Attaching package: ‘leaflet’

The following object is masked from ‘package:xts’:

addLegend

> library(broom)

>

>

> #connect to database

> domconnect <- dbConnect(RPostgreSQL::PostgreSQL(),

+ dbname = "domicile-reports",

+ host = "reporting-db.bookdomicile.com",

+ port = 5432,

+ user = "reporter",

+ password = "0Qo0m@%F2JCh")

>

> dbListTables(domconnect)

[1] "knex\_migrations" "knex\_migrations\_lock" "last\_run"

[4] "bookings"

> #download "bookings" table" from Dbase

> dombookings <- dbReadTable(domconnect, "bookings")

Warning message:

In postgresqlExecStatement(conn, statement, ...) :

RS-DBI driver warning: (unrecognized PostgreSQL field type uuid (id:2950) in column 17)

>

> #disconnect to dBase

> dbDisconnect(domconnect)

[1] TRUE

>

> dombookings <- dombookings %>%

+ mutate(check\_in\_date = ymd(check\_in\_date, tz = ""),

+ check\_out\_date = ymd(check\_out\_date, tz = ""),

+ days\_in\_advance = round(as.numeric(difftime(check\_in\_date, created\_at, tz = "", units = "days"))))

>

> dombookings %>% write\_csv("~/R\_files/Domicile/DomProject/DomData/bookings.csv")

>

> today <- ymd(cut(today(), "month"))

> monthstart <- ymd(paste(year(now()), "-", month(now()), "-01", sep = ""))

> ninetyseq <- seq.Date(monthstart, by = "day", length.out = 90)

#Color Palette

> DomColor <- c("#8B668B", "#6CA6CD", "#FF6347", "#A2CD5A", "#878787", "#CD853F", "#36648B", "#FFC125", "#fb9a99", "#53868B", "#8B795E")

> DomColor2 <- c("#FD3B36", "#6AA6E2", "#0F425D", "#888888", "#885EA8", "#7ED321", "#FF7F00", "#FFB90F", "#698B69", "#CDBA96", "#EEEE00")

> DomCol11 <- c("#885EA8", "#6AA6E2", "#FF6347", "#878787", "#CD853F", "#36648B", "#FFC125", "#FB9A99", "#589CA1FD", "#B89C76FE", "#A2CD5A")

> DomCol12 <- c("#8B668B", "#6CA6CD", "#FF6347", "#A2CD5A", "#878787", "#CD853F", "#36648B", "#FFC125", "#fb9a99", "#53868B", "#8B795E")

> DomColSource <- c("#8B668B", "#6CA6CD", "#FF6347", "#A2CD5A", "#878787", "#CD853F", "#36648B")

> #import, shape, and connect other tables.

> dombuildings <- read\_csv("~/R\_files/Domicile/DomProject/DomData/bldg.csv")

Parsed with column specification:

cols(

listing\_nickname = col\_character(),

building = col\_character(),

Bldg\_Name = col\_character(),

Address = col\_character(),

Latitude = col\_double(),

Longitude = col\_double()

)

> domneighborhoods <- read\_csv("~/R\_files/Domicile/DomProject/DomData/Data/Neighborhood.csv")

Parsed with column specification:

cols(

neighborhood = col\_character(),

Bldg\_Name = col\_character(),

building = col\_character(),

cohort = col\_character()

)

> BldgMaster <- dombuildings %>% left\_join(domneighborhoods, by = c("Bldg\_Name", "building"))

> #leftover from first iteration. There may be other DF's now attached to dommaster. preserve this until you make sure no bugs.

> dommaster <- dombookings %>% left\_join(dombuildings, by = "listing\_nickname") %>%

+ left\_join(domneighborhoods, by = c("building", "Bldg\_Name")) %>%

+ write\_csv("~/R\_files/Domicile/DomProject/DomData/master.csv")

save(dommaster, file = "~/R\_files/Domicile/DomProject/Domicile\_Metrics\_Shiny/dommaster.RData")

> save(dommaster, file = "~/R\_files/Domicile/DomProject/DomicileDashboardShiny/dommaster.RData")

> save(domneighborhoods, file = "~/R\_files/Domicile/DomProject/DomicileDashboardShiny/domneighborhoods.RData")

> ListingNicknames <- unique(dommaster$listing\_nickname)

> BuildingNames <- unique(dombuildings$Bldg\_Name)

>

> #Base DF. Raw booking data filted for reserved/confirmed rooms. Calcs' for number of days and nights. Adds a (Comp day, which aids calcs in later code to lengthen

> #bookings over the individual days booked. Cleans the Source column.

> DomBookings <- dommaster %>% filter(status %in% c("confirmed", "reserved") & host\_payout > 0 & (!is.na(check\_in\_date) | !is.na(check\_out\_date))) %>%

+ filter(!(status == "reserved" & check\_in\_date < ymd(today(), tz = "") )) %>%

+ mutate(comp\_out\_date = check\_out\_date - days(1),

+ num\_nights = as.numeric(difftime(check\_out\_date, check\_in\_date, "days")),

+ ADR = host\_payout / num\_nights,

+ source = str\_to\_lower(source),

+ source = case\_when(

+ str\_detect(source, "booking.com") ~ "booking.com",

+ str\_detect(source, "direct") ~ "direct",

+ str\_detect(source, "manual") ~ "manual",

+ str\_detect(source, "homeaway") ~ "HomeAway",

+ str\_detect(source, "expedia") ~ "Expedia",

+ str\_detect(source, "airbnb") ~ "Airbnb",

+ TRUE ~ "other")

+ ) %>% arrange(Bldg\_Name, listing\_nickname) %>%

+ select(confirmation\_code, source, status, Bldg\_Name, listing\_nickname, check\_in\_date,

+ comp\_out\_date, check\_out\_date, ADR, num\_nights, host\_payout, created\_at, days\_in\_advance) %>%

+ write\_csv("~/R\_files/Domicile/DomProject/DomData/DomBookings28.csv")

>

> save(DomBookings, file = "~/R\_files/Domicile/DomProject/DomicileDashboardShiny/DomBookings.RData")

Base table used in graphs for dashboard.

> DomVel <- DomBookings %>% left\_join(domneighborhoods, by = "Bldg\_Name") %>%

+ mutate(rev\_mo = ymd(cut(check\_in\_date, breaks = "month"), tz = ""),

+ weekday = factor(weekdays(check\_in\_date, abbreviate = T), levels = c("Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat")))

>

> #script to create a new DF called "bookedDF". Spreads the booking and booking revenue into individual days

> bookedDF <- data\_frame(conf = "abce4x-yyxtz3", listing\_nickname = "dummy",

+ booked = as.Date(ymd("2018-01-31", tz = "")), rev = 0,

+ status = "status", source = "source")

>

> for(i in 1:nrow(DomBookings)) {

+ listing\_nickname <- DomBookings$listing\_nickname[i]

+ booked <- as.Date(seq(DomBookings$check\_in\_date[i], DomBookings$comp\_out\_date[i], by = "day"))

+ rev <- DomBookings$ADR[i]

+ conf <- DomBookings$confirmation\_code[i]

+ source <- DomBookings$source[i]

+ status <- DomBookings$status[i]

+ z <- cbind.data.frame(conf, listing\_nickname, booked, rev, status, source)

+ bookedDF <- rbind(bookedDF, z)

+ }

bookedDF %>% write\_csv("~/R\_files/Domicile/DomProject/DomData/bookedDF.csv")

>

> #DomBookings$host\_payout[i] / as.numeric(difftime(DomBookings$check\_out\_date[i], DomBookings$check\_in\_date[i], units = "days"))

> BookedDF <- bookedDF %>% mutate(year = year(booked),

+ month = month(booked),

+ BT = 1,

+ rev\_mo = as.Date(ymd(cut(booked, "month"), tz = ""))) %>%

+ left\_join(dombuildings, by = "listing\_nickname") %>% left\_join(domneighborhoods, by = c("Bldg\_Name", "building")) %>%

+ write\_csv("~/R\_files/Domicile/DomProject/DomData/BookingMastertest1.csv")

>

> #Read in the Launch Dates and clean to remove duplicates. Create a new "long" data frame by sequencing by day between launch and end dates and adding a "Rev\_Mo" column. Work around the

> #Lubridate issue of month increments not putting the last rev\_mo if there wasn't a full month before the end date by changing both to first of month.

> DLT<- read\_csv("~/R\_files/Domicile/DomProject/DomData/Data/LaunchDateNew.csv")

Parsed with column specification:

cleanlaunch <- DLT[!(DLT$launch\_date =="1/15/18" & DLT$end\_date =="4/30/18" & DLT$building == "marina"), ]

> head(cleanlaunch)

CL <- cleanlaunch %>% mutate(launch\_date = mdy(launch\_date, tz = ""),

+ pdlaunch\_date = ymd(cut(launch\_date, "month"), tz = ""),

+ end\_date = mdy(end\_date, tz = ""),

+ pdend\_date = ymd(cut(end\_date, "month"), tz = "")) %>%

+ write\_csv("~/R\_files/Domicile/DomProject/DomData/clean.csv")

> glimpse(CL)

roomsDF <- data\_frame(listing\_nickname = "test",

+ rev\_mo = as.Date(ymd("2018-01-31", tz = "")),

+ launch\_date = ymd("2018-01-31", tz = ""),

+ end\_date = ymd("2018-01-31", tz = ""),

+ building = "testdummy")

>

> for(i in 1:nrow(CL)) {

+ listing\_nickname <- CL$listing\_nickname[i]

+ rev\_mo <- as.Date(seq.POSIXt(CL$pdlaunch\_date[i], CL$pdend\_date[i], by = "month"))

+ launch\_date <- CL$launch\_date[i]

+ end\_date <- CL$end\_date[i]

+ building <- CL$building[i]

+ z <- cbind.data.frame(listing\_nickname, rev\_mo, launch\_date, end\_date, building)

+ roomsDF <- rbind(roomsDF, z)

+ }

>

> roomsDF %>% write\_csv("~/R\_files/Domicile/DomProject/DomData/roomsDF.csv")

>

> #### New CODE TO construction a by-day version of the booking master file for analysis by day and week. Foundation for code below.

> roomsDFdays <- data\_frame(listing\_nickname = "test",

+ booked = as.Date(ymd("2018-01-31", tz = "")),

+ launch\_date = ymd("2018-01-31", tz = ""),

+ end\_date = ymd("2018-01-31", tz = ""),

+ building = "testdummy",

+ AT = 1)

>

> roomsDFdays

for(i in 1:nrow(CL)) {

+ listing\_nickname <- CL$listing\_nickname[i]

+ booked <- as.Date(seq.POSIXt(CL$launch\_date[i], CL$end\_date[i], by = "day"))

+ launch\_date <- CL$launch\_date[i]

+ end\_date <- CL$end\_date[i]

+ building <- CL$building[i]

+ AT = 1

+ z <- cbind.data.frame(listing\_nickname, booked, launch\_date, end\_date, building, AT)

+ roomsDFdays <- rbind(roomsDFdays, z)

+ }

>

> RoomsDFdays <- roomsDFdays %>% mutate(rev\_mo = as.Date(ymd(cut(booked, breaks = "months"), tz = ""))) %>%

+ write\_csv("~/R\_files/Domicile/DomProject/DomData/roomsDFdays.csv")

>

> glimpse(RoomsDFdays)

#New base dataframe to use for analysis. Data is displayed on a daily basis rather than monthly. Other dependencies exist for this data.

> DaysDF <- RoomsDFdays %>% full\_join(BookedDF, by = c("building", "listing\_nickname", "rev\_mo", "booked")) %>%

+ mutate(booked = ymd(booked, tz = ""),

+ rev\_mo = ymd(rev\_mo, tz = ""),

+ BT = case\_when(is.na(BT) ~ 0,

+ TRUE ~ BT),

+ AT = case\_when(is.na(AT) ~ 0,

+ TRUE ~ AT),

+ week = epiweek(booked), weekname = cut.POSIXt(booked, "week", start.on.monday = F),

+ weekdate = ymd\_hms(as.character(weekname), tz = ""))

>

> ymd\_hms(as.character(DaysDF$weekname), tz = "")

DaysDF %>%

+ write\_csv("~/R\_files/Domicile/DomProject/DomData/weektroubleroot.csv")

>

> DaysDownload <- DaysDF %>% select(neighborhood, cohort, Bldg\_Name,

+ listing\_nickname, conf, status, rev\_mo, weekname,

+ booked, AT, BT, rev, launch\_date, end\_date)

> #WeekOcc

> #calculates the number of avail days and booked days by unit, then adds back the information about cohort.

> #becuase of the way the data is strutured, if you add cohort to initial grouping, then the avail days always

> # = booked days, which is incorrect. This method calculates correctly.

> WeekOcc <- DaysDF %>% group\_by(listing\_nickname, year = year(weekdate), week, weekdate) %>%

+ summarize(avail\_days = sum(AT),

+ booked\_days = sum(BT),

+ occ\_rate = case\_when(avail\_days > 0 ~ booked\_days / avail\_days,

+ TRUE ~ 0),

+ Rev = sum(rev, na.rm = T),

+ ADR = case\_when(booked\_days > 0 ~ sum(rev, na.rm = T) / booked\_days,

+ TRUE ~ 0),

+ RevPar = occ\_rate \* ADR) %>% ungroup() %>%

+ left\_join(dombuildings, by = "listing\_nickname") %>%

+ left\_join(domneighborhoods, by = c("Bldg\_Name", "building"))

>

> BldgLL <- BldgMaster %>% group\_by(Bldg\_Name) %>%

+ summarize(Latitude = unique(Latitude), Longitude = unique(Longitude))

>

> ###Data for Leaflet Map in App

> MonthDet <- DaysDF %>% group\_by(listing\_nickname, rev\_mo) %>%

+ summarize(avail\_days = sum(AT),

+ booked\_days = sum(BT),

+ occ\_rate = case\_when(avail\_days > 0 ~ booked\_days / avail\_days,

+ TRUE ~ 0),

+ Rev = sum(rev, na.rm = T),

+ ADR = case\_when(booked\_days > 0 ~ sum(rev, na.rm = T) / booked\_days,

+ TRUE ~ 0),

+ RevPar = occ\_rate \* ADR) %>% ungroup() %>%

+ left\_join(dombuildings, by = "listing\_nickname") %>%

+ left\_join(domneighborhoods, by = c("Bldg\_Name", "building")) %>%

+ group\_by(cohort, Bldg\_Name, rev\_mo) %>% summarize(Rooms = length(unique(listing\_nickname)),

+ avail\_days = sum(avail\_days),

+ booked\_days = sum(booked\_days),

+ Rev = sum(Rev),

ADR = Rev / booked\_days,

+ Occ = booked\_days / avail\_days,

+ RevPar = ADR \* Occ) %>% ungroup() %>%

+ left\_join(BldgLL, by = "Bldg\_Name")

> save(MonthDet, file = "~/R\_files/Domicile/DomProject/DomicileDashboardShiny/MonthDet.RData")

>

> ##Data for graphs attached to LeafLet Map in App

> MonthAll <- DaysDF %>% group\_by(listing\_nickname, rev\_mo) %>%

+ summarize(avail\_days = sum(AT),

+ booked\_days = sum(BT),

+ occ\_rate = case\_when(avail\_days > 0 ~ booked\_days / avail\_days,

+ TRUE ~ 0),

+ Rev = sum(rev, na.rm = T),

+ ADR = case\_when(booked\_days > 0 ~ sum(rev, na.rm = T) / booked\_days,

+ TRUE ~ 0),

+ RevPar = occ\_rate \* ADR) %>% ungroup() %>%

+ left\_join(dombuildings, by = "listing\_nickname") %>%

+ left\_join(domneighborhoods, by = c("Bldg\_Name", "building")) %>%

+ group\_by(rev\_mo) %>% summarize(Rooms = length(unique(listing\_nickname)),

+ avail\_days = sum(avail\_days),

+ booked\_days = sum(booked\_days),

+ Rev = sum(Rev),

+ ADR = Rev / booked\_days,

+ Occ = booked\_days / avail\_days,

+ RevPar = ADR \* Occ) %>% ungroup()

> save(MonthAll, file = "~/R\_files/Domicile/DomProject/DomicileDashboardShiny/MonthAll.RData")

##RPOcc doesn't have the correct numbers for avail days due to source grouping.

> RPOcc <- DaysDF %>% group\_by(listing\_nickname, source, year = year(weekdate), week, weekdate) %>%

+ summarize(avail\_days = sum(AT),

+ booked\_days = sum(BT),

+ occ\_rate = case\_when(avail\_days > 0 ~ booked\_days / avail\_days,

+ TRUE ~ 0),

+ Rev = sum(rev, na.rm = T),

+ ADR = case\_when(booked\_days > 0 ~ sum(rev, na.rm = T) / booked\_days,

+ TRUE ~ 0),

+ RevPar = occ\_rate \* ADR) %>% ungroup() %>%

+ left\_join(dombuildings, by = "listing\_nickname") %>%

+ left\_join(domneighborhoods, by = c("Bldg\_Name", "building"))

>

>

> #dataframe used to calculate the weekly rolled up summary by cohort.

> WeekSum <- WeekOcc %>% group\_by(cohort, year = year(weekdate), weekdate) %>%

+ summarise(avail = sum(avail\_days),

+ booked = sum(booked\_days),

+ occ = booked / avail,

+ TotRev = sum(Rev, na.rm = T),

+ ADR = case\_when(booked > 0 ~ sum(Rev, na.rm = T) / booked,

+ TRUE ~ 0),

+ RevPar = ADR \* occ,

+ occhar = percent(occ),

+ adrchar = dollar(round(ADR)),

+ rparchar = dollar(round(RevPar))) %>% ungroup()

>

> WS <- WeekSum %>% filter(year(weekdate) == 2018, month(weekdate) >= month(now()))

> WT <- WS %>% group\_by(year, weekdate) %>%

+ summarise(avail = sum(avail),

+ booked = sum(booked),

+ occ = booked / avail,

+ ADR = mean(ADR),

+ occhar = percent(occ),

+ adrchar = dollar(ADR)) %>% mutate(cohort = "Total") %>% ungroup() %>%

+ select(cohort, year, weekdate, avail, booked, occ, ADR, occhar, adrchar)

> WST <- bind\_rows(WS, WT)

>

> WeekSum %>% write\_csv("~/R\_files/Domicile/DomProject/DomData/weeksum.csv")

> glimpse(WeekOcc)

save(WeekOcc, file = "~/R\_files/Domicile/DomProject/DomicileDashboardShiny/WeekOcc.RData")

> save(RPOcc, file = "~/R\_files/Domicile/DomProject/DomicileDashboardShiny/RPOcc.RData")

> save(WeekSum, file = "~/R\_files/Domicile/DomProject/DomicileDashboardShiny/WeekSum.RData")

> WeekOcc %>% write\_csv("~/R\_files/Domicile/DomProject/DomData/weektroubleT.csv")

>

> #At a Month level. Join the Launch Dates DF to the BookingMaster DF and calculate the days available within the month. Adds weekday as a factor.

> BookingMaster <- BookedDF %>% full\_join(roomsDF, by = c("listing\_nickname", "building", "rev\_mo")) %>%

+ mutate(weekday = factor(weekdays(booked, abbreviate = T), levels = c("Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat")) ,

+ avail\_days = case\_when(!is.na(booked) & month(launch\_date) == month(booked) & year(launch\_date) == year(booked) ~

+ abs(as.numeric(difftime(rollback(ceiling\_date(booked, unit = "month", change\_on\_boundary = T)), launch\_date, "days"))) + 1,

+ !is.na(booked) & month(end\_date) == month(booked) & year(end\_date) == year(booked) ~

+ abs(as.numeric(days\_in\_month(rev\_mo)) -

+ as.numeric(difftime(rollback(ceiling\_date(booked, unit = "month", change\_on\_boundary = T)), end\_date, "days"))) + 1,

+ is.na(booked) & month(launch\_date) == month(rev\_mo) & year(launch\_date) == year(rev\_mo) ~

+ abs(as.numeric(difftime(rollback(ceiling\_date(launch\_date, unit = "month", change\_on\_boundary = T)), launch\_date, "days"))) + 1,

+ is.na(booked) & month(end\_date) == month(rev\_mo) & year(end\_date) == year(rev\_mo) ~

+ abs(as.numeric(days\_in\_month(rev\_mo)) - as.numeric(difftime(rollback(ceiling\_date(end\_date, unit = "month", change\_on\_boundary = T)), end\_date, "days"))) + 1,

+ TRUE ~ as.numeric(days\_in\_month(rev\_mo)))) %>% write\_csv("~/R\_files/Domicile/DomProject/DomData/bookingmaster.csv")

>

>

> save(BookingMaster, file = "~/R\_files/Domicile/DomProject/Domicile\_Metrics\_Shiny/BookingMaster.RData")

> save(BookingMaster, file = "~/R\_files/Domicile/DomProject/DomicileDashboardShiny/BookingMaster.RData")

> str(BookingMaster)

#Occupancy Rates by Month by Room CODE AND CHART created 8/29/2018

> DomSummary <- BookingMaster %>% group\_by(Bldg\_Name, listing\_nickname, year, rev\_mo) %>%

+ summarise(days\_booked = sum(BT),

+ days\_avail = max(avail\_days),

+ occ\_rate = days\_booked / days\_avail,

+ Avg\_Rev = sum(rev) / sum(BT),

+ Rev\_Par = Avg\_Rev \* occ\_rate) %>% ungroup() %>%

+ write\_csv("~/R\_files/Domicile/DomProject/DomData/DomSummary.csv")

> save(DomSummary, file = "~/R\_files/Domicile/DomProject/Domicile\_Metrics\_Shiny/DomSummary.RData")

> save(DomSummary, file = "~/R\_files/Domicile/DomProject/DomicileDashboardShiny/DomSummary.RData")

> nrow(DomSummary)

[1] 2534

RevOcc <- DomSummary %>% mutate(rev\_mo = ymd(rev\_mo, tz = "")) %>%

+ group\_by(Bldg\_Name, year, rev\_mo) %>%

+ summarize(Booked\_Days = sum(days\_booked),

+ Available\_Days = sum(days\_avail),

+ Occupancy\_Rate = Booked\_Days / Available\_Days,

+ Average\_Rev\_Night = round(mean(Avg\_Rev)),

+ Rev\_PAR = round(mean(Rev\_Par))) %>% ungroup() %>%

+ select(Bldg\_Name, year, rev\_mo, Occupancy\_Rate, Average\_Rev\_Night) %>%

+ gather(key = Metric, value = value, 4:5)