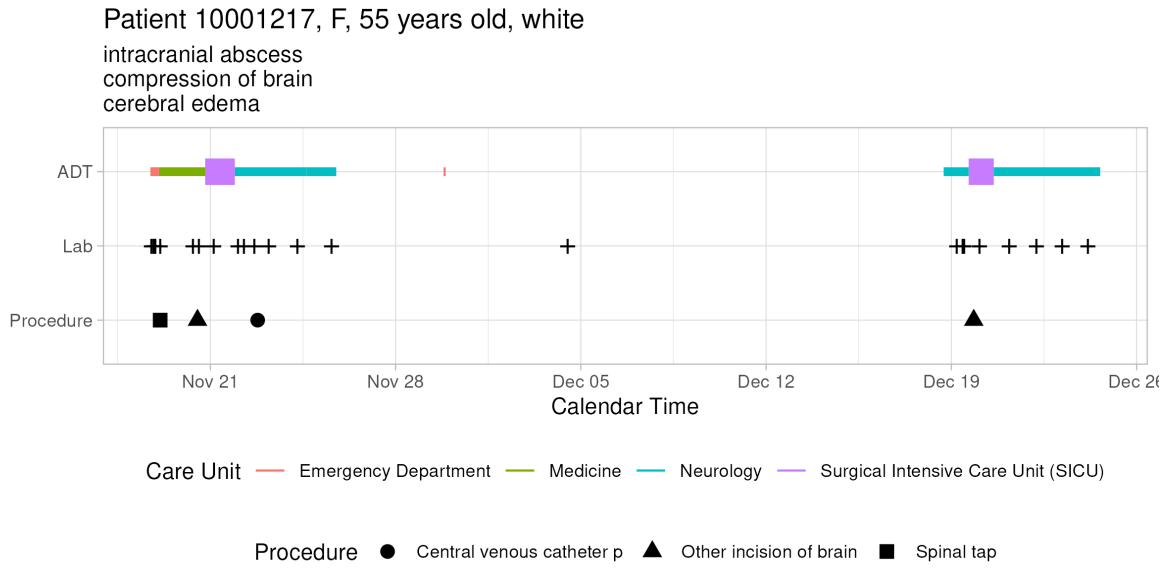
In this exercise, we use tidyverse (ggplot2, dplyr, etc) to explore the MIMIC-IV data introduced in homework 1 and to build a cohort of ICU stays.

Q1. Visualizing patient trajectory

Visualizing a patient's encounters in a health care system is a common task in clinical data analysis. In this question, we will visualize a patient's ADT (admission-discharge-transfer) history and ICU vitals in the MIMIC-IV data.

Q1.1 ADT history

A patient's ADT history records the time of admission, discharge, and transfer in the hospital. This figure shows the ADT history of the patient with `subject_id` 10001217 in the MIMIC-IV data. The x-axis is the calendar time, and the y-axis is the type of event (ADT, lab, procedure). The color of the line segment represents the care unit. The size of the line segment represents whether the care unit is an ICU/CCU. The crosses represent lab events, and the shape of the dots represents the type of procedure. The title of the figure shows the patient's demographic information and the subtitle shows top 3 diagnoses.



Do a similar visualization for the patient with `subject_id` 10063848 using ggplot.

Hint: We need to pull information from data files `patients.csv.gz`, `admissions.csv.gz`, `transfers.csv.gz`, `labevents.csv.gz`, `procedures_icd.csv.gz`, `diagnoses_icd.csv.gz`, `d_icd_procedures.csv.gz`, and `d_icd_diagnoses.csv.gz`. For the big file `labevents.csv.gz`, use the Parquet format you generated in Homework 2. For reproducibility, make the Parquet folder `labevents_pq` available at the current working directory `hw3`, for example, by a symbolic link. Make your code reproducible.

Solution: First, we will create a symbolic link to the `.parquet` file we created in`HW2`. This is done in the terminal.

Since we want to make sure that our code will be reproducible with any pre-specified `subject_id`, we will parameterize this value: We will load the `labevents` dataset via our symbolic link, filtering it by our `subject_id` of interest. We collect the values and are now ready to load in the other datasets from which we will be drawing information.

```
# Specify the subject id of interest
subjid <- 10063848

# Create labevents data.frame
labevents <- arrow::open_dataset('labevents_pq') %>%
  filter(subject_id == subjid) %>%
  collect()

# Get patient data
patients <- read_csv('~/mimic/hosp/patients.csv.gz') %>%
  filter(subject_id == subjid)
```

```
Rows: 364627 Columns: 6
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (2): gender, anchor_year_group
dbl  (3): subject_id, anchor_age, anchor_year
date (1): dod

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.
```

```
# Get admissions data
admissions <- read_csv('~/mimic/hosp/admissions.csv.gz') %>%
  filter(subject_id == subjid)
```

```
Rows: 546028 Columns: 16
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (8): admission_type, admit_provider_id, admission_location, discharge_l...
dbl  (3): subject_id, hadm_id, hospital_expire_flag
dttm (5): admittime, dischtime, deathtime, edregtime, edouttime

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.
```

```
# Get transfers data
transfers <- read_csv('~/mimic/hosp/transfers.csv.gz') %>%
  filter(subject_id == subjid)
```

```
Rows: 2413581 Columns: 7
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (2): eventtype, careunit
dbl  (3): subject_id, hadm_id, transfer_id
dttm (2): intime, outtime

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.
```

```r
# Get procedures_icd data
procedures_icd <- read_csv('~/mimic/hosp/procedures_icd.csv.gz') %>%
  filter(subject_id == subjid)
```

```
Rows: 859655 Columns: 6
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (1): icd_code
dbl  (4): subject_id, hadm_id, seq_num, icd_version
date (1): chartdate

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.
```

```r
# Get diagnoses_icd data
diagnoses_icd <- read_csv('~/mimic/hosp/diagnoses_icd.csv.gz') %>%
  filter(subject_id == subjid)
```

```
Rows: 6364488 Columns: 5
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (1): icd_code
dbl (4): subject_id, hadm_id, seq_num, icd_version

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.
```

```r
# Get d_icd_procedures data
d_icd_procedures <- read_csv('~/mimic/hosp/d_icd_procedures.csv.gz')
```

```
Rows: 86423 Columns: 3
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (2): icd_code, long_title
dbl (1): icd_version

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.
```

```r
# Get d_icd_diagnoses data
d_icd_diagnoses <- read_csv('~/mimic/hosp/d_icd_diagnoses.csv.gz')
```

```
Rows: 112107 Columns: 3
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (2): icd_code, long_title
dbl (1): icd_version

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.
```

Now, we will begin organizing our data. From `admissions`, we know we will need `admittime`, `dischtime`, and to specify the event type as `ADT`. We first slice off our demographic information and label it `patient_info`. Next, we add the `event_type` of `ADT` to our admissions dataframe, properly format `admittime` and `dischtime`, and select our columns of interest.

```r
# Prepare patient demographics
patient_info_viz <- patients %>%
  filter(subject_id == subjid) %>%
  select(subject_id,
         gender,
         anchor_age) %>%  # Get gender & age from patients
  left_join(
    admissions %>% select(
      subject_id,
      race),  # Get race from admissions
    by = 'subject_id'
  ) %>%
  slice(1)
# Format gender properly
patient_info_viz <- patient_info_viz %>%
  mutate(gender = ifelse(gender == 'M',
                         'Male',
                         'Female'),
         race = str_to_title(race))

# Prepare admissions_viz info
admissions_viz <- admissions %>%
  mutate(
    event_type = 'ADT',
```

```
    admittime = ymd_hms(admittime),
    dischtime = ymd_hms(dischtime)
  ) %>%
  select(
    subject_id,
    admittime,
    dischtime,
    event_type
  )

# prepare transfers_viz info
transfers_viz <- transfers %>%
  mutate(
    event_type = 'ADT',
    admittime = ymd_hms(intime),
    dischtime = ymd_hms(outtime)
  ) %>%
  select(
    subject_id,
    admittime,
    dischtime,
    event_type,
    careunit
  )

# Merge admissions and transfers information
adt_events_viz <- bind_rows(
  admissions_viz,
  transfers_viz
) %>%
  arrange(admittime)

# Account for missing care_unit values
adt_events_viz <- adt_events_viz %>%
  mutate(
    careunit = ifelse(is.na(careunit) | careunit == '',
                      'UNKNOWN',
                      careunit)
  )
# Account for ICU/CCU Status
adt_events_viz <- adt_events_viz %>%
  mutate(
```

```
    is_icu_ccu = ifelse(str_detect(str_to_lower(careunit), 'icu|ccu'), 'ICU/CCU', 'Other')
  )
```

Now we have organized `Admits, Transfers, and Discharges (ADT)`. We will proceed to prepare our lab events, which correspond to crosses in our visualization. To do this, we label an `event_type`, convert `charttime` to `UTC`, and select columns of interest. We can proceed to work with procedure information, which relies on`procedures_icd` as well as `d_icd_procedures`. We note that exact timestamps are not provided in the `procedures_icd` dataframe (dates only), so they take on a default value of `YYYY-MM-DD 00:00:00`.

```
# Prepare labevents_viz
labevents_viz <- labevents %>%
  mutate(
    event_type = 'Lab',
    charttime = with_tz(ymd_hms(charttime),
                        'UTC')
  ) %>%
  select(
    subject_id,
    charttime,
    event_type
  )

# Prepare procedures
procedures_viz <- procedures_icd %>%
  left_join(d_icd_procedures,
            by = 'icd_code') %>%
  mutate(
    event_type = 'Procedure',
    procedure_time = ymd(chartdate),
    procedure_desc = coalesce(long_title,
                              'Unknown')
    ) %>%
  select(
    subject_id,
    procedure_time,
    event_type,
    procedure_desc
  )

top_diagnoses <- diagnoses_icd %>%
  filter(subject_id == subjid) %>%
```

```
  left_join(d_icd_diagnoses, by = 'icd_code') %>%
  arrange(seq_num) %>%              # sort by sequence number
  distinct(long_title,
           .keep_all = TRUE) %>%  # keep only one row per unique diagnosis
  slice_head(n = 3) %>%             # take top 3 unique diagnoses
  pull(long_title)
```

```
Warning in left_join(., d_icd_diagnoses, by = "icd_code"): Detected an unexpected many-to-ma
i Row 17 of `x` matches multiple rows in `y`.
i Row 15793 of `y` matches multiple rows in `x`.
i If a many-to-many relationship is expected, set `relationship =
  "many-to-many"` to silence this warning.
```

Lastly, we will combine all of our information before creating the visualization. We will also set up some preliminaries for the visualization process. First, we factorize the event type, as we are ordering the chart based on it. Secondly, we will create the title text based on our patient demographic information. We ensure that all columns we are binding on are named uniformly, and we arrange our aggregated data.

```
timeline_events_viz <- bind_rows(
  adt_events_viz,
  labevents_viz %>%
    rename(admittime = charttime),
  procedures_viz %>%
    rename(admittime = procedure_time)
) %>%
  arrange(admittime)

# Factorize event_type to ensure proper ordering
timeline_events_viz$event_type <- factor(
  timeline_events_viz$event_type,
  levels = c('ADT',
             'Lab',
             'Procedure'),  # Correct order: ADT → Lab → Procedure
  ordered = TRUE
)


  # Format title with demographics
title_text <- paste0(
  'Patient ',
```

```
  patient_info_viz$subject_id,
  ', ',
  patient_info_viz$gender,
  ", ",
  patient_info_viz$anchor_age,
  ' years old, ',
  patient_info_viz$race
)
# Format subtitle (line break for each diagnosis)
subtitle_text <- paste(
  top_diagnoses,
  collapse = '\n')
```

Now it is time for visualization.

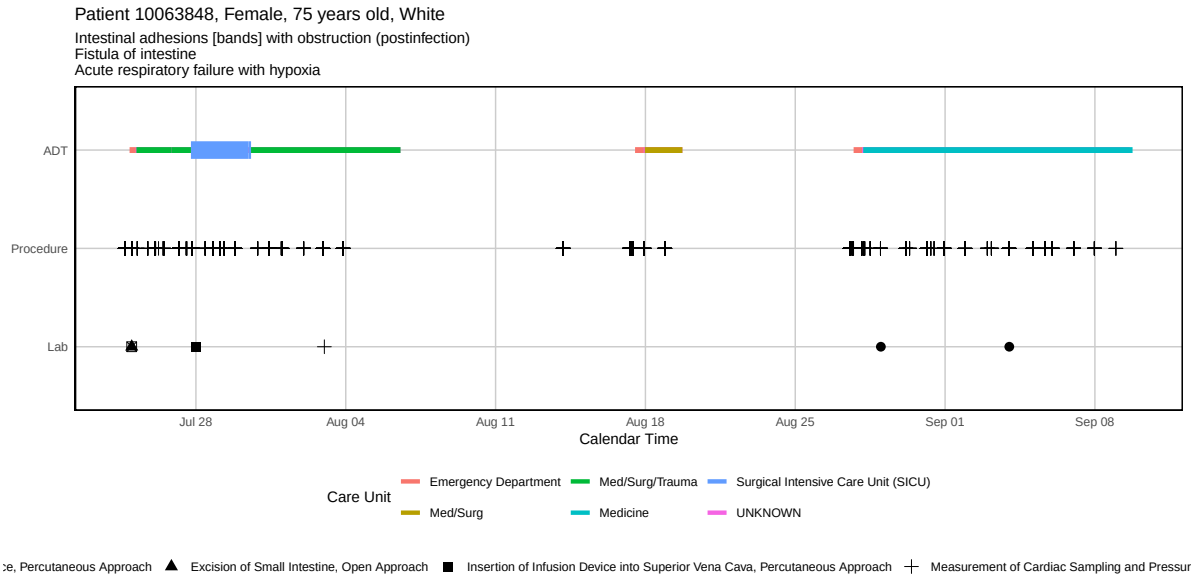
```
# Order the Event Type for Plotting
timeline_events_viz <- timeline_events_viz %>%
  mutate(
    y_position = case_when(
      event_type == 'ADT' ~ 3,
      event_type == 'Procedure' ~ 1,
      event_type == 'Lab' ~ 2,
      TRUE ~ NA_real_
    )
  )

ggplot(data = timeline_events_viz) +
  # 1) ADT as segments at y_position = 3
  geom_segment(
    data = timeline_events_viz %>%
      filter(event_type == 'ADT'),
    aes(
      x = admittime,
      xend = dischtime,
      y = y_position,
      yend = y_position,
      color = careunit,
      linewidth = is_icu_ccu == 'ICU/CCU'  # Thicker lines for ICU/CCU
    )
  ) +

  # 2) Procedure as points at y_position = 1
```

```
  geom_point(
    data = timeline_events_viz %>%
      filter(event_type == 'Procedure'),
    aes(
      x = admittime,
      y = y_position,
      shape = procedure_desc
    ),
    size = 3, color = "black"
  ) +

  # 3) Lab as points at y_position = 2
  geom_point(
    data = timeline_events_viz %>% filter(event_type == 'Lab'),
    aes(
      x = admittime,
      y = y_position
    ),
    shape = 3, size = 3, color = 'black'  # shape=3 = plus sign
  ) +

  # 4) Control the x-axis breaks/labels
  scale_x_datetime(date_breaks = '1 week',
                   date_labels = '%b %d') +

  # 5) Manually control the y-axis
  scale_y_continuous(
    name = NULL,
    breaks = c(1, 2, 3),
    limits = c(0.5, 3.5),
    labels = c('Lab',
               'Procedure',
               'ADT')
  ) +

  # 6) Title, subtitle, legend labels
  labs(
    title = paste0(
      'Patient ', patient_info_viz$subject_id, ', ',
      patient_info_viz$gender, ', ',
      patient_info_viz$anchor_age, ' years old, ',
      patient_info_viz$race
```

```
    ),
    subtitle = subtitle_text,
    x = 'Calendar Time',
    y = NULL,
    color = 'Care Unit',
    shape = 'Procedure Type'
  ) +

  # 7) Use a theme (you can keep theme_minimal or switch to theme_light)
  theme_minimal() +
  theme(
    panel.border = element_rect(color = 'black',
                                fill = NA,
                                linewidth = 1),
    panel.grid.major = element_line(color = 'gray80'),
    panel.grid.minor = element_blank(),
    axis.line = element_blank(),
    legend.position = 'bottom',
    legend.box = 'vertical'
  ) +

  # 8) Control the legend ordering and remove the linewidth legend
  guides(
    color = guide_legend(
      override.aes = list(linewidth = 1.5),
      order = 1
    ),
    shape = guide_legend(
      order = 2
    ),
    linewidth = 'none'
  )
```

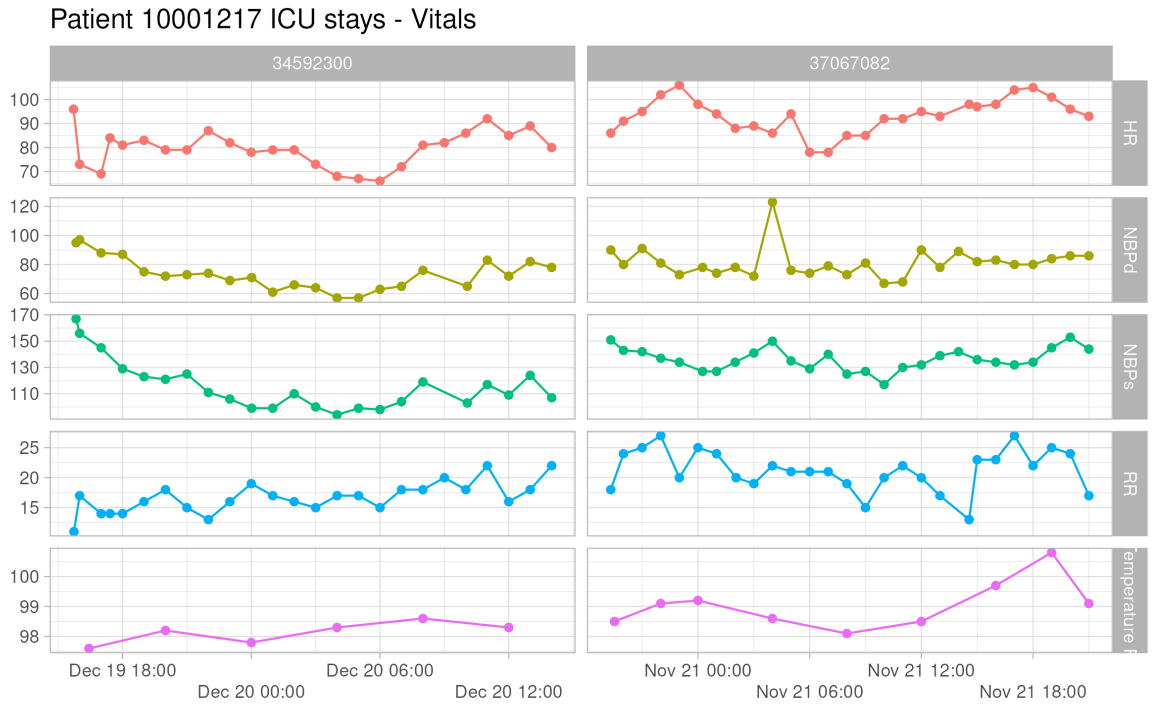
```
Warning: Using linewidth for a discrete variable is not advised.
```

```
Warning: Removed 3 rows containing missing values or values outside the scale range
(`geom_segment()`).
```



Q1.2 ICU stays

ICU stays are a subset of ADT history. This figure shows the vitals of the patient `10001217` during ICU stays. The x-axis is the calendar time, and the y-axis is the value of the vital. The color of the line represents the type of vital. The facet grid shows the abbreviation of the vital and the stay ID.



Do a similar visualization for the patient `10063848`. **Solution**: First, we must acquire our dataset. Once again, we will borrow this from our previous assignment using a symbolic link, which has been done in the terminal. I reference this file as `chartevents_parquet`. Now, we will load in the dataset, filtering for our preselected patient. We will attach the abbreviations for vital measurements by performing a join on `itemid` between `chartevents_viz` and `items`. We will convert timezone to UTC to ensure proper formatting for the plot. We now have all of our information aggregated into a singular dataframe, and we can begin plotting.

```
items <- read_csv('~/mimic/icu/d_items.csv.gz')
```

```
Rows: 4095 Columns: 9
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (6): label, abbreviation, linksto, category, unitname, param_type
dbl (3): itemid, lownormalvalue, highnormalvalue

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.
```

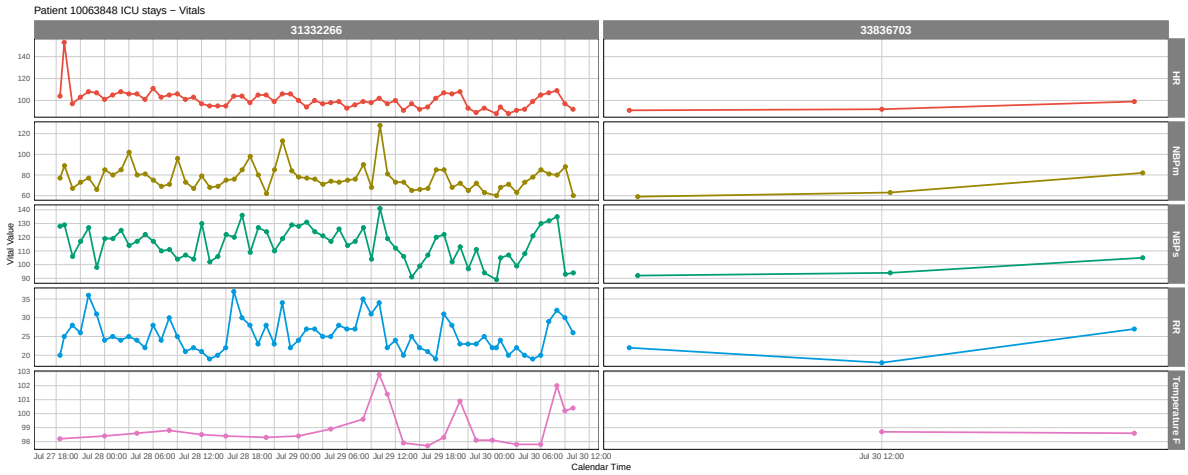
```
chartid <- 10063848
chartevents_viz <- arrow::open_dataset('chartevents_pq') %>%
  select(subject_id, itemid, charttime, valuenum, stay_id) %>%
  filter(subject_id == chartid,
         itemid %in% c(
           220045,  # Mean Arterial Pressure
           220181,  # Arterial Blood Pressure systolic
           220179,  # Arterial Blood Pressure diastolic
           223761,  # Heart Rate
           220210   # Respiratory Rate
         )) %>%
  arrange(subject_id, charttime, itemid) %>%
  collect()

# Attach abbreviations
chartevents_viz <- chartevents_viz %>%
  left_join(
    items %>% select(itemid, abbreviation),
    by = 'itemid'
  )
# Fix dates
chartevents_viz <- chartevents_viz %>%
  mutate(charttime = with_tz(charttime, 'UTC'))
```

In the plotting stage, we specify grouping by both the unique `stay_id` as well as the particular vital measurement being taken (`abbreviation`). We grid our plots such that each row corresponds to its own vital measurement, and each column corresponds to a specific stay. We specify the endpoints of our plots based on the minimum and maximum values in `charttime`, and we specify for line breaks to occur every three hours. Additionally, we specify that labeling should be done on every other xtick, starting with the second xtick. We also specify the manner in which we would like to format timestamps as `%b %d %H:%M`. We match the colors for each vital measurement to the colors in the example plot, and we label axes and titles accordingly. Lastly, we format the strip labels on the x and y axes, such that they are filled with grey and text is white. We specify that we would like the y-axis' strip to be on the right-hand side, while the x-axis' strip remains in its default position on top. We place our strips outside the plot itself, and ensure that x-axis labels are horizontal (as opposed to diagonal). Lastly, we remove the legend for vital measurement, as we have these labels in the y-axis' strip. The code for this procedure is below:

```
ggplot(chartevents_viz, aes(x = charttime, y = valuenum, color = abbreviation,
                            group = interaction(stay_id, abbreviation))) +
  geom_line(linewidth = 1) +
```

```
geom_point(size = 2) +


facet_grid(rows = vars(abbreviation), cols = vars(stay_id), scales = 'free') +

scale_x_datetime(
  breaks = seq(
    floor_date(min(chartevents_viz$charttime, na.rm = TRUE), unit = '3 hours'),
    ceiling_date(max(chartevents_viz$charttime, na.rm = TRUE), unit = '3 hours'),
    by = '3 hours'
  ),
  labels = function(x) {
    labels <- format(x, '%b %d %H:%M')
    labels[seq(2, length(labels), 2)] <- ''
    return(labels)
  }
)  +

scale_color_manual(values = c(
  'HR' = '#E74C3C',
  'NBPm' = '#9A8700',
  'NBPs' = '#009E73',
  'RR' = '#009ADE',
  'Temperature F' = '#E377C2'
)) +

labs(
  title = paste0('Patient ', chartid, ' ICU stays - Vitals'),
  x = 'Calendar Time',
  y = 'Vital Value'
) +

theme_minimal() +
theme(
  strip.text.x = element_text(face = 'bold', size = 14, color = 'white'),
  strip.text.y.right = element_text(face = 'bold', size = 12, color = 'white'),
  strip.background = element_rect(fill = 'gray50', color = 'gray50'),
  strip.placement = 'outside',

  # Add a thin border around each facet panel
  panel.border = element_rect(color = "black", fill = NA, linewidth = 0.5),
```

```
    # Add space between facet panels
    panel.spacing = unit(0.2, 'cm'),

    panel.grid.major = element_line(color = "gray80"),
    panel.grid.minor = element_blank(),

    axis.text.x = element_text(angle = 0, hjust = 0.5, size = 10),

    # Remove 'Vital Sign' Legend
    legend.position = 'none'
)
```



Q2. ICU stays

`icustays.csv.gz` (https://mimic.mit.edu/docs/iv/modules/icu/icustays/) contains data about Intensive Care Units (ICU) stays. The first 10 lines are

```
zcat < ~/mimic/icu/icustays.csv.gz | head
```

```
subject_id,hadm_id,stay_id,first_careunit,last_careunit,intime,outtime,los
10000032,29079034,39553978,Medical Intensive Care Unit (MICU),Medical Intensive Care Unit (MI
10000690,25860671,37081114,Medical Intensive Care Unit (MICU),Medical Intensive Care Unit (MI
10000980,26913865,39765666,Medical Intensive Care Unit (MICU),Medical Intensive Care Unit (MI
10001217,24597018,37067082,Surgical Intensive Care Unit (SICU),Surgical Intensive Care Unit
10001217,27703517,34592300,Surgical Intensive Care Unit (SICU),Surgical Intensive Care Unit
10001725,25563031,31205490,Medical/Surgical Intensive Care Unit (MICU/SICU),Medical/Surgical
```

```
10001843,26133978,39698942,Medical/Surgical Intensive Care Unit (MICU/SICU),Medical/Surgical
10001884,26184834,37510196,Medical Intensive Care Unit (MICU),Medical Intensive Care Unit (MI
10002013,23581541,39060235,Cardiac Vascular Intensive Care Unit (CVICU),Cardiac Vascular Inte
```

Q2.1 Ingestion

Import `icustays.csv.gz` as a tibble `icustays_tble`. **Solution**: We will first import the data, and then utilize `mutate` to specify our data types. Once again, we want to ensure that timezone is `UTC` and that all data types are properly stored. We can use the `str` command to verify this periodically,

```
# Ensure correct data types
icustays_tble <- read_csv('~/mimic/icu/icustays.csv.gz')
```

```
Rows: 94458 Columns: 8
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (2): first_careunit, last_careunit
dbl  (4): subject_id, hadm_id, stay_id, los
dttm (2): intime, outtime

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.
```

```
icustays_tble <- icustays_tble %>%
  mutate(
    subject_id = as.integer(subject_id),
    hadm_id = as.integer(hadm_id),
    stay_id = as.integer(stay_id),
    intime = with_tz(intime,
                     'UTC'),
    outtime = with_tz(outtime,
                      'UTC')
  )
```

Q2.2 Summary and visualization

How many unique `subject_id`? Can a `subject_id` have multiple ICU stays? Summarize the number of ICU stays per `subject_id` by graphs. **Solution**: First, we will see how many unique values of `subject_id` there are. We observe that there are 65,366 unique instances of

subject id. To check if a particular value for `subject_id` can correspond to more than one icu stay, characterized by 'stay_id'. We will do this by creating a summary table

```
# Number of unique subject id
print(n_distinct(icustays_tble$subject_id))
```

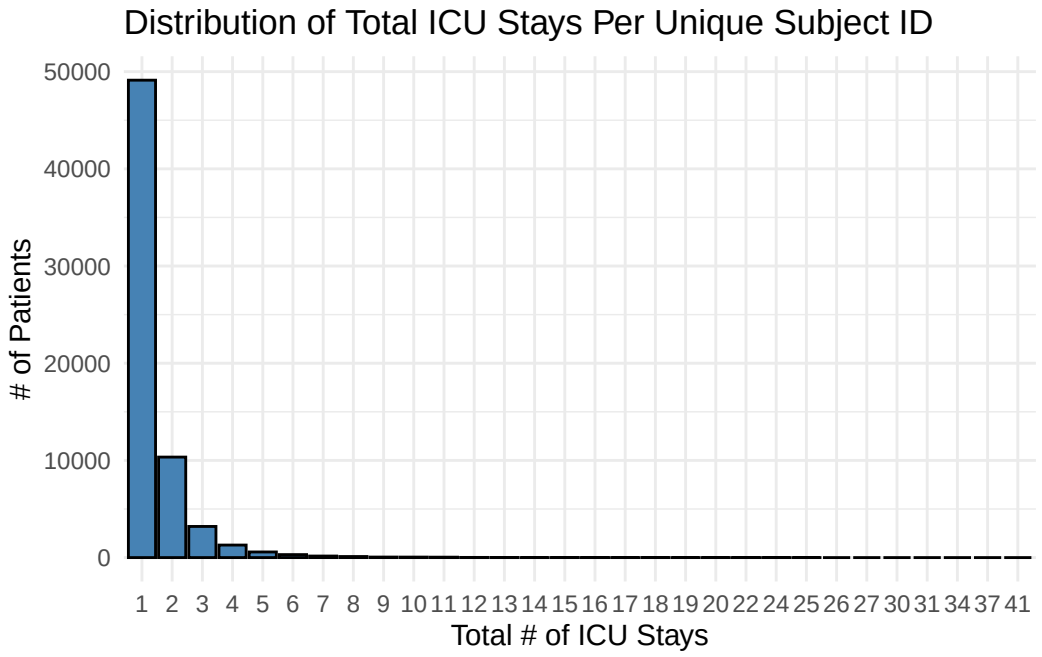
```
[1] 65366
```

```
# Get number of unique stay_id for each subject_id
icustays_by_subjid <- icustays_tble %>%
  group_by(subject_id) %>%
  summarise(
    num_stays = n_distinct(stay_id)) %>%
  count(num_stays,
        name = 'num_patients') %>%
  arrange(num_stays)

print(icustays_by_subjid)
```

```
# A tibble: 30 x 2
   num_stays num_patients
       <int>        <int>
 1         1        49124
 2         2        10341
 3         3         3206
 4         4         1290
 5         5          580
 6         6          308
 7         7          158
 8         8          110
 9         9           55
10        10           53
# i 20 more rows
```

```
# Visualize number of icu stays per unique subject id
ggplot(icustays_by_subjid,
       aes(x = factor(num_stays),
           y = num_patients)) +
  geom_bar(stat = 'identity',
           fill = 'steelblue',
           color = 'black') +
```

```
  labs(
    title = 'Distribution of Total ICU Stays Per Unique Subject ID',
    x = 'Total # of ICU Stays',
    y = '# of Patients'
  ) +
  theme_minimal()
```



Q3. `admissions` data

Information of the patients admitted into hospital is available in `admissions.csv.gz`. See https://mimic.mit.edu/docs/iv/modules/hosp/admissions/ for details of each field in this file. The first 10 lines are

```
zcat < ~/mimic/hosp/admissions.csv.gz | head
```

```
subject_id,hadm_id,admittime,dischtime,deathtime,admission_type,admit_provider_id,admission_l
10000032,22595853,2180-05-06 22:23:00,2180-05-07 17:15:00,,URGENT,P49AFC,TRANSFER FROM HOSPIT
10000032,22841357,2180-06-26 18:27:00,2180-06-27 18:49:00,,EW EMER.,P784FA,EMERGENCY ROOM,HOM
10000032,25742920,2180-08-05 23:44:00,2180-08-07 17:50:00,,EW EMER.,P19UTS,EMERGENCY ROOM,HOS
10000032,29079034,2180-07-23 12:35:00,2180-07-25 17:55:00,,EW EMER.,P06OTX,EMERGENCY ROOM,HOM
10000068,25022803,2160-03-03 23:16:00,2160-03-04 06:26:00,,EU OBSERVATION,P39NWO,EMERGENCY RO
```

```
10000084,23052089,2160-11-21 01:56:00,2160-11-25 14:52:00,,EW EMER.,P42H7G,WALK-IN/SELF REFE
10000084,29888819,2160-12-28 05:11:00,2160-12-28 16:07:00,,EU OBSERVATION,P35NE4,PHYSICIAN R
10000108,27250926,2163-09-27 23:17:00,2163-09-28 09:04:00,,EU OBSERVATION,P40JML,EMERGENCY R
10000117,22927623,2181-11-15 02:05:00,2181-11-15 14:52:00,,EU OBSERVATION,P47EY8,EMERGENCY R
```

Q3.1 Ingestion

Import `admissions.csv.gz` as a tibble `admissions_tble`. **Solution**: We will import the data and then ensure that all data types are properly formatted.

```
admissions_tble <- read_csv('~/mimic/hosp/admissions.csv.gz')
```

```
Rows: 546028 Columns: 16
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (8): admission_type, admit_provider_id, admission_location, discharge_l...
dbl  (3): subject_id, hadm_id, hospital_expire_flag
dttm (5): admittime, dischtime, deathtime, edregtime, edouttime

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.
```

```
# Convert IDs to integers & adjust time zones
admissions_tble <- admissions_tble %>%
  mutate(
    subject_id = as.integer(subject_id),
    hadm_id = as.integer(hadm_id),
    admittime = with_tz(admittime,
                        'UTC'),
    dischtime = with_tz(dischtime,
                        'UTC'),
    deathtime = with_tz(deathtime,
                        'UTC'),
    edregtime = with_tz(edregtime,
                        'UTC'),
    edouttime = with_tz(edouttime,
                        'UTC')
  )
```

Q3.2 Summary and visualization

Summarize the following information by graphics and explain any patterns you see.

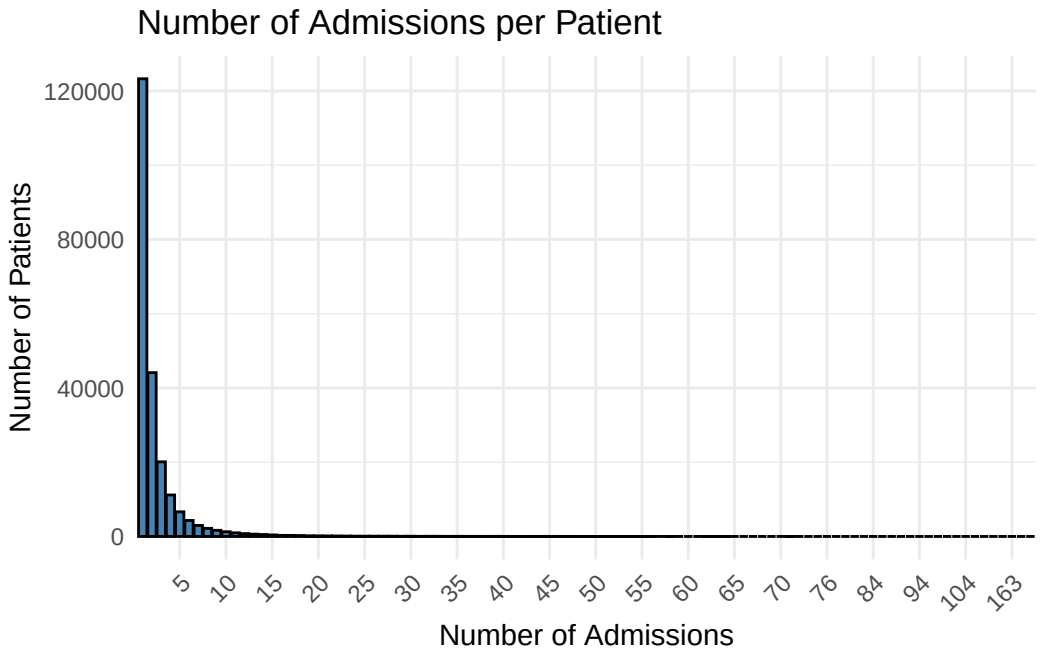
- number of admissions per patient
- admission hour (anything unusual?)
- admission minute (anything unusual?)
- length of hospital stay (from admission to discharge) (anything unusual?)

Solution:

Visualization 1: First, we will visualize the number of admissions per patient. We count the number of unique `hadm_id` values per each unique `subject_id`, and proceed to count the number of subject_ids corresponding to each observed number of admissions. We do this in a very similar manner to what was done in the previous exercise. Next, we create our visualization, which can be a bar chart. We choose the bin labels to occur for every 5th index, and we make them diagonal for readability. We notice something quite interesting, which is the presence of extreme right-skew in our data. The overwhelming majority of patients were only admitted once, with virtually all of the patients having less than ten admissions corresponding to their subject_id. However, we see that an extremely miniscule number of patients appear to have an extremely large number of hospital admissions, with some having upwards of 100 admissions.

```
# Get number of unique hadm_id for each subject_id
admissions_per_patient <- admissions_tble %>%
  group_by(subject_id) %>%
  summarise(
    num_admissions = n_distinct(hadm_id)) %>%
  count(num_admissions,
        name = 'num_patients') %>%
  arrange(num_admissions)

# Visualize number of admissions stays patient
ggplot(admissions_per_patient,
       aes(x = factor(num_admissions),
           y = num_patients)) +
  geom_bar(stat = 'identity',
           fill = 'steelblue',
           color = 'black') +

  # Reduce x-axis labels (show every few bins)
```

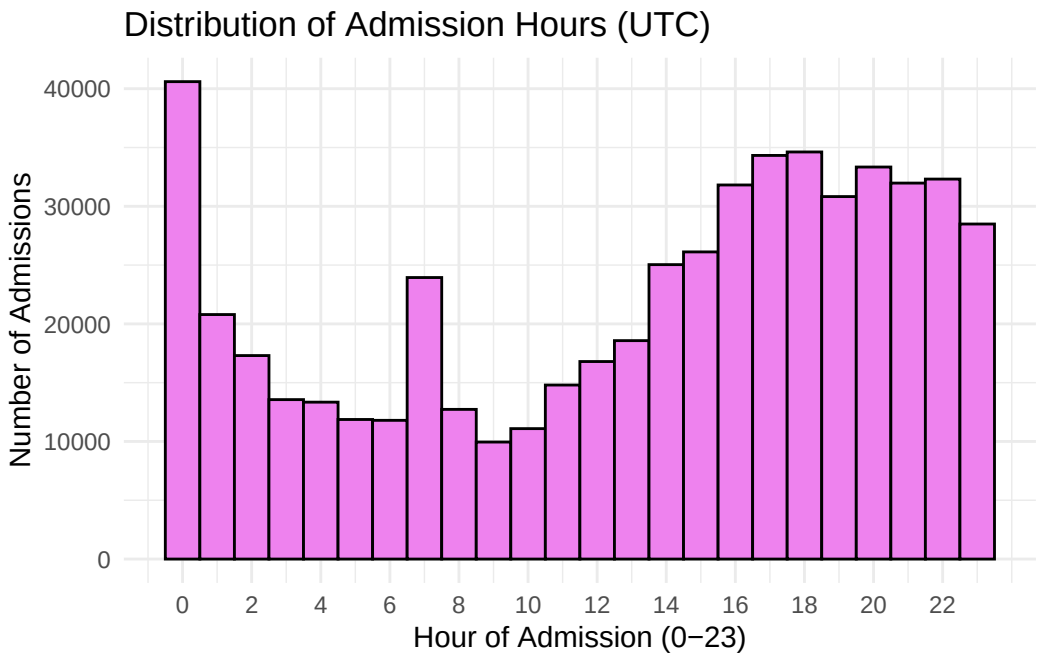
```r
  scale_x_discrete(breaks = function(x) x[seq(0,
                                              length(x),
                                              by = 5)]) +

  labs(
    title = 'Number of Admissions per Patient',
    x = 'Number of Admissions',
    y = 'Number of Patients'
  ) +
  theme_minimal() +
  theme(
    axis.text.x = element_text(angle = 45,
                               hjust = 1)  # Rotate for readability
  )
```



Visualization 2: Next, we will visualize the distribution of admission hour, i.e. at what time of day patients were admitted to the hospital. To do this, we must extract the hour from each `admittime` in the dataset, and plot a bar chart. We do notice something interesting when we plot this distribution; there is a considerable spike in admissions at `hour = 0`, which would correspond to `midnight UTC`. One possible explanation for this phenomenon may be the convention in the data collection process. Namely, it is possible that while some of these patients were indeed admitted around midnight, the data collection process may have involved using `hour = 0` as a default value for when admit time was left unspecified. I would suspect

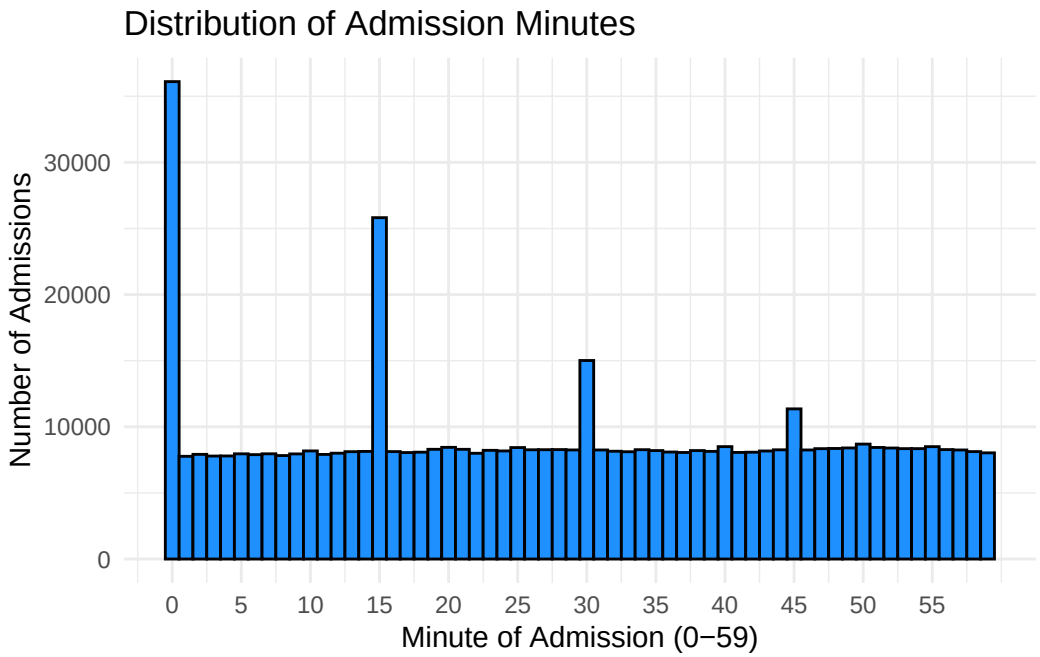
that many patients would share the exact admit time in that case (in terms of hour and minute), since it would serve as a `placeholder value`.

```
# Extract admission hour
admissions_tble <- admissions_tble %>%
  mutate(admit_hour = hour(admittime))  # Extract hour (0-23)

# Plot histogram of admission hours
ggplot(admissions_tble, aes(x = admit_hour)) +
  geom_histogram(binwidth = 1,
                 fill = 'violet',
                 color = 'black') +

  # Custom x-axis breaks (every 2 hours to reduce clutter)
  scale_x_continuous(breaks = seq(0, 23, by = 2)) +

  labs(
    title = 'Distribution of Admission Hours (UTC)',
    x = 'Hour of Admission (0-23)',
    y = 'Number of Admissions'
  ) +
  theme_minimal()
```



Visualization 3: We next plot the distribution of minute of admission, i.e. how far along in any given hour patients were admitted to the hospital. To do this, we must extract the minute value from our `admittime` timestamps and plot its distribution. Again, we observe something rather interesting in this plot. It appears that a large spike in minute of admission occurs at minute 0, which may be for the same reason we observed the spike at 0 in hour of admission. Namely, this may serve as a placeholder value for patients whose `admittime` is unknown. However, the spikes at 15-minute increments are also particularly interesting. We see that starting at 0, there is a spike in the number of admits in 15-minute increments (spikes at 0, 15, 30, and 45). This suggests that recorded times may be rounded to the nearest fifteen minutes, possibly for convenience. For instance, if a hospital admit came in two minutes ago and it is now 3:20, the data collector may still note the admission time as 3:15 despite it not being an exact metric. It also makes sense that 0 will be the mode of this distribution, since people may regress admit time back to 0 for any patients arriving within a certain vicinity of a new hour (e.g. patients arriving at 11:55 or 12:05 both having their admission times rounded to 12:00).

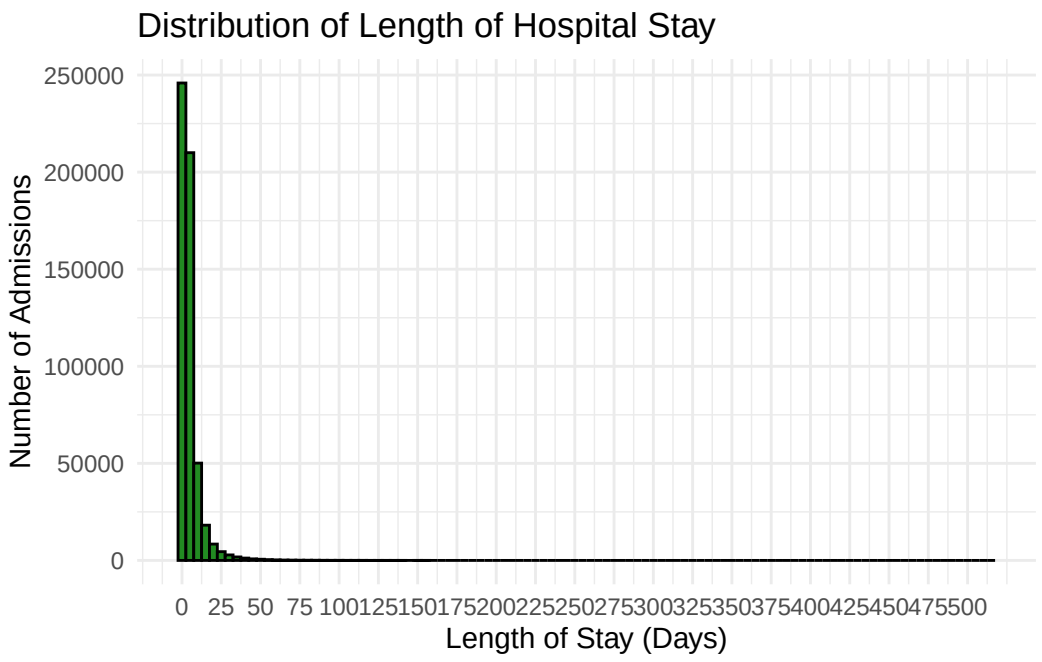
```
# Extract admission minute
admissions_tble <- admissions_tble %>%
  mutate(admit_minute = minute(admittime))  # Extract minute (0-59)

# Plot histogram of admission minutes
ggplot(admissions_tble, aes(x = admit_minute)) +
  geom_histogram(binwidth = 1, fill = "dodgerblue", color = "black") +

  # Custom x-axis breaks (every 5 minutes to reduce clutter)
  scale_x_continuous(breaks = seq(0, 59, by = 5)) +

  labs(
    title = 'Distribution of Admission Minutes',
    x = 'Minute of Admission (0-59)',
    y = 'Number of Admissions'
  ) +
  theme_minimal()
```



Visualization 4: Lastly, we will plot the distribution of length of hospital stay, from admit to discharge. We will need to create a new variable `length_of_stay` that represents the difference between `dischtime` and `admittime` in days. Then, we can follow our similar procedure of plotting. We once again observe something very interesting, which is the presence of extreme right-skew in our plot. Once again, it appears that the vast majority of patients had a length of stay totaling around 1 day, and most left within a week of their stay. In fact, nearly all patients completed their stay within 25 days. The presence of right-skew is likely a byproduct of an extremely small minority of patients having extremely prolonged stays in the hospital. Some patients may have their treatment regime be entirely inpatient and charted along the span of months and years rather than days. Because this is not the situation for a vast majority of patients, we see the extremely large right-tail present in our data.

```
# Compute length_of_stay in days
admissions_tble <- admissions_tble %>%
  mutate(length_of_stay = as.numeric(difftime(dischtime,
                                              admittime,
                                              units = 'days')))

# Compute length_of_stay in days
admissions_tble <- admissions_tble %>%
  mutate(length_of_stay = as.numeric(difftime(dischtime,
                                              admittime,
                                              units = 'days')))
```

```r
# Plot histogram of Length of Stay
ggplot(admissions_tble,
       aes(x = length_of_stay)) +
  geom_histogram(binwidth = 5,
                 fill = 'forestgreen',
                 color = 'black') +

  scale_x_continuous(breaks = seq(0,
                                  ceiling(max(admissions_tble$length_of_stay,
                                              na.rm = TRUE)),
                                  by = 25)) +

  labs(
    title = 'Distribution of Length of Hospital Stay',
    x = 'Length of Stay (Days)',
    y = 'Number of Admissions'
  ) +
  theme_minimal()
```



According to the MIMIC-IV documentation,

> All dates in the database have been shifted to protect patient confidentiality. Dates will be internally consistent for the same patient, but randomly distributed in the

future. Dates of birth which occur in the present time are not true dates of birth. Furthermore, dates of birth which occur before the year 1900 occur if the patient is older than 89. In these cases, the patient's age at their first admission has been fixed to 300.

Q4. `patients` data

Patient information is available in `patients.csv.gz`. See https://mimic.mit.edu/docs/iv/modules/hosp/patients/ for details of each field in this file. The first 10 lines are

```
zcat < ~/mimic/hosp/patients.csv.gz | head
```

```
subject_id,gender,anchor_age,anchor_year,anchor_year_group,dod
10000032,F,52,2180,2014 - 2016,2180-09-09
10000048,F,23,2126,2008 - 2010,
10000058,F,33,2168,2020 - 2022,
10000068,F,19,2160,2008 - 2010,
10000084,M,72,2160,2017 - 2019,2161-02-13
10000102,F,27,2136,2008 - 2010,
10000108,M,25,2163,2014 - 2016,
10000115,M,24,2154,2017 - 2019,
10000117,F,48,2174,2008 - 2010,
```

Q4.1 Ingestion

Import `patients.csv.gz` (https://mimic.mit.edu/docs/iv/modules/hosp/patients/) as a tibble `patients_tble`. **Solution**: We will first import the data. Next, we will make sure that all data types are properly formatted. This was done in the console using `str`.

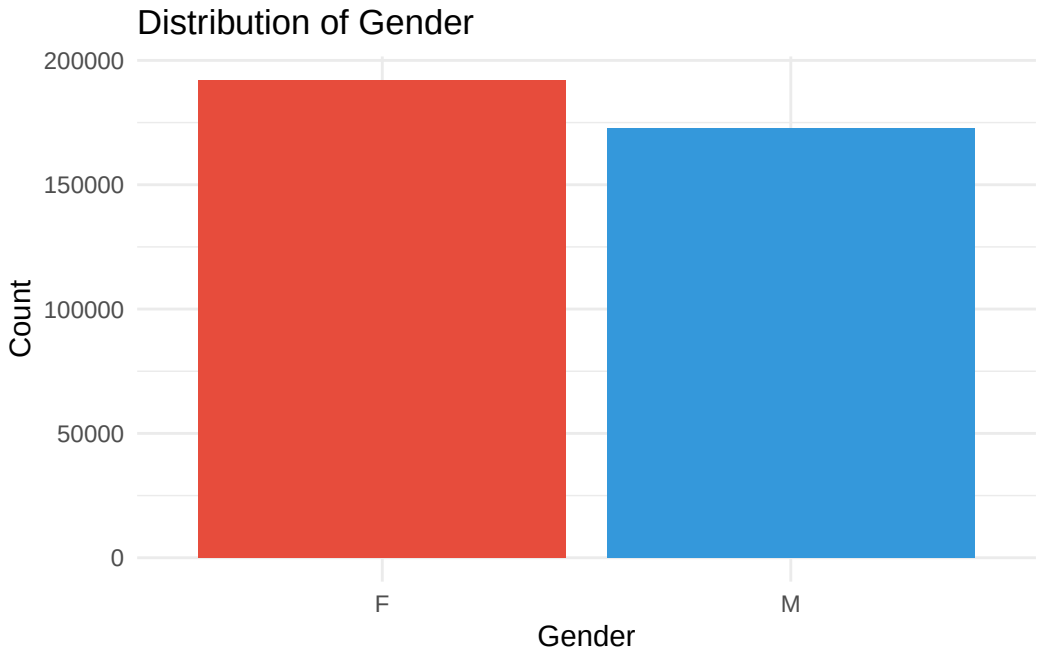
```
patients_tble <- read_csv('~/mimic/hosp/patients.csv.gz')
```

```
Rows: 364627 Columns: 6
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (2): gender, anchor_year_group
dbl  (3): subject_id, anchor_age, anchor_year
date (1): dod

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.
```

Q4.2 Summary and visualization

Summarize variables `gender` and `anchor_age` by graphics, and explain any patterns you see. **Solution**: We proceed to visualize the variables of interest. We will use a bar chart to visualize the distribution of `gender`, and a histogram to visualize the distribution of `anchor_age`. First, we plot the distribution of patient gender:

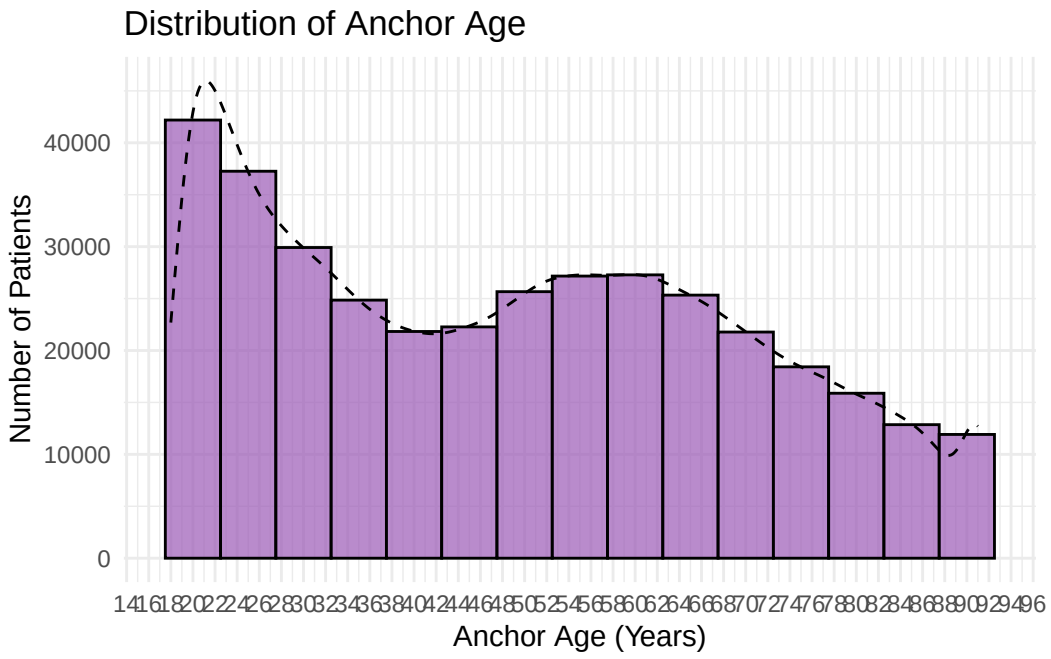
```
ggplot(patients_tble,
       aes(x = gender,
           fill = gender)) +
  geom_bar() +
  scale_fill_manual(values = c('F' = '#E74C3C',
                               'M' = '#3498DB')) +
  labs(
    title = 'Distribution of Gender',
    x = 'Gender',
    y = 'Count'
  ) +
  theme_minimal() +
  theme(legend.position = 'none')
```



Next, we plot the distribution of patient age as a histogram with an overlaid, scaled density curve:

```
ggplot(patients_tble,
       aes(x = anchor_age)) +
  geom_histogram(binwidth = 5,
                 fill = '#9B59B6',
                 color = 'black',
                 alpha = 0.7) +
  geom_density(aes(y = ..count.. * 5),
               color = 'black',
               linetype = 'dashed') +  # Density curve scaled to match histogram
  scale_x_continuous(breaks = seq(0,
                                  100,
                                  by = 2)) +  # Tick marks every 10 years
  labs(
    title = 'Distribution of Anchor Age',
    x = 'Anchor Age (Years)',
    y = 'Number of Patients'
  ) +
  theme_minimal()
```

```
Warning: The dot-dot notation (`..count..`) was deprecated in ggplot2 3.4.0.
i Please use `after_stat(count)` instead.
```



We see that there are slightly more female patients than male patients, and that the distribution of anchor age is unimodal around the late teen years. There is another small spike visible in the mid-50. It is worth noting that the ages, birth years, and anchor years are all separate quantities, and that anchor_age is merely the patient's age in the anchor_year. Hence, it is difficult to interpret the variable `anchor_age` without the added context of their `anchor_year_group`, and their `anchor_year`. Essentially, this variable corresponds to the age of the patient at the time of their shifted year, which varies among the patients.
Q5. Lab results

`labevents.csv.gz` (https://mimic.mit.edu/docs/iv/modules/hosp/labevents/) contains all laboratory measurements for patients. The first 10 lines are

```
zcat < ~/mimic/hosp/labevents.csv.gz | head
```

```
labevent_id,subject_id,hadm_id,specimen_id,itemid,order_provider_id,charttime,storetime,valu
1,10000032,,2704548,50931,P69FQC,2180-03-23 11:51:00,2180-03-23 15:56:00,___,95,mg/dL,70,100
2,10000032,,36092842,51071,P69FQC,2180-03-23 11:51:00,2180-03-23 16:00:00,NEG,,,,,ROUTINE,
3,10000032,,36092842,51074,P69FQC,2180-03-23 11:51:00,2180-03-23 16:00:00,NEG,,,,,ROUTINE,
4,10000032,,36092842,51075,P69FQC,2180-03-23 11:51:00,2180-03-23 16:00:00,NEG,,,,,ROUTINE,"
5,10000032,,36092842,51079,P69FQC,2180-03-23 11:51:00,2180-03-23 16:00:00,NEG,,,,,ROUTINE,
6,10000032,,36092842,51087,P69FQC,2180-03-23 11:51:00,,,,,,,ROUTINE,RANDOM.
7,10000032,,36092842,51089,P69FQC,2180-03-23 11:51:00,2180-03-23 16:15:00,,,,,,ROUTINE,PRESU
8,10000032,,36092842,51090,P69FQC,2180-03-23 11:51:00,2180-03-23 16:00:00,NEG,,,,,ROUTINE,M
9,10000032,,36092842,51092,P69FQC,2180-03-23 11:51:00,2180-03-23 16:00:00,NEG,,,,,ROUTINE,"(
```

`d_labitems.csv.gz` (https://mimic.mit.edu/docs/iv/modules/hosp/d_labitems/) is the dictionary of lab measurements.

```
zcat < ~/mimic/hosp/d_labitems.csv.gz | head
```

```
itemid,label,fluid,category
50801,Alveolar-arterial Gradient,Blood,Blood Gas
50802,Base Excess,Blood,Blood Gas
50803,"Calculated Bicarbonate, Whole Blood",Blood,Blood Gas
50804,Calculated Total CO2,Blood,Blood Gas
50805,Carboxyhemoglobin,Blood,Blood Gas
50806,"Chloride, Whole Blood",Blood,Blood Gas
50808,Free Calcium,Blood,Blood Gas
50809,Glucose,Blood,Blood Gas
50810,"Hematocrit, Calculated",Blood,Blood Gas
```

We are interested in the lab measurements of creatinine (50912), potassium (50971), sodium (50983), chloride (50902), bicarbonate (50882), hematocrit (51221), white blood cell count (51301), and glucose (50931). Retrieve a subset of `labevents.csv.gz` that only containing these items for the patients in `icustays_tble`. Further restrict to the last available measurement (by `storetime`) before the ICU stay. The final `labevents_tble` should have one row per ICU stay and columns for each lab measurement.

```
> labevents_tble
# A tibble: 88,086 × 10
   subject_id  stay_id bicarbonate chloride creatinine glucose potassium sodium hematocrit   wbc
        <dbl>    <dbl>       <dbl>    <dbl>      <dbl>   <dbl>     <dbl>  <dbl>       <dbl> <dbl>
 1   10000032 39553978          25       95        0.7     102       6.7    126        41.1   6.9
 2   10000690 37081114          26      100        1        85       4.8    137        36.1   7.1
 3   10000980 39765666          21      109        2.3      89       3.9    144        27.3   5.3
 4   10001217 34592300          30      104        0.5      87       4.1    142        37.4   5.4
 5   10001217 37067082          22      108        0.6     112       4.2    142        38.1  15.7
 6   10001725 31205490          NA       98       NA       NA        4.1    139        NA    NA
 7   10001843 39698942          28       97        1.3     131       3.9    138        31.4  10.4
 8   10001884 37510196          30       88        1.1     141       4.5    130        39.7  12.2
 9   10002013 39060235          24      102        0.9     288       3.5    137        34.9   7.2
10   10002114 34672098          18       NA        3.1      95       6.5    125        34.3  16.8
# ℹ 88,076 more rows
# ℹ Use `print(n = ...)` to see more rows
```

Hint: Use the Parquet format you generated in Homework 2. For reproducibility, make `labevents_pq` folder available at the current working directory `hw3`, for example, by a symbolic link.

Solution: We first load in our dataset using the`parquet` format we generated in Homework 2. We have made it available in our working directory `hw3` via a symbolic link we created at the time of completing Q1. We filter for our metrics of interest, arrange the dataset, and aggregate the values. We also need to filter for `subject_id`'s that are present in `icustays_tble`.

```
lab_items <- c(50912,
               50971,
               50983,
               50902,
               50882,
               51221,
               51301,
               50931)
labevents_tble <- open_dataset(
  sources = 'labevents_pq',
  format = 'parquet'
) %>%
  to_duckdb() %>%
  select(subject_id,
```

```
        itemid,
        storetime,
        valuenum) %>%
  filter(itemid %in% lab_items) %>%
  left_join(
    select(icustays_tble,
           subject_id,
           stay_id,
           intime),
    by = c('subject_id'),
    copy = TRUE
) %>%
  filter(storetime < intime) %>%
  group_by(subject_id,
           stay_id,
           itemid) %>%
  slice_max(storetime,
            n = 1) %>%
  select(-storetime,
         -intime) %>%
  ungroup() %>%
  pivot_wider(names_from = itemid,
              values_from = valuenum) %>%
  collect() %>%
  arrange(subject_id, stay_id) %>%
  relocate(subject_id, stay_id, sort(names(.))) %>%
  rename(
    creatinine = `50912`,
    potassium = `50971`,
    sodium = `50983`,
    chloride = `50902`,
    bicarbonate = `50882`,
    hematocrit = `51221`,
    wbc_count = `51301`,
    glucose = `50931`
  ) %>%

  print(width = Inf)
```

```
# A tibble: 88,086 x 10
   subject_id  stay_id bicarbonate chloride creatinine glucose potassium sodium
        <dbl>    <int>       <dbl>    <dbl>      <dbl>   <dbl>     <dbl>  <dbl>
```

```
 1   10000032 39553978          25        95        0.7      102        6.7    126
 2   10000690 37081114          26       100        1         85        4.8    137
 3   10000980 39765666          21       109        2.3       89        3.9    144
 4   10001217 34592300          30       104        0.5       87        4.1    142
 5   10001217 37067082          22       108        0.6      112        4.2    142
 6   10001725 31205490          NA        98       NA         NA        4.1    139
 7   10001843 39698942          28        97        1.3      131        3.9    138
 8   10001884 37510196          30        88        1.1      141        4.5    130
 9   10002013 39060235          24       102        0.9      288        3.5    137
10   10002114 34672098          18        NA        3.1       95        6.5    125
   hematocrit wbc_count
        <dbl>     <dbl>
 1       41.1       6.9
 2       36.1       7.1
 3       27.3       5.3
 4       37.4       5.4
 5       38.1      15.7
 6       NA        NA
 7       31.4      10.4
 8       39.7      12.2
 9       34.9       7.2
10       34.3      16.8
# i 88,076 more rows
```

Q6. Vitals from charted events

`chartevents.csv.gz` (https://mimic.mit.edu/docs/iv/modules/icu/chartevents/) contains all the charted data available for a patient. During their ICU stay, the primary repository of a patient's information is their electronic chart. The `itemid` variable indicates a single measurement type in the database. The `value` variable is the value measured for `itemid`. The first 10 lines of `chartevents.csv.gz` are

```
zcat < ~/mimic/icu/chartevents.csv.gz | head
```

```
subject_id,hadm_id,stay_id,caregiver_id,charttime,storetime,itemid,value,valuenum,valueuom,wa
10000032,29079034,39553978,18704,2180-07-23 12:36:00,2180-07-23 14:45:00,226512,39.4,39.4,kg
10000032,29079034,39553978,18704,2180-07-23 12:36:00,2180-07-23 14:45:00,226707,60,60,Inch,0
10000032,29079034,39553978,18704,2180-07-23 12:36:00,2180-07-23 14:45:00,226730,152,152,cm,0
10000032,29079034,39553978,18704,2180-07-23 14:00:00,2180-07-23 14:18:00,220048,SR (Sinus Rhy
10000032,29079034,39553978,18704,2180-07-23 14:00:00,2180-07-23 14:18:00,224642,Oral,,,0
10000032,29079034,39553978,18704,2180-07-23 14:00:00,2180-07-23 14:18:00,224650,None,,,0
```

```
10000032,29079034,39553978,18704,2180-07-23 14:00:00,2180-07-23 14:20:00,223761,98.7,98.7,°F
10000032,29079034,39553978,18704,2180-07-23 14:11:00,2180-07-23 14:17:00,220179,84,84,mmHg,0
10000032,29079034,39553978,18704,2180-07-23 14:11:00,2180-07-23 14:17:00,220180,48,48,mmHg,0
```

`d_items.csv.gz` (https://mimic.mit.edu/docs/iv/modules/icu/d_items/) is the dictionary for the `itemid` in `chartevents.csv.gz`.

```
zcat < ~/mimic/icu/d_items.csv.gz | head
```

```
itemid,label,abbreviation,linksto,category,unitname,param_type,lownormalvalue,highnormalvalue
220001,Problem List,Problem List,chartevents,General,,Text,,
220003,ICU Admission date,ICU Admission date,datetimeevents,ADT,,Date and time,,
220045,Heart Rate,HR,chartevents,Routine Vital Signs,bpm,Numeric,,
220046,Heart rate Alarm - High,HR Alarm - High,chartevents,Alarms,bpm,Numeric,,
220047,Heart Rate Alarm - Low,HR Alarm - Low,chartevents,Alarms,bpm,Numeric,,
220048,Heart Rhythm,Heart Rhythm,chartevents,Routine Vital Signs,,Text,,
220050,Arterial Blood Pressure systolic,ABPs,chartevents,Routine Vital Signs,mmHg,Numeric,90
220051,Arterial Blood Pressure diastolic,ABPd,chartevents,Routine Vital Signs,mmHg,Numeric,60
220052,Arterial Blood Pressure mean,ABPm,chartevents,Routine Vital Signs,mmHg,Numeric,,
```

We are interested in the vitals for ICU patients: heart rate (220045), systolic non-invasive blood pressure (220179), diastolic non-invasive blood pressure (220180), body temperature in Fahrenheit (223761), and respiratory rate (220210). Retrieve a subset of `chartevents.csv.gz` only containing these items for the patients in `icustays_tble`. Further restrict to the first vital measurement (by `storetime`) within the ICU stay. The final `chartevents_tble` should have one row per ICU stay and columns for each vital measurement.

```
> chartevents_tble
# A tibble: 94,363 × 7
   subject_id  stay_id heart_rate non_invasive_blood_pressure_diastolic non_invasive_blood_pressure_systolic respiratory_rate temperature_fahrenheit
        <dbl>    <dbl>      <dbl>                                 <dbl>                                <dbl>            <dbl>                 <dbl>
 1   10000032 39553978         91                                  48                                   84               24                  98.7
 2   10000690 37081114         78                                  56.5                                106               24.3                97.7
 3   10000980 39765666         76                                 102                                  154               23.5                98
 4   10001217 34592300         79.3                                93.3                                156               14                  97.6
 5   10001217 37067082         86                                  90                                  151               18                  98.5
 6   10001725 31205490         86                                  56                                   73               19                  97.7
 7   10001843 39698942        124.                                 78                                  110               16.5                97.9
 8   10001884 37510196         49                                  30.5                                174.              13                  98.1
 9   10002013 39060235         80                                  62                                   98.5             14                  97.2
10   10002114 34672098        110.                                 80                                  112               21                  97.9
# ℹ 94,353 more rows
# ℹ Use `print(n = ...)` to see more rows
```

Hint: Use the Parquet format you generated in Homework 2. For reproducibility, make `chartevents_pq` folder available at the current working directory, for example, by a symbolic link. **Solution**:

```r
# Define relevant item IDs for vitals and desired final labels.
vitals_items <- c(220045,
                  220179,
                  220180,
                  223761,
                  220210)
final_names <- c(
  '220045' = 'Heart Rate',
  '220179' = 'Noninvasive BP Systolic',
  '220180' = 'Noninvasive BP Diastolic',
  '223761' = 'Temperature (F)',
  '220210' = 'Respiratory Rate'
)

# Process chartevents:
chartevents_tble <- open_dataset('chartevents_pq',
                                 format = 'parquet') %>%
  to_duckdb() %>%
  select(subject_id,
         stay_id,
         itemid,
         storetime,
         valuenum) %>%
  filter(itemid %in% vitals_items) %>%
  inner_join(
    select(icustays_tble,
           subject_id,
           stay_id,
           intime),
    by = c('subject_id',
           'stay_id'),
    copy = TRUE
  ) %>%
  # Keep only measurements after ICU admission
  filter(storetime >= intime) %>%
  # For each ICU stay and vital, select first measurement (using storetime)
  group_by(subject_id,
           stay_id,
           itemid) %>%
  slice_min(storetime,
            n = 1,
            with_ties = TRUE) %>%
```

```
  summarise(avg_valuenum = mean(valuenum,
                                na.rm = TRUE),
            .groups = 'drop') %>%
  # Pivot so each ICU stay is one row, with a column per itemid.
  pivot_wider(names_from = itemid,
              values_from = avg_valuenum,
              names_prefix = 'lab_') %>%
  collect() %>%
  arrange(subject_id,
          stay_id) %>%
  relocate(subject_id,
           stay_id,
           sort(names(.)))

# Rename pivoted columns using mapping in final_names.
for(old in names(final_names)) {
  old_name <- paste0('lab_',
                     old)
  new_name <- final_names[old]
  if(old_name %in% colnames(chartevents_tble)) {
    colnames(
      chartevents_tble)[colnames(chartevents_tble) == old_name] <- new_name
  }
}

# Round numeric columns (all except subject_id and stay_id) to 2 decimal places.
chartevents_tble <- chartevents_tble %>%
  mutate(across(-c(subject_id,
                   stay_id),
                ~ round(.x,
                        2)))

# Remove rows that have all NA for lab measurements.
chartevents_tble <- chartevents_tble %>%
  filter(if_any(-c(subject_id,
                   stay_id),
                ~ !is.na(.)))

# Display final result and row count
print(chartevents_tble,
      width = Inf)
```

```
# A tibble: 94,438 x 7
   subject_id  stay_id `Heart Rate` `Noninvasive BP Systolic`
        <dbl>    <dbl>        <dbl>                     <dbl>
 1   10000032 39553978         91                        84
 2   10000690 37081114         78                       106
 3   10000980 39765666         76                       154
 4   10001217 34592300         79.3                     156
 5   10001217 37067082         86                       151
 6   10001725 31205490         86                        73
 7   10001843 39698942        124.                      110
 8   10001884 37510196         49                       174.
 9   10002013 39060235         80                        98.5
10   10002114 34672098        110.                      112
   `Noninvasive BP Diastolic` `Respiratory Rate` `Temperature (F)`
                        <dbl>              <dbl>             <dbl>
 1                       48                 24                98.7
 2                       56.5               24.3              97.7
 3                      102                 23.5              98
 4                       93.3               14                97.6
 5                       90                 18                98.5
 6                       56                 19                97.7
 7                       78                 16.5              97.9
 8                       30.5               13                98.1
 9                       62                 14                97.2
10                       80                 21                97.9
# i 94,428 more rows
```

```
print(nrow(chartevents_tble))
```

```
[1] 94438
```

Q7. Putting things together

Let us create a tibble `mimic_icu_cohort` for all ICU stays, where rows are all ICU stays of adults (age at `intime` >= 18) and columns contain at least following variables

- all variables in `icustays_tble`
- all variables in `admissions_tble`
- all variables in `patients_tble`

- the last lab measurements before the ICU stay in `labevents_tble`
- the first vital measurements during the ICU stay in `chartevents_tble`

The final `mimic_icu_cohort` should have one row per ICU stay and columns for each variable.

```
> mimic_icu_cohort
# A tibble: 94,458 × 41
   subject_id  hadm_id  stay_id first_careunit        last_careunit intime              outtime              los admittime           dischtime           deathtime
        <dbl>    <dbl>    <dbl> <chr>                 <chr>         <dttm>              <dttm>             <dbl> <dttm>              <dttm>              <dttm>
 1   10000032 29079034 39553978 Medical Intensive Car… Medical Inte… 2180-07-23 14:00:00 2180-07-23 23:50:47 0.410 2180-07-23 12:35:00 2180-07-25 17:55:00 NA
 2   10000690 25860671 37081114 Medical Intensive Car… Medical Inte… 2150-11-02 19:37:00 2150-11-06 17:03:17 3.89  2150-11-02 18:02:00 2150-11-12 13:45:00 NA
 3   10000980 26913865 39765666 Medical Intensive Car… Medical Inte… 2189-06-27 08:42:00 2189-06-27 20:38:27 0.498 2189-06-27 07:38:00 2189-07-03 03:00:00 NA
 4   10001217 24597018 37067082 Surgical Intensive Ca… Surgical Int… 2157-11-20 19:18:02 2157-11-21 22:08:00 1.12  2157-11-18 22:56:00 2157-11-25 18:00:00 NA
 5   10001217 27703517 34592300 Surgical Intensive Ca… Surgical Int… 2157-12-19 15:42:24 2157-12-20 14:27:41 0.948 2157-12-18 16:58:00 2157-12-24 14:55:00 NA
 6   10001725 25563031 31205490 Medical/Surgical Inte… Medical/Surg… 2110-04-11 15:52:22 2110-04-12 23:59:56 1.34  2110-04-11 15:08:00 2110-04-14 15:00:00 NA
 7   10001843 26133978 39698942 Medical/Surgical Inte… Medical/Surg… 2134-12-05 18:50:03 2134-12-06 14:38:26 0.825 2134-12-05 00:10:00 2134-12-06 12:54:00 2134-12-06 12:54:00
 8   10001884 26184834 37510196 Medical Intensive Car… Medical Inte… 2131-01-11 04:20:05 2131-01-20 08:27:30 9.17  2131-01-07 20:39:00 2131-01-20 05:15:00 2131-01-20 05:15:00
 9   10002013 23581541 39060235 Cardiac Vascular Inte… Cardiac Vasc… 2160-05-18 10:00:53 2160-05-19 17:33:33 1.31  2160-05-18 07:45:00 2160-05-23 13:30:00 NA
10   10002114 27793700 34672098 Coronary Care Unit (C… Coronary Car… 2162-02-17 23:30:00 2162-02-20 21:16:27 2.91  2162-02-17 22:32:00 2162-03-04 15:16:00 NA
# ℹ 94,448 more rows
# ℹ 30 more variables: admission_type <chr>, admit_provider_id <chr>, admission_location <chr>, discharge_location <chr>, insurance <chr>, language <chr>,
#   marital_status <chr>, race <chr>, edregtime <dttm>, edouttime <dttm>, hospital_expire_flag <dbl>, gender <chr>, anchor_age <dbl>, anchor_year <dbl>,
#   anchor_year_group <chr>, dod <date>, bicarbonate <dbl>, chloride <dbl>, creatinine <dbl>, glucose <dbl>, potassium <dbl>, sodium <dbl>, hematocrit <dbl>, wbc <dbl>,
#   heart_rate <dbl>, non_invasive_blood_pressure_systolic <dbl>, non_invasive_blood_pressure_diastolic <dbl>, respiratory_rate <dbl>, temperature_fahrenheit <dbl>,
#   age_intime <dbl>
# ℹ Use `print(n = ...)` to see more rows
```

```r
mimic_icu_cohort <- icustays_tble %>%
  # Join admissions on subject_id and hadm_id (patients don't have stay_id)
  left_join(admissions_tble,
            by = c('subject_id',
                   'hadm_id')) %>%
  # Join patients on subject_id only
  left_join(patients_tble,
            by = 'subject_id') %>%
  # Join lab events by subject_id and stay_id
  left_join(labevents_tble,
            by = c('subject_id',
                   'stay_id')) %>%
  # Join vital events by subject_id and stay_id
  left_join(chartevents_tble,
            by = c('subject_id',
                   'stay_id')) %>%
  # Filter for adults
  filter(anchor_age >= 18)
head(mimic_icu_cohort)
```

```
# A tibble: 6 x 43
  subject_id  hadm_id  stay_id first_careunit  last_careunit intime
       <dbl>    <int>    <dbl> <chr>           <chr>         <dttm>
1   10000032 29079034 39553978 Medical Intens~ Medical Inte~ 2180-07-23 14:00:00
2   10000690 25860671 37081114 Medical Intens~ Medical Inte~ 2150-11-02 19:37:00
3   10000980 26913865 39765666 Medical Intens~ Medical Inte~ 2189-06-27 08:42:00
```

```
4   10001217 24597018 37067082 Surgical Inten~ Surgical Int~ 2157-11-20 19:18:02
5   10001217 27703517 34592300 Surgical Inten~ Surgical Int~ 2157-12-19 15:42:24
6   10001725 25563031 31205490 Medical/Surgic~ Medical/Surg~ 2110-04-11 15:52:22
# i 37 more variables: outtime <dttm>, los <dbl>, admittime <dttm>,
#   dischtime <dttm>, deathtime <dttm>, admission_type <chr>,
#   admit_provider_id <chr>, admission_location <chr>,
#   discharge_location <chr>, insurance <chr>, language <chr>,
#   marital_status <chr>, race <chr>, edregtime <dttm>, edouttime <dttm>,
#   hospital_expire_flag <dbl>, admit_hour <int>, admit_minute <int>,
#   length_of_stay <dbl>, gender <chr>, anchor_age <dbl>, ...
```

Q8. Exploratory data analysis (EDA)

Summarize the following information about the ICU stay cohort `mimic_icu_cohort` using appropriate numerics or graphs:

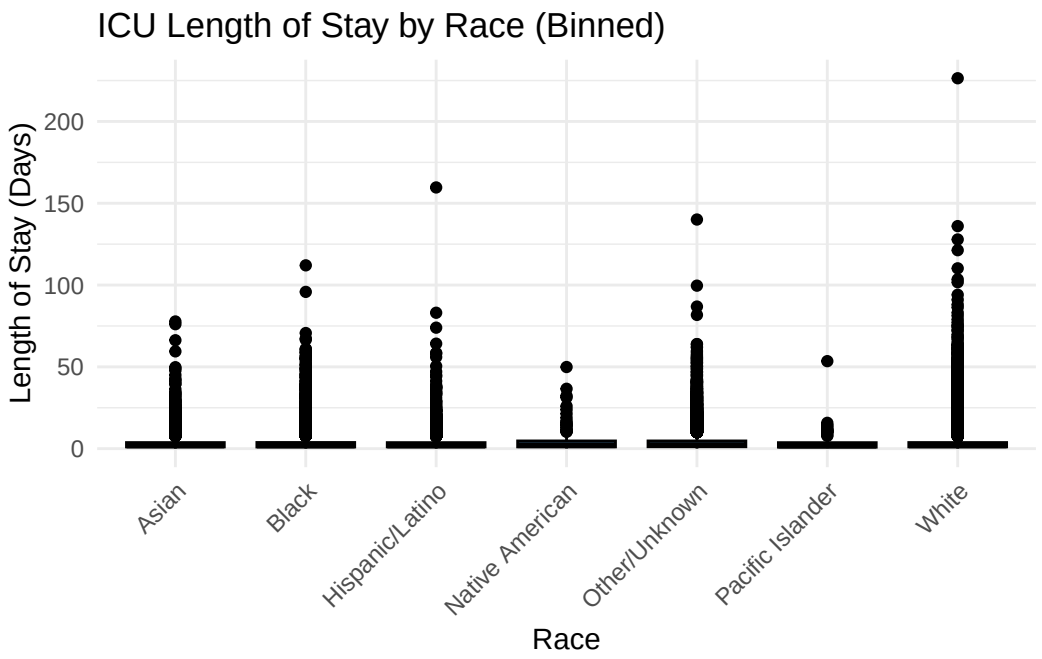
- Length of ICU stay `los` vs demographic variables (race, insurance, marital_status, gender, age at intime)
- Length of ICU stay `los` vs the last available lab measurements before ICU stay
- Length of ICU stay `los` vs the first vital measurements within the ICU stay
- Length of ICU stay `los` vs first ICU unit

Solution: First, we will visualize length of stay against demographic variables. We will begin with `race`. We observe that there are 33 different race categories with considerable overlap, so we will bin them together for intuitive visualization.

```
# los and race
mimic_icu_cohort <- mimic_icu_cohort %>%
  mutate(race_bin = case_when(
    race %in% c('WHITE',
                'WHITE - RUSSIAN',
                'WHITE - OTHER EUROPEAN',
                'WHITE - BRAZILIAN',
                'WHITE - EASTERN EUROPEAN') ~ 'White',
    race %in% c('BLACK/AFRICAN AMERICAN',
                'BLACK/CAPE VERDEAN',
                'BLACK/AFRICAN',
                'BLACK/CARIBBEAN ISLAND') ~ 'Black',
    race %in% c('HISPANIC/LATINO - SALVADORAN',
                'HISPANIC/LATINO - PUERTO RICAN',
                'HISPANIC/LATINO - GUATEMALAN',
```

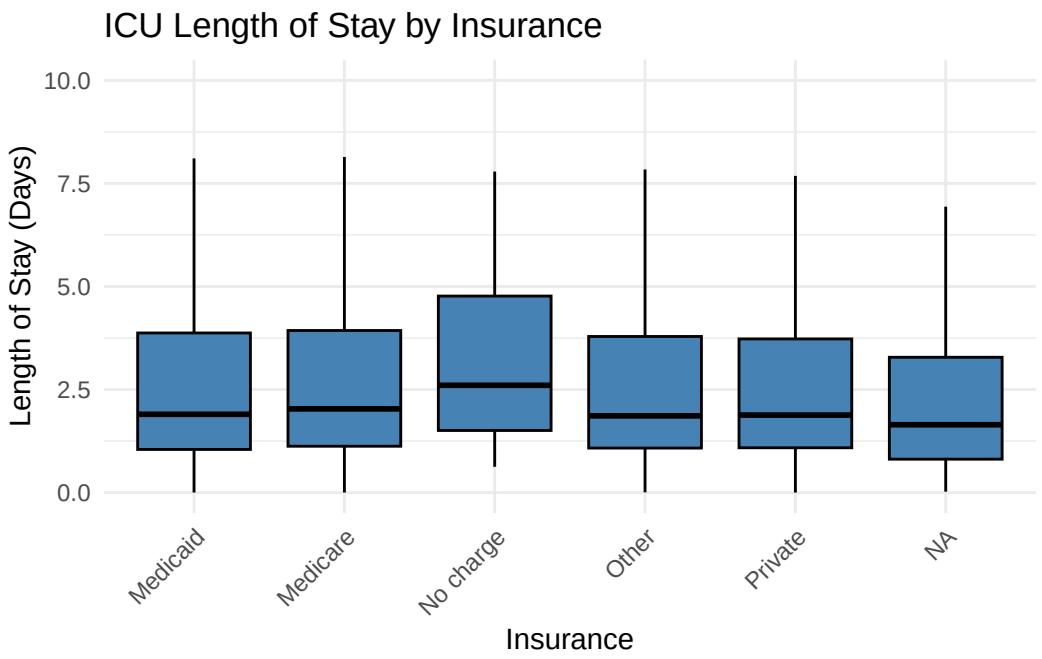
```
                'HISPANIC/LATINO - DOMINICAN',
                'HISPANIC OR LATINO',
                'HISPANIC/LATINO - CUBAN',
                'HISPANIC/LATINO - CENTRAL AMERICAN',
                'HISPANIC/LATINO - HONDURAN',
                'HISPANIC/LATINO - COLUMBIAN',
                'HISPANIC/LATINO - MEXICAN') ~ 'Hispanic/Latino',
    race %in% c('ASIAN',
                'ASIAN - SOUTH EAST ASIAN',
                'ASIAN - CHINESE',
                'ASIAN - KOREAN',
                'ASIAN - ASIAN INDIAN') ~ 'Asian',
    race %in% c('AMERICAN INDIAN/ALASKA NATIVE') ~ 'Native American',
    race %in% c(
      'NATIVE HAWAIIAN OR OTHER PACIFIC ISLANDER') ~ 'Pacific Islander',
    TRUE ~ 'Other/Unknown'
  ))

ggplot(mimic_icu_cohort, aes(x = race_bin, y = los)) +
  geom_boxplot(fill = 'steelblue', color = 'black') +
  labs(title = 'ICU Length of Stay by Race (Binned)',
       x = 'Race',
       y = 'Length of Stay (Days)') +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45,
                                   hjust = 1))
```

```
Warning: Removed 14 rows containing non-finite outside the scale range
(`stat_boxplot()`).
```



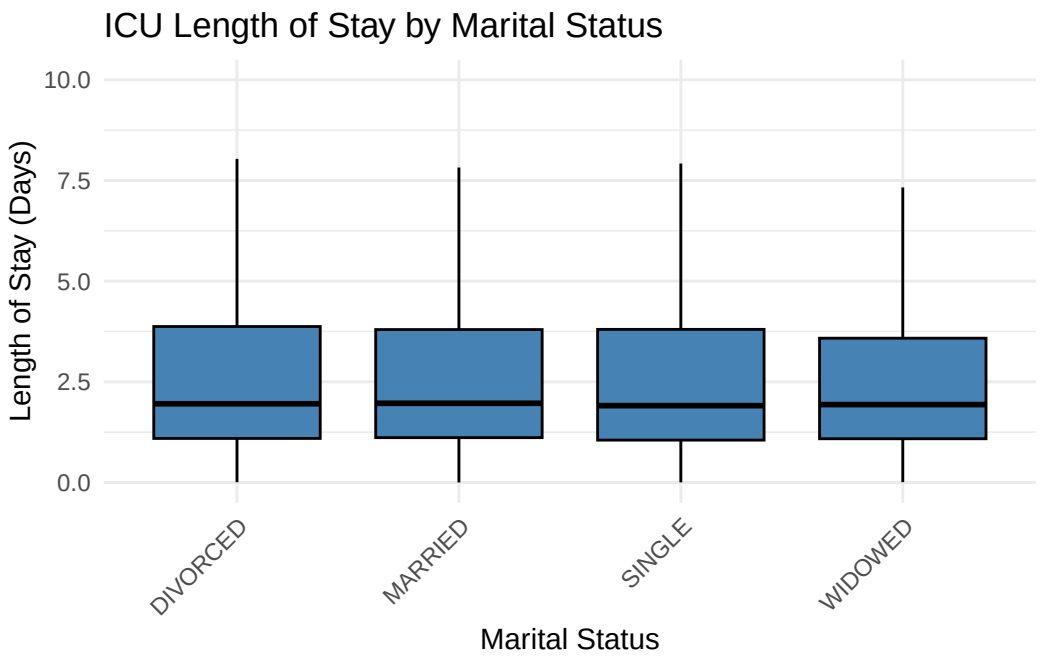
Next, we will plot the distribution of length of stay by insurance.

```
ggplot(mimic_icu_cohort %>% filter(!is.na(los)),
       aes(x = insurance,
           y = los)) +
  geom_boxplot(fill = 'steelblue',
               color = 'black',
               outlier.shape = NA) +
  labs(
    title = 'ICU Length of Stay by Insurance',
    x = 'Insurance',
    y = 'Length of Stay (Days)'
  ) +
  theme_minimal() +
  theme(
    axis.text.x = element_text(angle = 45,
                               hjust = 1)
  ) +
  coord_cartesian(ylim = c(0,
                           10))
```



Next, we will plot length of stay by marital status:

```
ggplot(mimic_icu_cohort %>% filter(!is.na(los),
                                   !is.na(marital_status)),
       aes(x = marital_status,
           y = los)) +
  geom_boxplot(fill = 'steelblue',
               color = 'black',
               outlier.shape = NA) +
  labs(
    title = 'ICU Length of Stay by Marital Status',
    x = 'Marital Status',
    y = 'Length of Stay (Days)'
  ) +
  theme_minimal() +
  theme(
    axis.text.x = element_text(angle = 45,
                               hjust = 1)
  ) +
  coord_cartesian(ylim = c(0, 10))
```



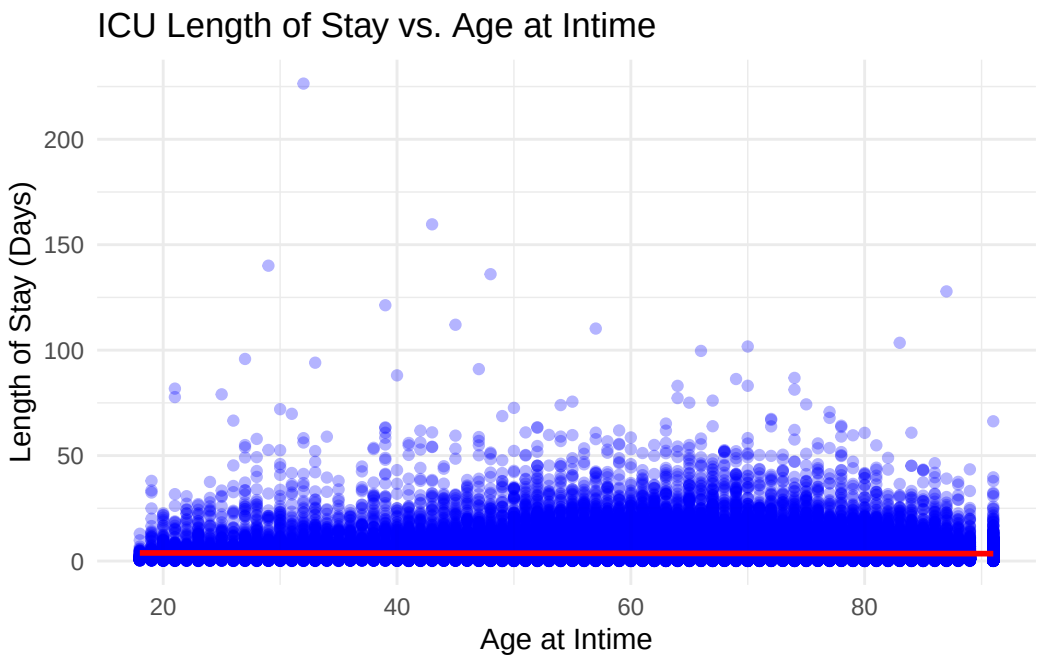
Next, we will plot length of stay by age at intime. We will utilize a scatter plot with an overlaid trend line to observe any monotonic patterns in the data.

```
ggplot(mimic_icu_cohort,
       aes(x = anchor_age,
           y = los)) +
  geom_point(alpha = 0.3,
             color = 'blue') +
  geom_smooth(method = 'lm', color = 'red') +  # Linear trend line
  labs(
    title = 'ICU Length of Stay vs. Age at Intime',
    x = 'Age at Intime',
    y = 'Length of Stay (Days)'
  ) +
  theme_minimal()
```

```
`geom_smooth()` using formula = 'y ~ x'
```

```
Warning: Removed 14 rows containing non-finite outside the scale range
(`stat_smooth()`).
```

```
Warning: Removed 14 rows containing missing values or values outside the scale range
(`geom_point()`).
```



Next, we will summarize length of stay by last available lab measurements before ICU Stay. We will use faceted scatter plots to summarize this information intuitively. As a secondary plot, I included faceted 2D histograms with color gradients denoting density.

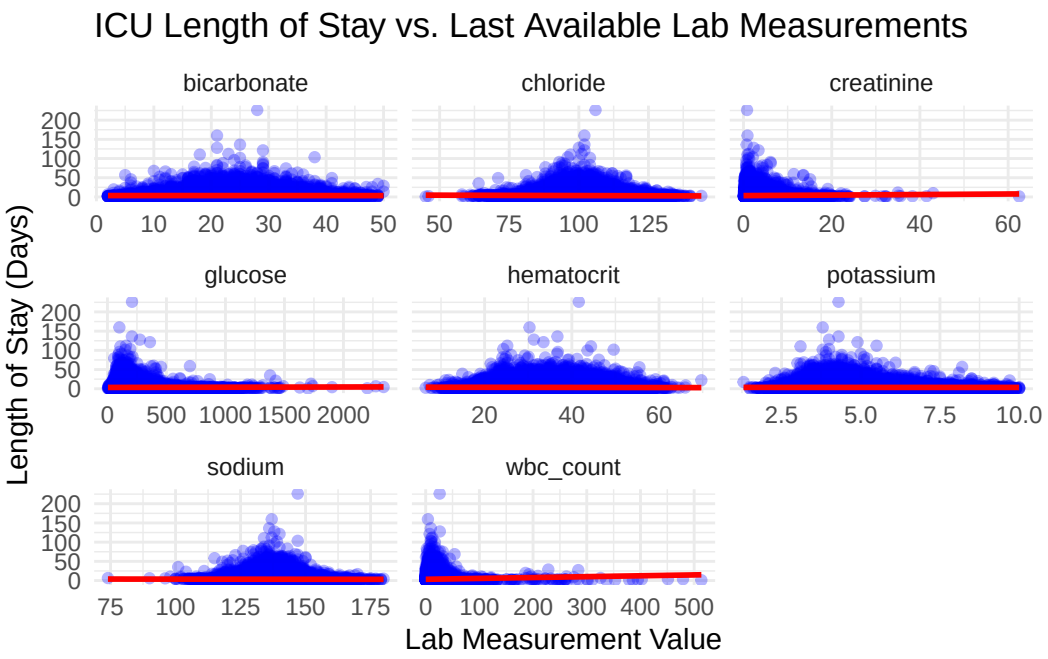
```
# Select relevant lab measurements
lab_vars <- c('bicarbonate',
              'chloride',
              'creatinine',
              'glucose',
              'potassium',
              'sodium',
              'hematocrit',
              'wbc_count')

# Pivot longer to create one column for lab names and values
mimic_icu_cohort_long <- mimic_icu_cohort %>%
  select(los,
         all_of(lab_vars)) %>%
  pivot_longer(cols = -los,
               names_to = 'Lab_Test',
               values_to = 'Lab_Value')

# Scatter plots with linear regression lines
```

```r
ggplot(mimic_icu_cohort_long,
       aes(x = Lab_Value,
           y = los)) +
  geom_point(alpha = 0.3,
             color = 'blue') +
  geom_smooth(method = 'lm',
              color = 'red') +  # Trend line
  facet_wrap(~ Lab_Test,
             scales = 'free_x') +  # Facet by lab test
  labs(
    title = 'ICU Length of Stay vs. Last Available Lab Measurements',
    x = 'Lab Measurement Value',
    y = 'Length of Stay (Days)'
  ) +
  theme_minimal()
```

```
`geom_smooth()` using formula = 'y ~ x'
```

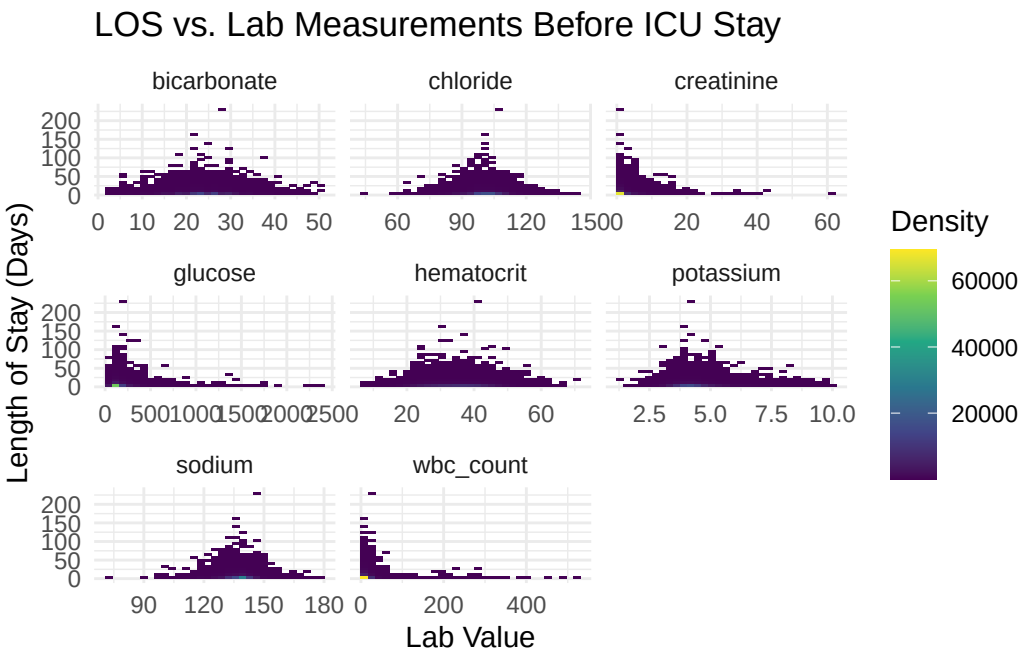
```
Warning: Removed 79011 rows containing non-finite outside the scale range
(`stat_smooth()`).
```

```
Warning: Removed 79011 rows containing missing values or values outside the scale range
(`geom_point()`).
```




```r
# Pivot longer to reshape lab variables for faceted histograms
mimic_icu_cohort_long <- mimic_icu_cohort %>%
  select(los,
         all_of(lab_vars)) %>%
  pivot_longer(cols = -los,
               names_to = 'Lab_Test',
               values_to = 'Lab_Value')

# Plot LOS (y-axis) vs Lab Value (x-axis) in faceted histograms
ggplot(mimic_icu_cohort_long,
       aes(x = Lab_Value,
           y = los)) +
  geom_bin2d(bins = 30) +
  scale_fill_viridis_c() +
  facet_wrap(~ Lab_Test,
             scales = 'free_x') +
  labs(
    title = 'LOS vs. Lab Measurements Before ICU Stay',
    x = 'Lab Value',
    y = 'Length of Stay (Days)',
    fill = 'Density'
  ) +
  theme_minimal()
```

```
Warning: Removed 79011 rows containing non-finite outside the scale range
(`stat_bin2d()`).
```



Next, we will plot length of stay against first ICU vital measurements. As we did above, we will include both faceted scatter plots and histograms to summarize this information visually:

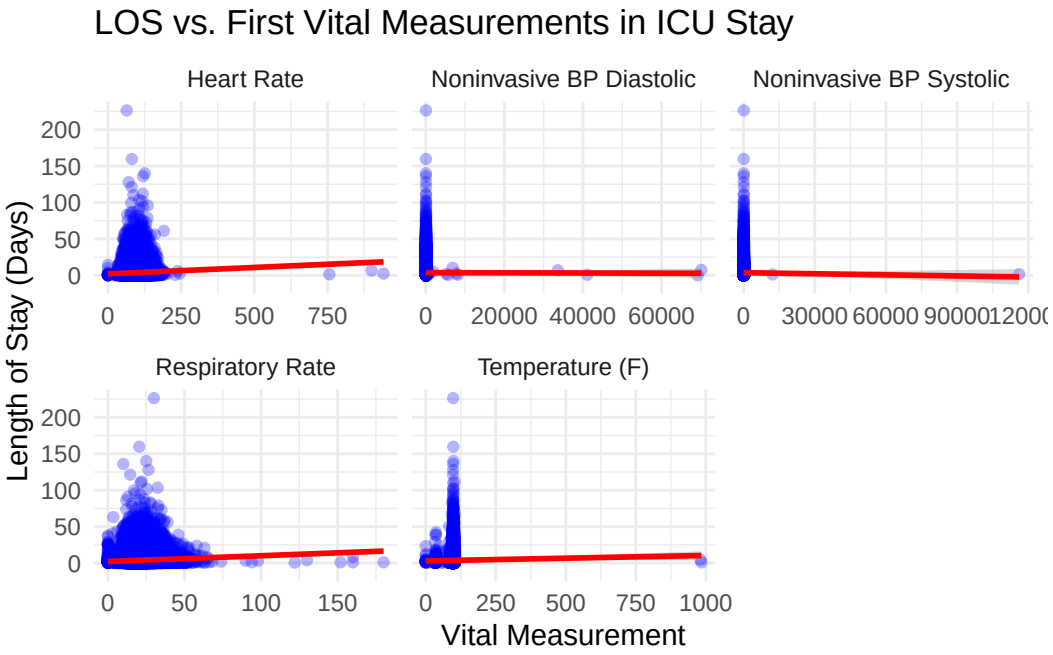
```
# Select LOS and the first ICU vital measurements
vital_vars <- c('Heart Rate',
                'Noninvasive BP Systolic',
                'Noninvasive BP Diastolic',
                'Respiratory Rate',
                'Temperature (F)')

# Pivot longer to reshape vital variables for faceted scatter plots
mimic_icu_cohort_long <- mimic_icu_cohort %>%
  select(los,
         all_of(vital_vars)) %>%
  pivot_longer(cols = -los,
               names_to = 'Vital_Sign',
               values_to = 'Vital_Value')

# Scatter plot of LOS vs. Vital Measurements
ggplot(mimic_icu_cohort_long, aes(x = Vital_Value, y = los)) +
```

```
  geom_point(alpha = 0.3, color = 'blue') +  # Transparent blue dots
  geom_smooth(method = 'lm', color = 'red', se = TRUE) +  # Linear trend line with confidence
  facet_wrap(~ Vital_Sign, scales = 'free_x') +  # Facet by vital sign
  labs(
    title = 'LOS vs. First Vital Measurements in ICU Stay',
    x = 'Vital Measurement',
    y = 'Length of Stay (Days)'
  ) +
  theme_minimal()
```

```
`geom_smooth()` using formula = 'y ~ x'
```

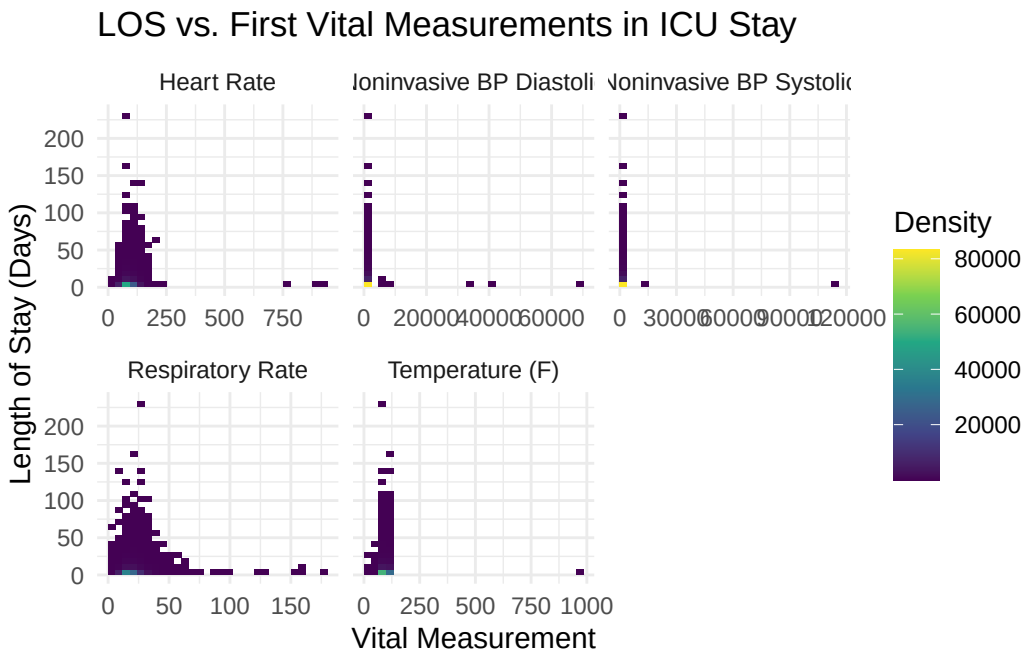
```
Warning: Removed 4511 rows containing non-finite outside the scale range
(`stat_smooth()`).
```

```
Warning: Removed 4511 rows containing missing values or values outside the scale range
(`geom_point()`).
```



```r
# Plot LOS (y-axis) vs Vital Measurement (x-axis) in faceted histograms
ggplot(mimic_icu_cohort_long,
       aes(x = Vital_Value,
           y = los)) +
  geom_bin2d(bins = 30) +
  scale_fill_viridis_c() +
  facet_wrap(~ Vital_Sign,
             scales = 'free_x') +  # Facet by vital sign
  labs(
    title = 'LOS vs. First Vital Measurements in ICU Stay',
    x = 'Vital Measurement',
    y = 'Length of Stay (Days)',
    fill = 'Density'
  ) +
  theme_minimal()
```

```
Warning: Removed 4511 rows containing non-finite outside the scale range
(`stat_bin2d()`).
```

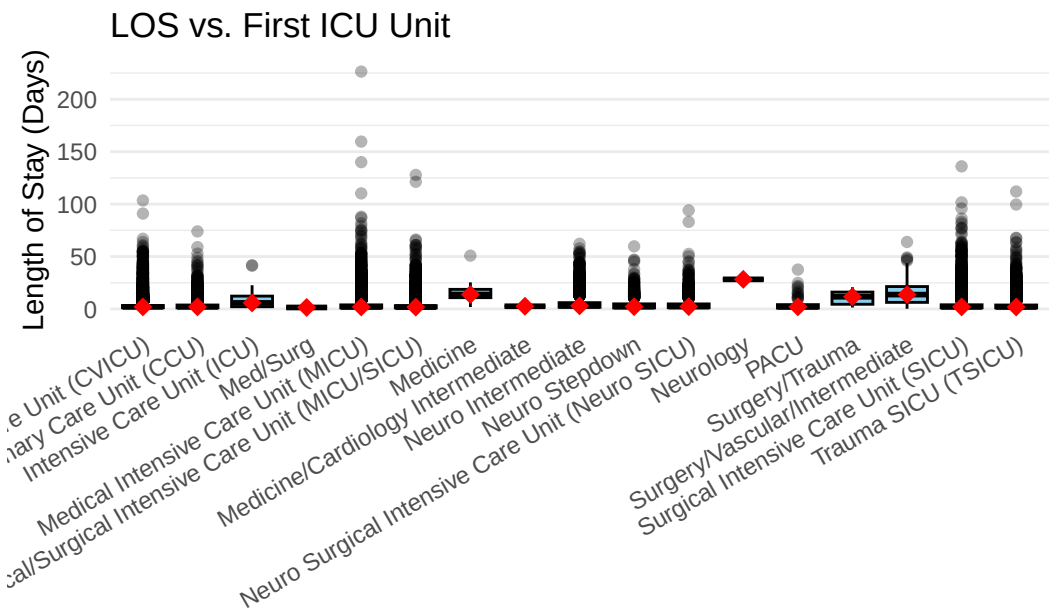


Lastly, we will plot length of stay against first ICU unit. We will first plot a series of box plots to summarize the distributions. For more clarity, we will also construct scatter plots along the same x-axis:

```r
# Box plot of LOS vs First ICU Unit
ggplot(mimic_icu_cohort, aes(x = first_careunit,
                             y = los)) +
  geom_boxplot(fill = 'skyblue',
               color = 'black',
               outlier.alpha = 0.3) +
  stat_summary(fun = median,
               geom = 'point',
               shape = 18,
               size = 3,
               color = 'red') +
  labs(
    title = 'LOS vs. First ICU Unit',
    x = 'First ICU Unit',
    y = 'Length of Stay (Days)'
  ) +
  theme_minimal() +
  theme(
    axis.text.x = element_text(angle = 30,
                               hjust = 1),
    panel.grid.major.x = element_blank()
  )
```

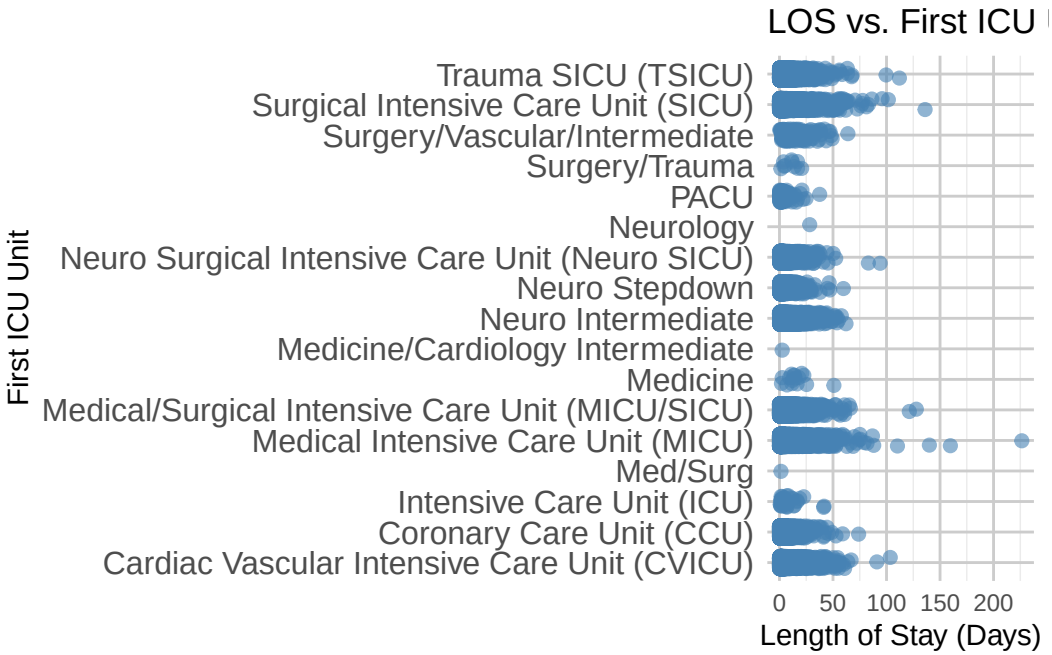
```
Warning: Removed 14 rows containing non-finite outside the scale range
(`stat_boxplot()`).
```

```
Warning: Removed 14 rows containing non-finite outside the scale range
(`stat_summary()`).
```




```
ggplot(mimic_icu_cohort, aes(y = first_careunit,
                             x = los)) +
  geom_jitter(alpha = 0.6,
              color = 'steelblue',
              height = 0.2,
              width = 0,
              size = 2) +
  labs(title = 'LOS vs. First ICU Unit',
       x = 'Length of Stay (Days)',
       y = 'First ICU Unit') +
  theme_minimal() +
  theme(axis.text.y = element_text(size = 12),
        panel.grid.major = element_line(color = "gray80"))
```

```
Warning: Removed 14 rows containing missing values or values outside the scale range
(`geom_point()`).
```


LOS vs. First ICU
First ICU Unit
Trauma SICU (TSICU)
Surgical Intensive Care Unit (SICU)
Surgery/Vascular/Intermediate
Surgery/Trauma
PACU
Neurology
Neuro Surgical Intensive Care Unit (Neuro SICU)
Neuro Stepdown
Neuro Intermediate
Medicine/Cardiology Intermediate
Medicine
Medical/Surgical Intensive Care Unit (MICU/SICU)
Medical Intensive Care Unit (MICU)
Med/Surg
Intensive Care Unit (ICU)
Coronary Care Unit (CCU)
Cardiac Vascular Intensive Care Unit (CVICU)
0
50
100
150
200
Length of Stay (Days)