Mid term Assignment

```
setwd("/Users/jessicasaini/Desktop/UoW/Stat 847/MidTerm Stat 847")
library(GGally)
```

Word Count: 1768 words

```
## Loading required package: ggplot2
```

```
## Warning in register(): Can't find generic `scale_type` in package ggplot2 to
## register S3 method.
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1
## v purrr   0.3.4
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(mice)
```

```
##
## Attaching package: 'mice'
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

```
hotel_door = read.csv("hotel_door.csv")
#head(hotel_door)

hotel_elevator = read.csv("hotel_elevator.csv")
#head(hotel_elevator)

hotel_desk = read.csv("hotel_frontdesk.csv")
#head(hotel_desk)
```

Question 1

Answer

The information about the checkin, check-out is recorded in three different datasets , i.e hotel_desk, hotel_elevator and hotel_door respectively. We can explore these datasets independently and in combination to check for security flaws, see the activity of customers, check for some suspicious activity and to find different patterns/stories.

Workflow of the guest Check-in: The guest checks-in at the front desk at a particular date and time (recorded in hotel_frontdesk.csv), uses the cars in hotel elevator (recorded in hotel_elevator.csv) to access the room and unlocks the room using the hotel_key. If the open_success = True, the room is successfully unlocked (recorded in hotel_door.csv).

Check-Out:

The guests checkout at the front-desk. The date and time of the checkout are recorded.

We can trace the activity of the guests living in the hotel to some extent by combining the datasets. Since it is the same guests over the same time, we can merge the hotel_frontdesk and hotel_door dataset.

Combining Hotel_desk and Hotel_door based on guest_id

By merging these two datasets based on column **"guest_id"**, we can analyse if there were any customers who went to floors other than the ones on which their room was assigned/booked.

Upon combining, it is observed that there was one guest who went to floors other than the ones on which the room is booked.

```
merge_desk_door = merge(hotel_desk,hotel_door,by="guest_id",all=TRUE)
o = merge_desk_door%>% filter(merge_desk_door$floor.x!=merge_desk_door$floor.y)
unique(o$guest_id)
```

```
## [1] 1024
```

Combining hotel_desk and hotel_elevator based on guest_id and room_id

- Change the "user_id" in hotel_elevator to guest_id.
- The use of an elevator by a particular user can be recorded by combining the hotel_Desk and hotel_elevator. The guest_id for each guest is unique and room_id gives information about the room and the floor that the guest is staying in. If the guest uses the elevator to go to a floor that is different from his room_id, it can be easily traced by combining these datasets. The next step is to filter the data on the merged dataset based on the condition that "floor"(in hotel_desk) is not equal "to" (hotel_elevator). Since the values in "to" column are either 1 or the floor assigned, we use the following query.

```
hotel_elevator2 = hotel_elevator
colnames(hotel_elevator2)[colnames(hotel_elevator2) == 'user_id'] <- 'guest_id'
merge_desk_elevator = merge(hotel_desk,hotel_elevator2,by=c("guest_id","room_id"),all=TRUE)
h = merge_desk_elevator%>% filter(merge_desk_elevator$floor!=merge_desk_elevator$to
    && merge_desk_elevator$floor!=1)
dim(h)
```

```
## [1]  0 22
```

We find that no guest used the elevator to go any floor other than the ones in which their room was assigned, not even Guest 1024.

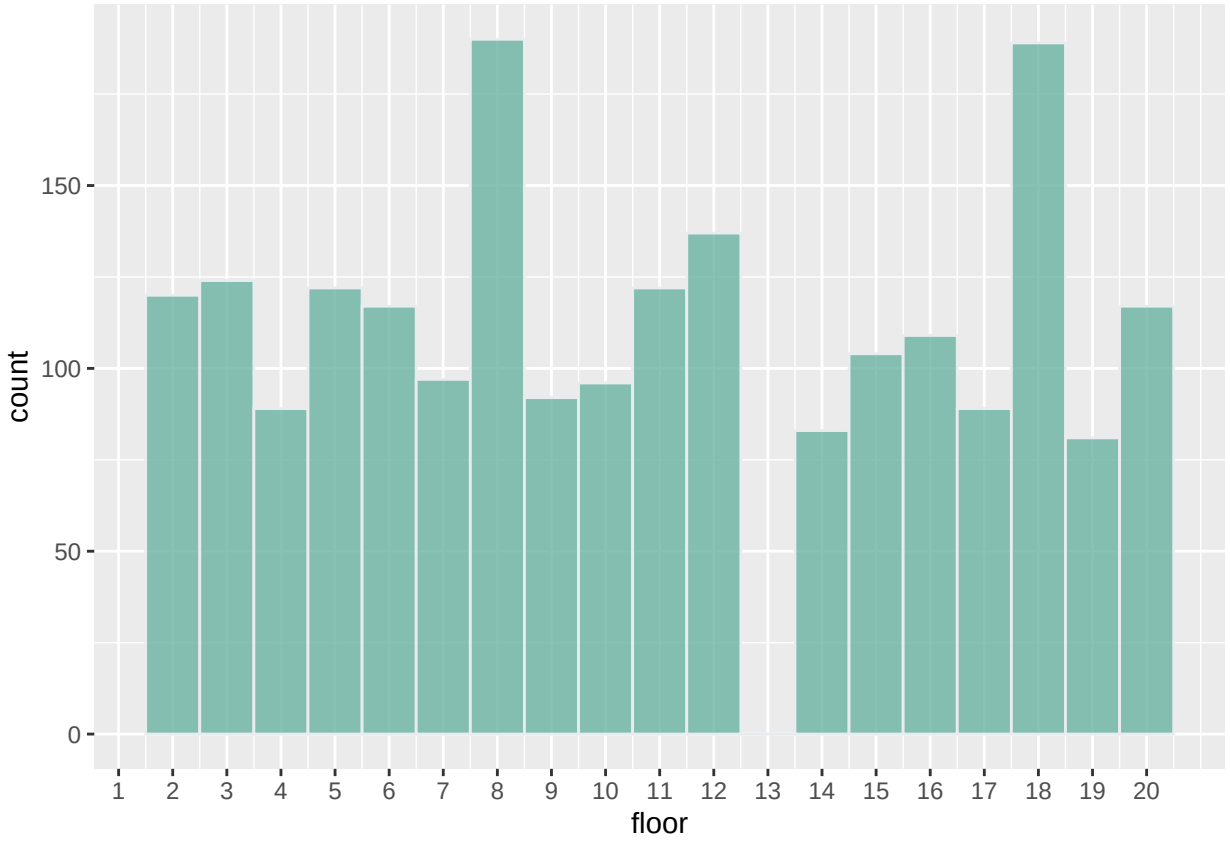
Independent Exploration of each dataset

Upon independently exploring the hotel_desk, one may assume that there is no Floor 13 and there are perhaps no rooms on Floor 1. Moreover, if we explore the hotel_elevator dataset alone, we see that the

elevator does not stop at Floor 13 (because no rooms are booked on that floor as per the data available in hotel_Desk).

```
#Exploring hotel_desk
ggplot(data= hotel_desk, mapping=aes(x=floor)) + geom_histogram(binwidth=1,
fill="#69b3a2", color="#e9ecef", alpha=0.8)+ scale_x_continuous(breaks=1:20)
```

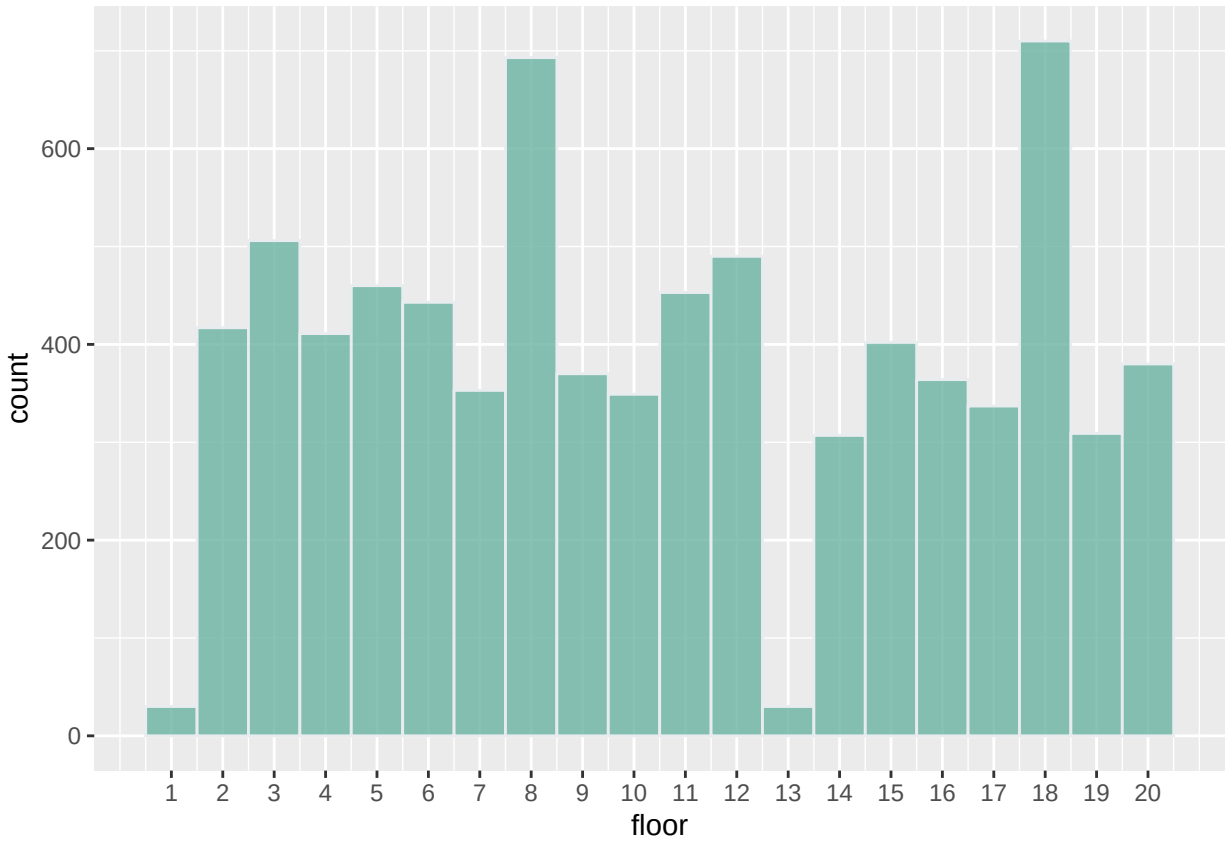
```
## Warning: Removed 17 rows containing non-finite values (stat_bin).
```



Based on these assumptions, one may infer that Floor 13 does not exist. However, this analysis is not correct. Upon exploring the hotel_door dataset, we see that there are entries for Floor 13. Moreover, we find that entries for rooms on Floor 1 also exist. This can further be explored by combining the datasets as referred above.

```
#Exploring hotel_door
ggplot(data= hotel_door, mapping=aes(x=floor)) + geom_histogram(binwidth=1,
fill="#69b3a2", color="#e9ecef", alpha=0.8)+ scale_x_continuous(breaks=1:20)
```

```
## Warning: Removed 176 rows containing non-finite values (stat_bin).
```



Combining the three data sets for a particular guest

While combining the three datasets will generate a large number of rows and we may not be able to derive meaningful information from it, we can combine the three datasets for guests with some suspicious activity. As explained in detail below, Guest 1024 had entries on each floor. Therefore, we combined the hotel_desk, hotel_elevator and hotel_door where guest_id==1024.

This can also help in detecting if the guest used the elevator to access other floors or accessed the floor through other methods.

Attempting to find missing values

We can filter the list of guest_id for all the rooms for which the data is missing. Since the guest_id is unique is across all the datasets, the list of guest_ids can be used in exploring the hotel_door and hotel_elevator dataset.

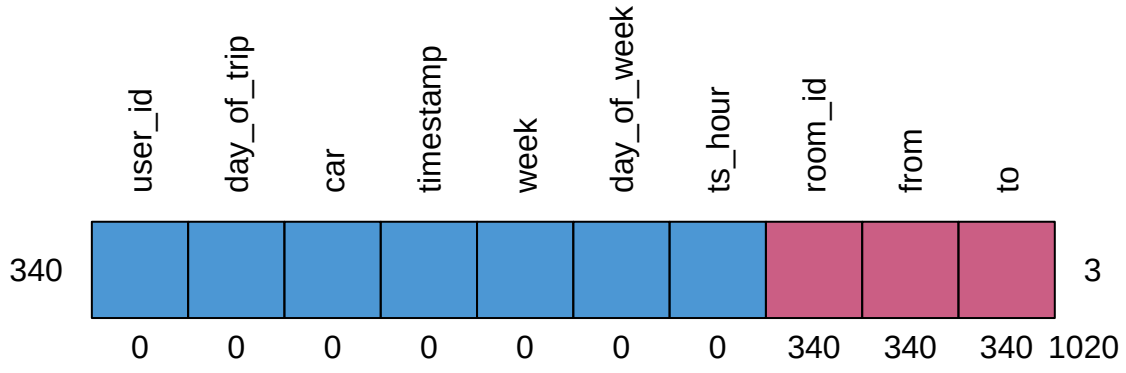
Since every guest uses a elevator to go to his room, we can check the elevator dataset for the list of guest_ids and see if "from", "to" and "floor" variables are available.

Similarly, we can check the hotel_door dataset, to see if the room_id, room_on_floor and floor are available.

If any of the datasets, have values for the variables mentioned above, the NA values for the guests can be filled.

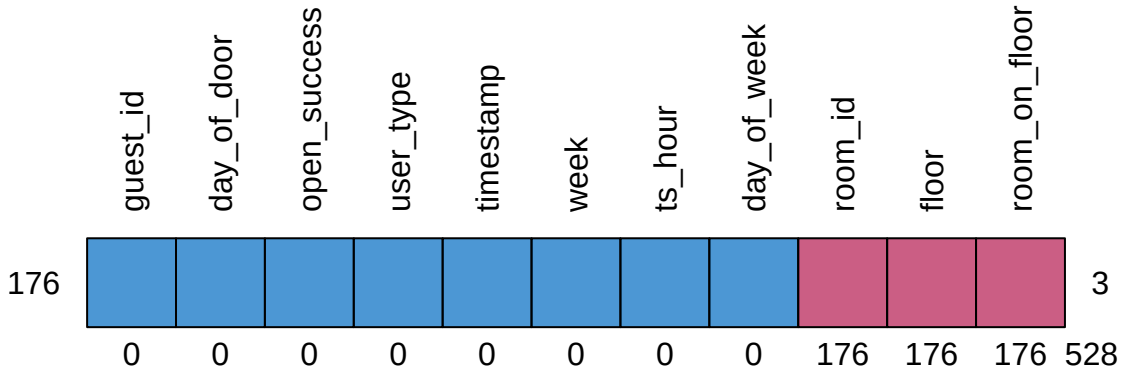
```
r = subset( hotel_desk,is.na(hotel_desk$room_id))
s = unique(r$guest_id)

m =  hotel_elevator %>% filter(hotel_elevator$user_id%in%s)
md.pattern(m,rotate.names = TRUE)
```




```
##     user_id day_of_trip car timestamp week day_of_week ts_hour room_id from  to
## 340       1           1   1         1    1           1       1       0    0   0
##           0           0   0         0    0           0       0     340  340 340
##
## 340    3
##     1020
```

```
j = hotel_door%>% filter(hotel_door$guest_id%in%s)
md.pattern(j,rotate.names=TRUE)
```




```
##     guest_id day_of_door open_success user_type timestamp week ts_hour
## 176        1           1            1         1         1    1       1
##            0           0            0         0         0    0       0
##     day_of_week room_id floor room_on_floor
## 176           1       0     0             0   3
##               0     176   176           176 528
```

We see that room_id, floor and room_on_floor are missing across all the three datasets for the 17 guests.

Question 2

Stories in Dataset

- There are 20 rooms on each floor and Room #1 on any floor is never booked.
- Even though, the rooms on Floor 13 are not booked. Floor 13 exists and there are 20 rooms on it.
- **Guest 1024**

```
b = hotel_door %>% filter(hotel_door$open_success==FALSE)
#table(b$guest_id,b$floor)

c = b %>% filter(b$guest_id==1024)
```

```
table(c$guest_id,c$floor)
```

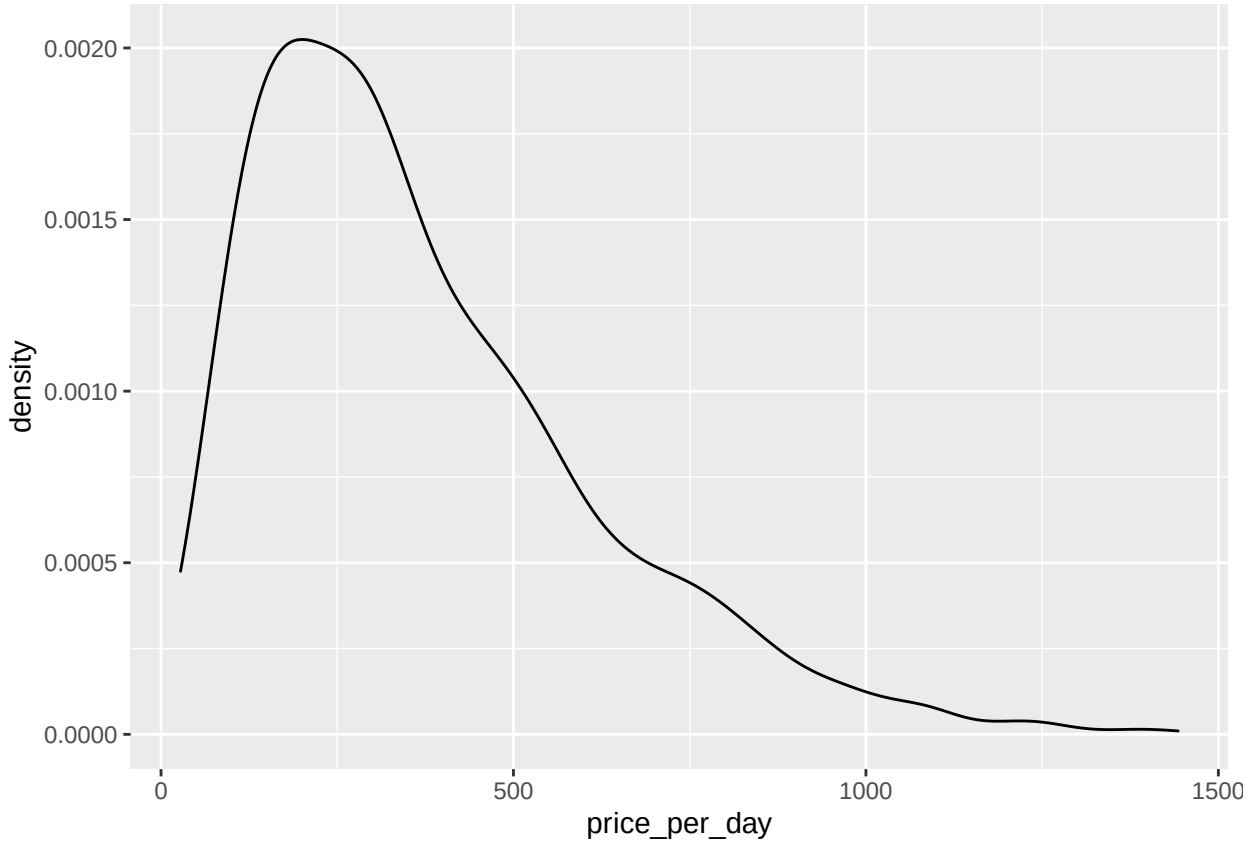
```
##
##          1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##   1024 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
```

It is interesting to note that Guest 1024 has entries on each floor. Upon exploring further, we find that Guest 1024 tried accessing every room on each floor i.e. Room 1-20 on Floor 1-20. This includes Floor 1 and Floor 13 also.

Now, we can try to check if the guest used elevator to access this floor. Upon combining the three data sets, we find that the guest 1024 only used the elevator to go to Floor 5 from Floor 1. One plausible explanation could be that the guest used the stairs to access the floor or may have collaborated with any staff member who has access to other floors.

- **Rooms Never Booked**: Rooms with room_id 607, 412, 1710, 1705, 1525, 1428, 1412 are never booked along with Room 1 on each floor.
- **Expensive Booking**: Most expensive rooms were booked in Week 3.
- **Room Prices**: There is huge variation in the prices for rooms. Some rooms are booked for less than 40 dollars and others more than 1000 for one night.

```
price_per_day = hotel_desk$price/hotel_desk$length_of_stay
ggplot(data= hotel_desk, aes(x=price_per_day))+ geom_density()
```

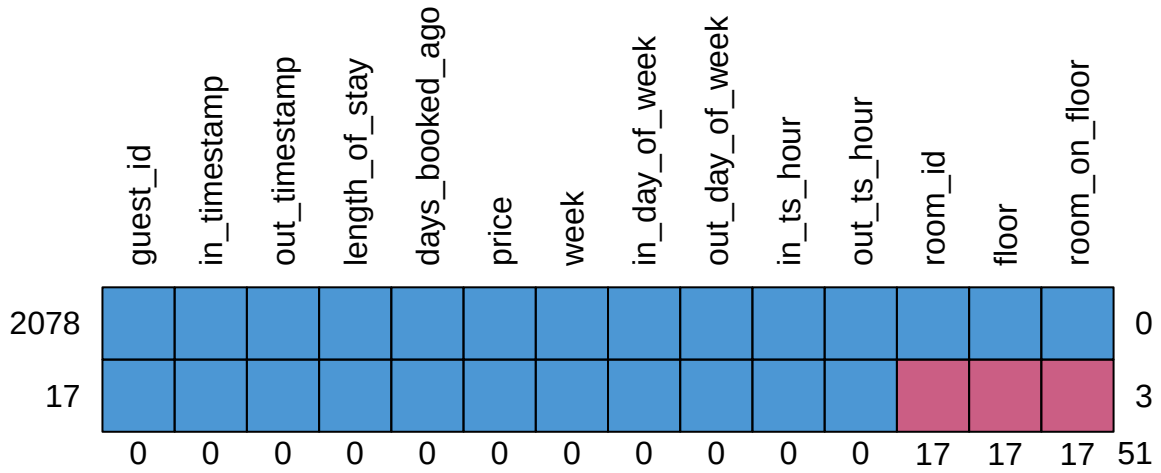


- **Room 404:** Guest 2022 staying in Room 404 faced maximum issues with his room lock. He wasnot successful in opening the door for 30 times during his stay of 2 days. Similarly, another guest 10164 who stayed in this room was unable to open the door 12 times. This indicates that there could be some issue with the room lock.

- **Missing Information for rooms**: The data in the datasets is recorded for 13 weeks. For some guests, the information related to room_id, room num and floor is missing. Upon visualizing the data, we see that 14/17 of the missing values for room_id and other variables come from Week 3 and Week 8.

This indicates that data is not missing completely missing at random (not MCAR).

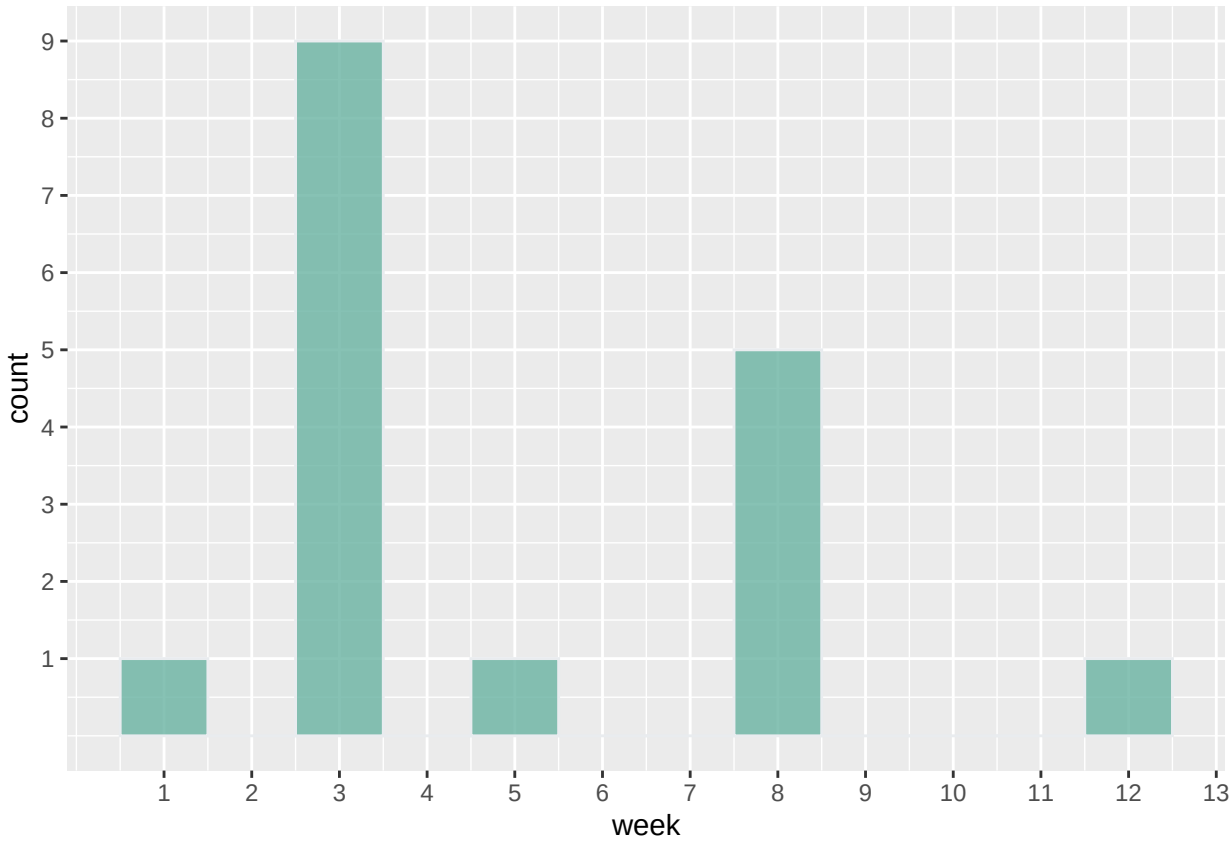
```
md.pattern(hotel_desk, rotate.names = TRUE)
```




```
##      guest_id in_timestamp out_timestamp length_of_stay days_booked_ago price
## 2078        1            1             1              1               1     1
## 17          1            1             1              1               1     1
##             0            0             0              0               0     0
##      week in_day_of_week out_day_of_week in_ts_hour out_ts_hour room_id floor
## 2078    1              1               1          1           1       1     1
## 17      1              1               1          1           1       0     0
##         0              0               0          0           0      17    17
##      room_on_floor
## 2078             1  0
## 17               0  3
##                 17 51
```

```
l = subset( hotel_desk,is.na(hotel_desk$room_id))
l["price_per_day"] = l$price/l$length_of_stay

ggplot(data= l, mapping=aes(x=week)) + geom_histogram(binwidth=1,
fill="#69b3a2", color="#e9ecef", alpha=0.8)+
scale_x_continuous(breaks=1:13)+ scale_y_continuous(breaks=1:13)
```



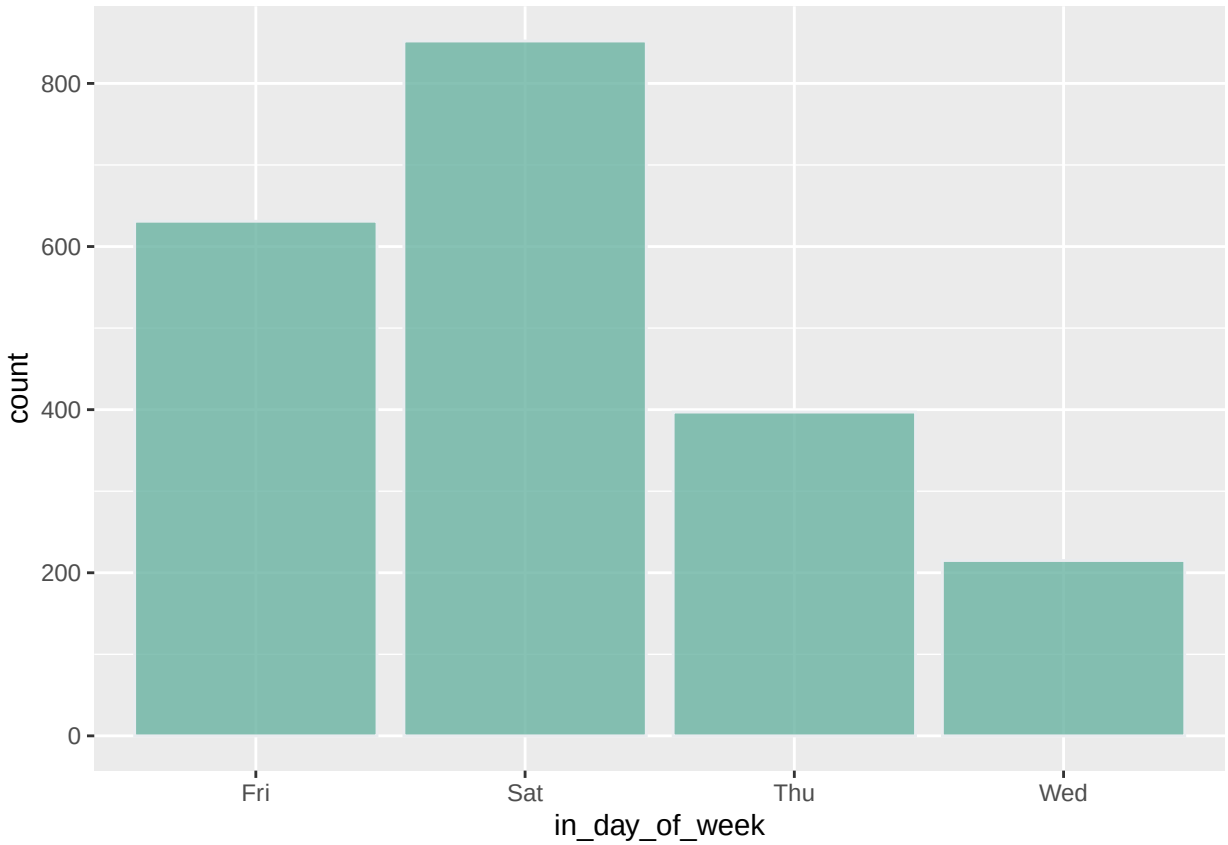
- **Missing room information for guests paying high prices**: No room information for guests paying exorbitantly high prices. For all the guest who are booking a room for more than 4000 dollars; room_id, floor_id, room_on_floor is not mentioned. The room_ids are also missing in hotel_elevator dataset for these guests.

```
k = hotel_desk %>% filter(hotel_desk$price>=4000)
print(k[,c(1,5,6,7,9)])
```

```
##   guest_id room_id floor room_on_floor   price
## 1     3034      NA    NA            NA 4134.91
## 2     3099      NA    NA            NA 4065.48
## 3     3120      NA    NA            NA 4831.25
## 4     3214      NA    NA            NA 4015.39
## 5     3220      NA    NA            NA 4581.40
## 6     8006      NA    NA            NA 4741.74
## 7     8101      NA    NA            NA 4639.93
```

- **Check-ins on certain days**: While guests checkout on all seven days of the week, they check-in on Wednesday, Thursday, Friday and Sat. Most of check-ins happen on Saturday.

```
ggplot(data= hotel_desk, mapping=aes(x=in_day_of_week)) +
geom_bar(fill="#69b3a2", color="#e9ecef", alpha=0.8)
```



- **Rooms booked for a week**: All the rooms booked for length_of stay==7 were booked on Wednesday. Total number of people is 13.
- No trips made in elevators from 0 to hour 4 (midnight)
- **Security Flaws**: Most security flaws happened on Monday.This is because Guest 1024 tried accessing the rooms using his hotel key on Monday

```
b = hotel_door %>% filter(hotel_door$open_success==FALSE)
table(b$open_success , b$day_of_week)
```

```
##
##         Fri Mon Sat Sun Thu Tue Wed
##   FALSE  29 624  85  22  15  13   3
```

Question 3: Build a model to describe and predict hotel prices.

```
corrs = hotel_desk[,c(9,1,2,3,4,5,6,7,8,10,11,12,13,14)]
ggcorr(corrs)
```

```
## Warning in ggcorr(corrs): data in column(s) 'in_timestamp', 'out_timestamp',
## 'in_day_of_week', 'out_day_of_week' are not numeric and were ignored
```


```
mod1 = lm(price~length_of_stay+floor+guest_id+week,data=hotel_desk)
summary(mod1)
```

```
##
## Call:
## lm(formula = price ~ length_of_stay + floor + guest_id + week,
##     data = hotel_desk)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1231.90  -222.68   -20.89   196.13  2029.81
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -438.0280    30.3384  -14.44   <2e-16 ***
## length_of_stay   259.9485     5.9336   43.81   <2e-16 ***
## floor             53.6999     1.4565   36.87   <2e-16 ***
## guest_id           2.4640     0.1545   15.95   <2e-16 ***
## week           -2485.6225   154.1597  -16.12   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 371.8 on 2073 degrees of freedom
##   (17 observations deleted due to missingness)
## Multiple R-squared:  0.6354, Adjusted R-squared:  0.6347
## F-statistic: 903.4 on 4 and 2073 DF,  p-value: < 2.2e-16
```

```
sprintf("The r squared for the model is %f",summary(mod1)$r.squared)
```

```
## [1] "The r squared for the model is 0.635448"
```

Interpretation: The price depends on guest_id, floor, length_of_stay, week. The multiple r-squared value is 0.6354, which means that the model is explaining around 63.5% of the variation.

Question 4: Build a model to describe and predict lengths of stay.

```
ln1 = lm(length_of_stay~price+floor+guest_id+week+out_day_of_week,data= hotel_desk)
summary(ln1)
```

```
##
## Call:
## lm(formula = length_of_stay ~ price + floor + guest_id + week +
##     out_day_of_week, data = hotel_desk)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.0633 -0.4270 -0.1611  0.3452  2.9072
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          1.368e+00  6.367e-02  21.489  < 2e-16 ***
## price                8.002e-04  3.409e-05  23.469  < 2e-16 ***
## floor               -4.569e-02  3.114e-03 -14.671  < 2e-16 ***
## guest_id            -2.253e-03  2.815e-04  -8.006 1.96e-15 ***
## week                 2.271e+00  2.812e-01   8.077 1.11e-15 ***
## out_day_of_weekMon   1.072e+00  6.007e-02  17.849  < 2e-16 ***
## out_day_of_weekSat   1.712e-01  5.763e-02   2.971   0.0030 **
## out_day_of_weekSun   2.719e-01  5.435e-02   5.003 6.12e-07 ***
## out_day_of_weekThu -1.705e-01  8.330e-02  -2.047   0.0408 *
## out_day_of_weekTue   1.842e+00  7.009e-02  26.285  < 2e-16 ***
## out_day_of_weekWed   2.647e+00  6.684e-02  39.594  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6488 on 2067 degrees of freedom
##   (17 observations deleted due to missingness)
## Multiple R-squared:  0.7786, Adjusted R-squared:  0.7776
## F-statistic:   727 on 10 and 2067 DF,  p-value: < 2.2e-16
```

```
sprintf("The r squared for the model is %f",summary(ln1)$r.squared)
```

```
## [1] "The r squared for the model is 0.778633"
```

Interpretation: The length of stay depends on guest_id, floor, price, week and out_day_of_the_week (categorical). The multiple r-squared value is 0.7786328, which means that the model is explaining around 77.8% of the variation.

Question 5: Recommendations

- It is stated that the price of the stay is dependent on the airline prices. Therefore, to make better price predictions, airline prices should be given in the dataset.

- The dataset contains data only based on guests but staff must also be using elevators. Therefore, the use of elevators can be better tracked if data related to staff was also provided.
- Guest 1024 accessed floors other than the one in which his room was booked. He also tried using his key to unlock other doors. This is a security breach. The hotel must have a mechanism to track if the door key is used on other doors.
- There is huge variation in the prices of hotel rooms. It could be in the type or the standard of the rooms offered. For example: Some hotels have luxury, deluxe suites that are more expensive than other rooms. The information on why some rooms are more expensive than others is missing.
- Hotel prices tend to go up in certain seasons. For example: Tourists are more likely to book hotels in pleasant weather. Data on busy and idle seasons can also help in predictions.
- 14 of 17 missing observations come from Week 3 and Week 5. Therefore, the missing data is not MCAR (Missing Completely At Random). It is also noted that for most of missing data in Week 3, the price for the room booked for total stay was greater than 4000 dollars. The hotel should therefore look into why this data was not recorded.