OSEI Senior Data Analyst Exercise: Exploratory Analysis

Mayra Smith-Coronado

2024-06-24

Intro

The included datasets come from two tables within the DSD relational database: Incarceration and Person. The incarceration table's unit of analysis is one booking into the jail and includes booking-related data such as the times into and out of the jail. The person table's unit of analysis is one person and includes demographic information like age and race. The "Person_id" column is the common variable between them. (For this exercise, the real Person_id has been suppressed and replaced with a unique random number to protect identities.)

The point of this exercise is to demonstrate how you think analytically as much as it is to arrive at the "correct" answers. Please provide your best answers to the questions below, using the tools and methods you deem most effective. Please submit written answers in a clear and concise form by the deadline. **Please also share your code so we can review it.**

```r
# Load in Incarceration Data ---------------------------------------------
# note time is not going to be read in for this data set, only date information
incarceration_data <- read.xlsx("~/osei_data_exercise/01 - data/Incarceration.xlsx") %>%
  # convert to tibble
  tibble() %>%
  # update column names to follow snake case naming convention
  janitor::clean_names() %>%
  # correctly read excel dates as dates instead of numeric values
  mutate(across(.cols = c("date_in", "release_out"),
                .fns = janitor::excel_numeric_to_date))

# check to make sure that there are no duplicate booking numbers and that
# each row represents one booking
incarceration_data %>%
  count(booking_number) %>%
  filter(n > 1)
```

```
## # A tibble: 0 x 2
## # i 2 variables: booking_number <chr>, n <int>
```

```r
# Load in Person Data ----------------------------------------------------
person_data <- read.xlsx("~/osei_data_exercise/01 - data/Person.xlsx") %>%
  # convert to tibble
  tibble() %>%
  # update column names to follow snake case naming convention
  janitor::clean_names() %>%
  # correctly read excel dates as dates instead of numeric values
```

```r
  mutate(across(.cols = c("dob"),
                .fns = janitor::excel_numeric_to_date))

# check to make sure that there are no duplicate people and that
# each row represents one person
person_data %>%
  count(person_id) %>%
  filter(n > 1)
```

```
## # A tibble: 0 x 2
## # i 2 variables: person_id <dbl>, n <int>
```

Exercise

Consider the year from July 1, 2021 to June 30, 2022 as the analysis period.

---

1. How many total bookings into the jail were there in that time period?

```r
bookings_in_analysis_period <- incarceration_data %>%
  filter(date_in >= "2021-07-01" &
           date_in <= "2022-06-30")

nrow(bookings_in_analysis_period) %>%
  scales::comma()
```

```
## [1] "21,842"
```

---

2. How many unique people were booked into the jail?

```r
bookings_in_analysis_period %>%
  select(person_id) %>%
  distinct() %>%
  count() %>%
  pull() %>%
  scales::comma()
```

```
## [1] "15,510"
```

---

3. How many people were in the jail at the moment of the data extraction?

This would include not only the people who were booked during the analysis period, but the people who were booked before analysis period, but were not yet released

```
# gather bookings where a person was booked before the analysis period, but
# has no release date
bookings_still_incarcerated <- incarceration_data %>%
  filter(date_in < "2021-07-01",
         is.na(release_out))

nrow(bookings_still_incarcerated) %>%
  scales::comma()
```

```
## [1] "130"
```

```
bookings_still_incarcerated %>%
  summarise(first_booking = min(date_in),
            last_booking = max(date_in))
```

```
## # A tibble: 1 x 2
##   first_booking last_booking
##   <date>        <date>
## 1 2016-09-07    2021-06-30
```

```
# gather bookings where a person was booked before the analysis period,
# but they were released within the analysis period
bookings_released_during_extraction <- incarceration_data %>%
  filter(date_in < "2021-07-01",
         release_out >= "2021-07-01" &
           release_out <= "2022-06-30")

nrow(bookings_released_during_extraction) %>%
  scales::comma()
```

```
## [1] "1,276"
```

```
bookings_released_during_extraction %>%
  summarise(first_release = min(release_out),
            last_release = max(release_out))
```

```
## # A tibble: 1 x 2
##   first_release last_release
##   <date>        <date>
## 1 2021-07-01    2022-06-29
```

```
# now create a table with all the bookings that we have identified to
# have occurred during the extraction and bookings that
# occurred before the extraction, but were not yet released
bookings_during_extraction <- bookings_in_analysis_period %>%
  bind_rows(bookings_still_incarcerated) %>%
```

```
  bind_rows(bookings_released_during_extraction)

# verify no duplicate bookings
bookings_during_extraction %>%
  count(booking_number) %>%
  filter(n > 1)
```

```
## # A tibble: 0 x 2
## # i 2 variables: booking_number <chr>, n <int>
```

```
# now count the number of people in jail at the moment of the
# data extraction
bookings_during_extraction %>%
  select(person_id) %>%
  distinct() %>%
  count() %>%
  pull() %>%
  scales::comma()
```

```
## [1] "16,456"
```

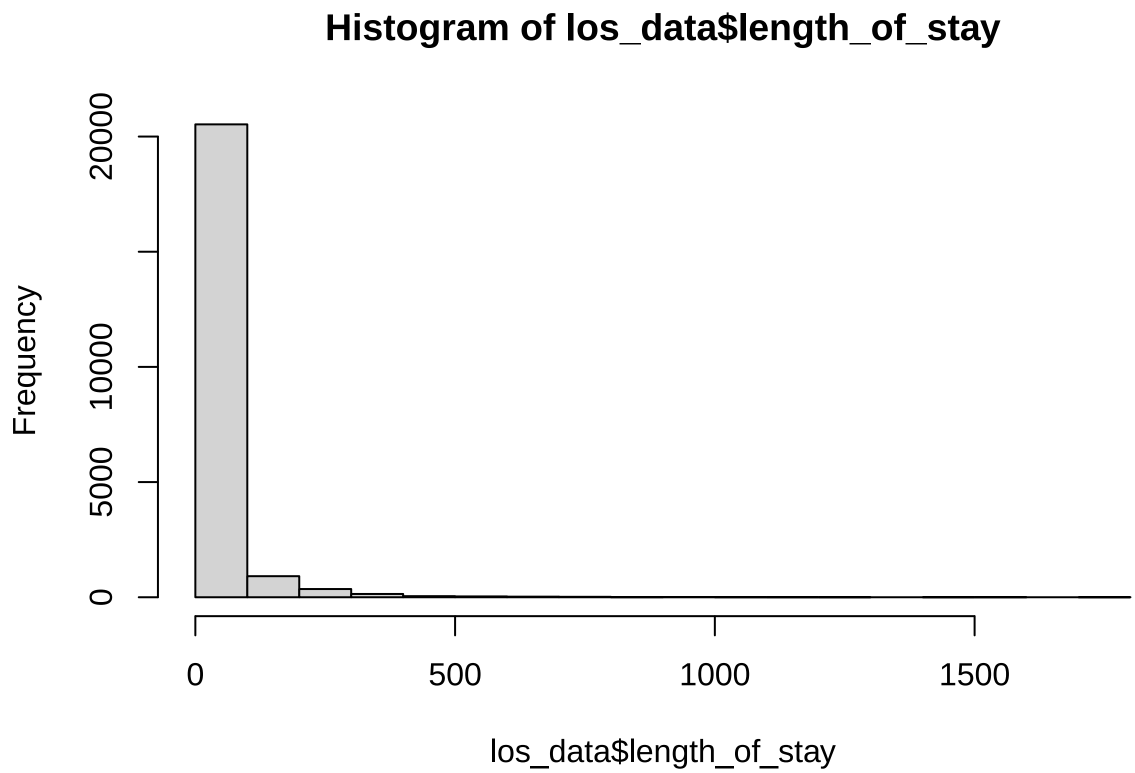
---

4. Consider the length of stay (LOS): the duration of each booking. Describe the LOS over the year analysis period. What insights (statistical and otherwise) does it provide you about variations in the jail population?

To look at length of stay during the analysis period, I think it is best to look at the bookings that were from July 1, 2021 to June 30, 2022. This include people who were booked before the time period, but not released.

- About 48.5% of bookings had a length of stay of at least 2 days.
- The most frequent length of stay being 1 day.
- There are about 25% of bookings that ranged from 14 days to 1,719.
- About 75.3% of people were only booked once

```
los_data <- bookings_during_extraction %>%
  # remove bookings with no release date
  filter(!is.na(release_out)) %>%
  # calculate length of stay
  mutate(length_of_stay = as.numeric(release_out - date_in))


# examine the distribution of the length of stay for this analysis period
hist(los_data$length_of_stay)
```




```r
# get descriptive statistics to better understand the distribution
los_data %>%
  select(length_of_stay) %>%
  summary()
```

```
##  length_of_stay
##  Min.   :   0.0
##  1st Qu.:   1.0
##  Median :   3.0
##  Mean   :  26.1
##  3rd Qu.:  14.0
##  Max.   :1719.0
```

```r
los_data %>%
  tabyl(length_of_stay) %>%
  adorn_pct_formatting() %>%
  slice(1:21)
```

```
##  length_of_stay    n percent
##               0 2404   10.9%
##               1 6060   27.4%
##               2 2254   10.2%
##               3 1238    5.6%
##               4  871    3.9%
##               5  775    3.5%
```

```
##                6  742    3.4%
##                7  668    3.0%
##                8  486    2.2%
##                9  283    1.3%
##               10  225    1.0%
##               11  178    0.8%
##               12  189    0.9%
##               13  195    0.9%
##               14  160    0.7%
##               15  128    0.6%
##               16  119    0.5%
##               17   83    0.4%
##               18  117    0.5%
##               19   67    0.3%
##               20   92    0.4%
```

```
los_data %>%
  mutate(grouped_los = ifelse(length_of_stay >= 14, "14+", length_of_stay),
         grouped_los = factor(grouped_los, levels = c(0:13, "14+"))) %>%
  tabyl(grouped_los) %>%
  adorn_pct_formatting()
```

```
##  grouped_los    n percent
##            0 2404   10.9%
##            1 6060   27.4%
##            2 2254   10.2%
##            3 1238    5.6%
##            4  871    3.9%
##            5  775    3.5%
##            6  742    3.4%
##            7  668    3.0%
##            8  486    2.2%
##            9  283    1.3%
##           10  225    1.0%
##           11  178    0.8%
##           12  189    0.9%
##           13  195    0.9%
##          14+ 5526   25.0%
```

```
# how many times are people rebooked (booked at least one time)
rebooking <- los_data %>%
  arrange(person_id, date_in) %>%
  group_by(person_id) %>%
  mutate(frequency_booked = 1:n()) %>%
  ungroup() %>%
  arrange(person_id, desc(frequency_booked)) %>%
  distinct(person_id, .keep_all = T) %>%
  mutate(booked_multiple_times = ifelse(frequency_booked >= 2, "2+", "1"))

# review the total times people have been booked
rebooking %>%
  tabyl(frequency_booked) %>%
  adorn_pct_formatting()
```

```
##   frequency_booked     n percent
##                  1 11950   75.3%
##                  2  2543   16.0%
##                  3   840    5.3%
##                  4   315    2.0%
##                  5   115    0.7%
##                  6    59    0.4%
##                  7    28    0.2%
##                  8     7    0.0%
##                  9     6    0.0%
##                 13     1    0.0%
##                 14     1    0.0%
##                 16     1    0.0%
```

```r
# review how many people have been booked more than once
rebooking %>%
  tabyl(booked_multiple_times) %>%
  adorn_pct_formatting()
```

```
##   booked_multiple_times     n percent
##                       1 11950   75.3%
##                      2+  3916   24.7%
```

From the review of the distribution, a survival analysis would make the most sense. In this scenario, we would be trying to understand the amount of time it takes a person who was booked to be released from jail during our analysis period. This data set should include people who were in jail before the analysis period as well, but were not yet released.

- release dates after study period are considered censored event
- release dates that are missing are considered a censored event

```r
# create the event flag and create a time flag that represents length of stay
# remembering that this length of stay is until the end of the study period
survival_data <- bookings_during_extraction %>%
  mutate(event = case_when(
    is.na(release_out) ~ 0,
    release_out > "2022-06-30" ~ 0,
    T ~ 1)) %>%
  mutate(time = case_when(
    is.na(release_out) ~ as.numeric(ymd("2022-06-30") - date_in),
    release_out > "2022-06-30" ~ as.numeric(ymd("2022-06-30") - date_in),
    T ~ as.numeric(release_out - date_in)
  )) %>%
  select(booking_number, time, event)

# calculate the Kaplan-Meier estimate
km <- survfit(Surv(time, event) ~ 1,
  data = survival_data
)

km
```

```
## Call: survfit(formula = Surv(time, event) ~ 1, data = survival_data)
```

```
##
##          n events median 0.95LCL 0.95UCL
## [1,] 23248  21582      3       3       3
```

```
summary(km)$table["median"]
```

```
## median
##      3
```

A next step would be to examine if an individual being booked again during the study period would impact their length of stay of examine length of stay by demographics.

---

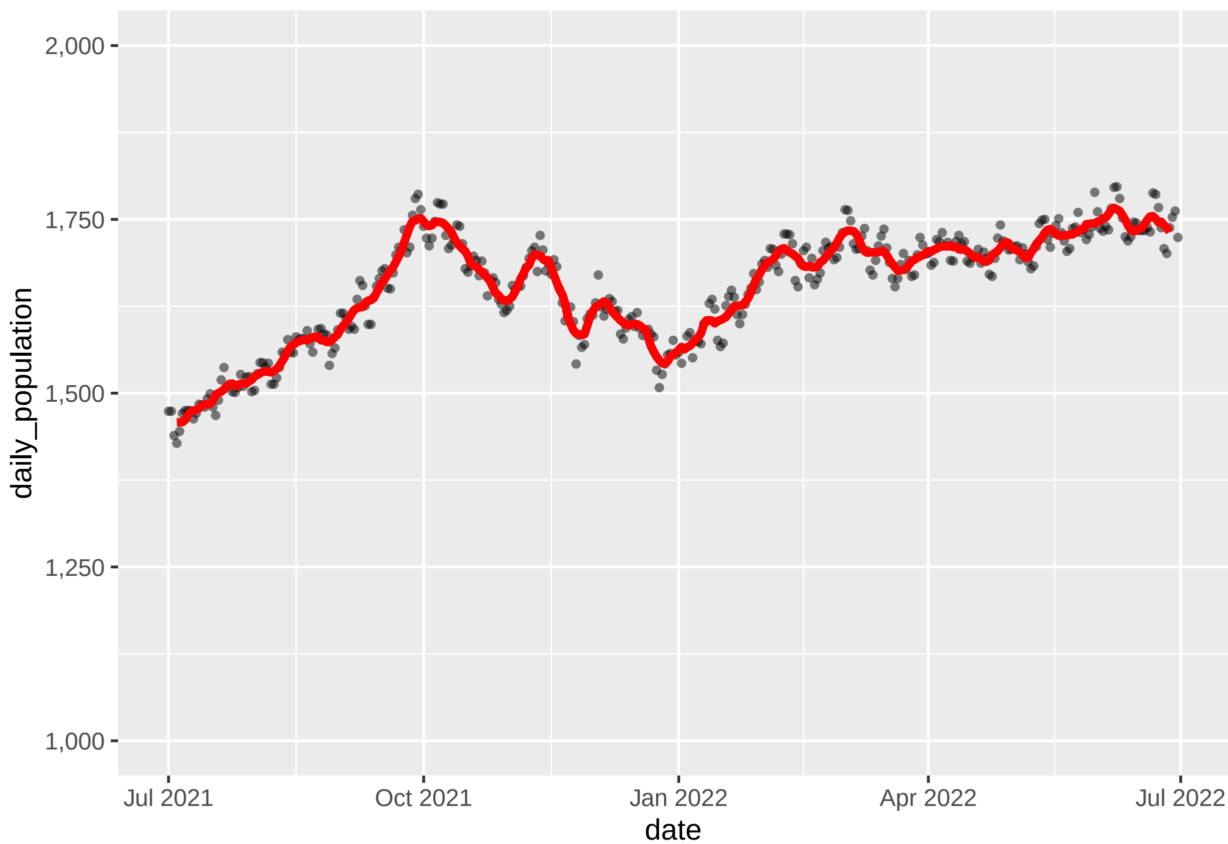
5. What was the average daily population in the jail during that year? Daily population ought to include anyone who spent even one minute in the jail in a given day. Please describe the methods/approach you used to answer this question. What tool did you use? What functions or other capabilities within the tool? (That is, help another analyst replicate what you did. Sharing code with your answers is encouraged but by no means required.)

```
booking_long = NULL

for(ith_booking in 1:nrow(bookings_during_extraction)){

  booking = bookings_during_extraction[ith_booking, ]

  if(!is.na(booking$release_out)) {
    release_out <- booking$release_out
  } else {
     release_out = as.Date("2022-06-30")
  }

  dates_incarcerated = seq(booking$date_in, release_out, by = "1 day")
  ith_booking_long <- tibble(date = dates_incarcerated,
                             booking_number = booking$booking_number,
                             person_id = booking$person_id)
  booking_long <- booking_long %>%
    bind_rows(ith_booking_long)
}
```

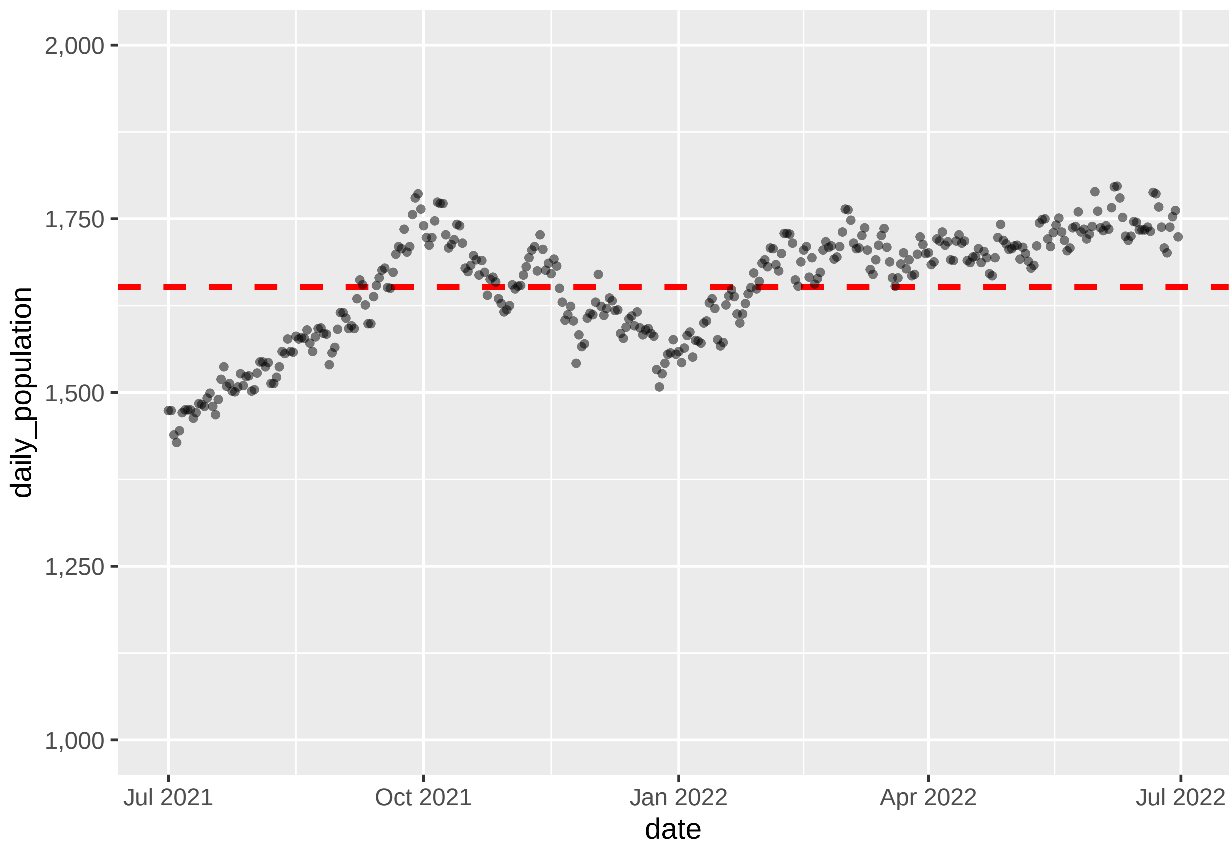
```
# count the number of people in jail each day
bookings_by_day <- booking_long %>%
  filter(date >= "2021-07-01" & date <= "2022-06-30") %>%
  arrange(date) %>%
  group_by(date) %>%
  count() %>%
  ungroup() %>%
  rename(daily_population = n)

# What is the average daily population for the analysis period?
(average_daily_population <- mean(bookings_by_day$daily_population))
```

```
## [1] 1651.923
```

```r
# what does the daily population look like overtime? note using a moving average
# to smooth the estimates a little
bookings_by_day %>%
  mutate(seven_day_moving_average = zoo::rollmean(daily_population, k = 7, fill = NA)) %>%
  ggplot(aes(x = date, y = daily_population)) +
  geom_point(alpha = 0.5, size = 1) +
  geom_line(aes(x = date, y = seven_day_moving_average), color = "red", size = 1.5) +
  scale_y_continuous(limit = c(1000, 2000), labels = comma)
```




```r
# View the daily population against the average daily population. Around
# what months is the daily population greater than the average?
bookings_by_day %>%
  ggplot(aes(x = date, y = daily_population)) +
  geom_hline(aes(yintercept = average_daily_population), color = "red", size = 1, linetype = "dashed") +
  geom_point(alpha = 0.5, size = 1) +
  scale_y_continuous(limit = c(1000, 2000), labels = comma)
```



---

6. Which day during that year had the lowest daily population? What was the population that day? Which day had the highest daily population? What was the population that day?

```r
# date with the min population
bookings_by_day %>%
  filter(daily_population == min(daily_population))
```

```
## # A tibble: 1 x 2
##   date       daily_population
##   <date>                <int>
## 1 2021-07-04             1428
```

```r
# date with the max population
bookings_by_day %>%
  filter(daily_population == max(daily_population))
```

```
## # A tibble: 1 x 2
##   date       daily_population
##   <date>                <int>
## 1 2022-06-08             1797
```

---

7. Please provide a basic analysis of jail demographics during that year period. Are there meaningful statistical relationships among the different groups in the jail?

---

8. What other insights, useful facts, or questions/concerns did you uncover, if any, in the data?