```r
## This is a sequenced set of routines to be followed for Fleet Modelling Purposes
## This version has been amended to help with code surgery Coffee and Coding - 26th of June 2019
## Import CAA movement data, IBA fleet  and other lookup data - which is cleaned, merged and then processed (some steps are
ran multiple times)to provide FMM parameters
## There are review or QA steps at several points - please check working directory for spreadsheets that are produced during
execution
## For details on suitable use of the underlying movements data, please speak to the Stats team
## Dharmender Tathgur AMA January 2019
## This code was amended after MH carried out an (internal) QA process in April 2019

# Step 0 - Global declarations, file locations and libraries   ----  Please double check locations and input year on a
regular basis
## worksheets may have been added, amended or removed from the lookup files so be aware

## Global definition lists  ----

OECD_countries = c("Australia","Austria","Belgium","Canada","Chile","Czech
Republic","Denmark","Estonia","Finland","France","Germany","Greece","Hungary","Iceland","Ireland",
                   "Israel","Italy","Japan","Korea, Republic of","Luxembourg","Mexico","Netherlands","New
                   Zealand","Norway","Poland","Portugal","Slovakia","Slovenia","Spain",
                   "Sweden","Switzerland","Turkey","United Kingdom","United States")

UK_modelled_airports = c(461:487,491:492)                    # For further detail see aviation model documentation - 599 is a
NON MODELLED UK AIRPORT


# Working directory - not necessarily the location of datasets or lookup files
setwd("<INSERT DIR HERE>")

#Input year here
year <- c("<INSERT YEAR HERE>")

#Location of CAA ATM movement file here for the input year - this could be Server pipe in the future e.g. SQL - please amend
accordingly
ATM_file <- "<INSERT FILE HERE>"

#Input location of aviation model lookup files
IATA_ICAO_Aircraft_codes_file <-  "<INSERT FILE HERE>"
WorldZoneCon_file <-  "<INSERT FILE HERE>"

#Location of Aircraft inventory by registration code as of the beginning and end of input year to allow maximum coverage
#Download instructions and the order of columns are in the following spreadsheets
#Aircraft registrations change across a year and the following method is able to catch as much as it can
#Aircraft_inventory_file_missing is for registrations that are either not recorded by IBA e.g. certain airlines and
helicopters or registrations that change multiple times in the input year
Aircraft_inventory_file_Jan <- "<INSERT FILE HERE>"
Aircraft_inventory_file_Dec <- "<INSERT FILE HERE>"
Aircraft_inventory_file_Missing <- "<INSERT FILE HERE>"

# Location of Aircraft Backlog file as of end of input year again for maximum coverage
Aircraft_order_file <- "<INSERT FILE HERE>"

#Import following libraries to carry out relevant functions
library(tidyverse)
library(data.table)
library(openxlsx)
library(janitor)
library(readxl)

# Step 1 - Import files and lookups  into R Environment -----------------------------------

##  Step 1.1 Import aicraft inventory files downloaded from IBA fleets into memory and the missing data manually input
## Please refer to the procedure document to find out the description of the IBA datasets below and how to extract it
CORRECTLY

Aircraft_inventory_Jan <- read_excel(Aircraft_inventory_file_Jan, sheet = "Report",
                                 col_types = c("text", "text", "text",
                                               "text", "text", "text", "text", "numeric",
                                               "text", "text", "text", "text", "text",
                                               "numeric", "numeric", "text"),
                                 na = "NA", skip = 9)

Aircraft_inventory_Dec <- read_excel(Aircraft_inventory_file_Dec, sheet = "Report",
                                        col_types = c("text", "text", "text",
                                                      "text", "text", "text", "text", "numeric",
                                                      "text", "text", "text", "text", "text",
                                                      "numeric", "numeric", "text"),
                                        na = "NA", skip = 9)

Aircraft_inventory_Missing <- read_excel(Aircraft_inventory_file_Missing, sheet = "Report",
                                      col_types = c("text", "text", "text",
                                                    "text", "text", "text", "text", "numeric",
                                                    "text", "text", "text", "text", "text",
                                                    "numeric", "numeric", "text"),
                                      na = "NA", skip = 9)

Aircraft_order <- read_excel(Aircraft_order_file, sheet = "Report", na = "", skip = 9)


## Step 1.2 Import CAA movements file into memory
## Read in the (caa) ATM data correctly formatted ~ Please see metadata in the source directory
## The following method should also apply to all ATM files from CAA - for all years - Hence the temp suffix
## In case of any discrepancy or error/warning thrown, Please speak to the stats team

ATM_temp <- readr::read_csv(ATM_file,
                                          col_types = cols(
                                            year = col_integer()
                                            ,month = col_factor(levels = c("Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul",
                                            "Aug", "Sep", "Oct", "Nov", "Dec"), ordered = TRUE)
                                            ,month_no = col_integer()
                                            ,reporting_airport = col_character()
                                            ,reporting_airport_code = col_character()
                                            ,lastnext_airport = col_character()
                                            ,lastnext_country = col_character()
                                            ,origdest_airport = col_character()
```

```r
                                           ,origdest_country = col_character()
                                           ,aircraft_name = col_character()
                                           ,aircraft_reg = col_character()
                                           ,aoc_holder_name = col_character()
                                           ,consent_to_publish = col_character()
                                           ,service_type = col_character()
                                           ,operation_type = col_character()
                                           ,direction_type = col_character()
                                           ,atm_date = col_date(format = "%Y-%m-%d")
                                           ,weekday1 = col_factor(levels = c("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"), ordered = TRUE)
                                           ,weekday2 = col_factor(levels = c("Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"), ordered = TRUE)
                                           ,time = col_time(format = "%H:%M:%S")
                                           ,time_hour = col_integer()
                                           ,time_band = col_character()
                                           ,atm_count = col_integer()
                                           ,lastnext_sectordistance_km = col_integer()
                                           ,available_seats = col_integer()
                                           ,pax_terminal_dom = col_integer()
                                           ,pax_terminal_int = col_integer()
                                           ,pax_terminal_total = col_integer()
                                           ,pax_transit_total = col_integer()
                                           ,freight_weight_kg = col_integer()
                                           ,mail_weight_kg = col_integer()
                                           ,cargo_weight_kg = col_integer()
                                           ,freight_weight_tonnes = col_double()
                                           ,mail_weight_tonnes = col_double()
                                           ,cargo_weight_tonnes = col_double()
                                         ),
                            trim_ws=TRUE
)

## Step 1.3 Import aviation model lookup files into memory
SPASMZone <- read_excel(WorldZoneCon_file, sheet = "WorldZoneCon", range = cell_cols("A:B"))
CAAICAOIATA <- read_excel(IATA_ICAO_Aircraft_codes_file,sheet = "CAAicaoIATA",range = cell_cols("A:F"))

# Step 1.4 Clean - Remove records with blank or NA values - these incomplete, mostly legacy, records are of no use
# Remove records with blank or NA values for Registration in aircraft inventory
Aircraft_inventory_Jan <- Aircraft_inventory_Jan[!is.na(Aircraft_inventory_Jan$Registration),]
Aircraft_inventory_Dec <- Aircraft_inventory_Dec[!is.na(Aircraft_inventory_Dec$Registration),]
Aircraft_inventory_Missing <- Aircraft_inventory_Missing[!is.na(Aircraft_inventory_Missing$Registration),]

# Step 1.5 Rename specific cols in lookup files to prevent duplication and ease understanding
names(SPASMZone)[names(SPASMZone) == "Name"] <- "AP_name"
names(SPASMZone)[names(SPASMZone) == "SPASMZONE"] <- "SPASM_zone"

# Step 1.6 Text cleaning routines for consistency with CAA movements file
# the following commands ensure that ALL values in specific columns are CAPS & without trailing/leading spaces.
Aircraft_inventory_Jan$Registration <- trimws(toupper(c(Aircraft_inventory_Jan$Registration)))
Aircraft_inventory_Dec$Registration <- trimws(toupper(c(Aircraft_inventory_Dec$Registration)))
Aircraft_inventory_Missing$Registration <- trimws(toupper(c(Aircraft_inventory_Missing$Registration)))

ATM_temp$lastnext_airport <- trimws(toupper(c(ATM_temp$lastnext_airport)))
ATM_temp$origdest_airport <- trimws(toupper(c(ATM_temp$origdest_airport)))
ATM_temp$aircraft_name <- trimws(toupper(c(ATM_temp$aircraft_name)))
ATM_temp$aircraft_reg <- trimws(toupper(c(ATM_temp$aircraft_reg)))

SPASMZone$AP_name <- toupper(SPASMZone$AP_name)

# Remove hyphens and pluses from aircraft registrations in both to provide better merge later on

Aircraft_inventory_Jan$Registration <- gsub("-", "", Aircraft_inventory_Jan$Registration)
Aircraft_inventory_Jan$Registration <- gsub("\\+", "", Aircraft_inventory_Jan$Registration)
Aircraft_inventory_Dec$Registration <- gsub("-", "", Aircraft_inventory_Dec$Registration)
Aircraft_inventory_Dec$Registration <- gsub("\\+", "", Aircraft_inventory_Dec$Registration)
Aircraft_inventory_Missing$Registration <- gsub("-", "", Aircraft_inventory_Missing$Registration)
Aircraft_inventory_Missing$Registration <- gsub("\\+", "", Aircraft_inventory_Missing$Registration)

ATM_temp$aircraft_reg <- gsub("-", "", ATM_temp$aircraft_reg)

# Step 1.7 Harmonising IBA aircraft inventory - Create a new col called Age_midYear
# 1.7.1 Add half a year to January and subtract half year to Dec search ages to represent mid year age profile

Aircraft_inventory_Jan$Age_midYear <- Aircraft_inventory_Jan$`Search Age` + 0.5
Aircraft_inventory_Dec$Age_midYear <- Aircraft_inventory_Dec$`Search Age` - 0.5
Aircraft_inventory_Missing$Age_midYear <- Aircraft_inventory_Missing$`Search Age`


# 1.7.2 Consolidate Dec, Jan and Missing. Missing given the highest priority followed by Dec and then Jan
# rename age col appropriately and round to the nearest integer

Aircraft_inventory <- bind_rows(Aircraft_inventory_Dec, Aircraft_inventory_Jan[!(Aircraft_inventory_Jan$Registration
                                                                             %in% Aircraft_inventory_Dec$Registration),])
Aircraft_inventory <- bind_rows(Aircraft_inventory_Missing, Aircraft_inventory[!(Aircraft_inventory$Registration
                                                                             %in% Aircraft_inventory_Missing$Registration),])
Aircraft_inventory$Age_midYear <- round(Aircraft_inventory$Age_midYear,0)

#1.7.3 The resulting harmonised inventory may have a few duplicate values for Registration which means the following order of importance should be carried out for distinct values

order_of_importance <- c("Active", "On Order", "Retained","Parted Out","Stored","Destroyed",  "Retired","Parked","Damaged")

Aircraft_inventory <- Aircraft_inventory %>%
  mutate(Status = factor(Status, order_of_importance)) %>%
  arrange(Registration, Status) %>%
  distinct(Registration, .keep_all = TRUE)

# Step 1.8 Set up factors - only for columns that are of cateogorical nature - speeds up processing later on and for producing plots
Aircraft_inventory$Registration <- as.factor(Aircraft_inventory$Registration)
Aircraft_inventory$`Aircraft Class` <- as.factor(Aircraft_inventory$`Aircraft Class`)
Aircraft_inventory$`Aircraft Family`<- as.factor(Aircraft_inventory$`Aircraft Family`)
Aircraft_inventory$`Aircraft Manufacturer`<-as.factor(Aircraft_inventory$`Aircraft Manufacturer`)
Aircraft_inventory$`Aircraft Series`<-as.factor(Aircraft_inventory$`Aircraft Series`)
```

```r
Aircraft_inventory$`Aircraft Type` <- as.factor(Aircraft_inventory$`Aircraft Type`)
Aircraft_inventory$`Operator Country` <-as.factor(Aircraft_inventory$`Operator Country`)
Aircraft_inventory$`Operator Region` <- as.factor(Aircraft_inventory$`Operator Region`)
Aircraft_inventory$`Operator Subregion`<- as.factor(Aircraft_inventory$`Operator Subregion`)
Aircraft_inventory$Operator<-as.factor(Aircraft_inventory$Operator)
Aircraft_inventory$Status<-as.factor(Aircraft_inventory$Status)

Aircraft_order$`Aircraft Family` <- as.factor(Aircraft_order$`Aircraft Family`)

ATM_temp$month <- as.factor(ATM_temp$month)
ATM_temp$month_no <- as.factor(ATM_temp$month_no)
ATM_temp$reporting_airport <- as.factor(ATM_temp$reporting_airport)
ATM_temp$reporting_airport_code <- as.factor(ATM_temp$reporting_airport_code)
ATM_temp$lastnext_airport <- as.factor(ATM_temp$lastnext_airport)
ATM_temp$lastnext_country <- as.factor(ATM_temp$lastnext_country)
ATM_temp$origdest_airport <- as.factor(ATM_temp$origdest_airport)
ATM_temp$origdest_country <- as.factor(ATM_temp$origdest_country)
ATM_temp$aoc_holder_name <- as.factor(ATM_temp$aoc_holder_name)
ATM_temp$service_type <- as.factor(ATM_temp$service_type)
ATM_temp$operation_type <- as.factor(ATM_temp$operation_type)
ATM_temp$direction_type <- as.factor(ATM_temp$direction_type)
ATM_temp$weekday1 <- as.factor(ATM_temp$weekday1)
ATM_temp$weekday2 <- as.factor(ATM_temp$weekday2)
ATM_temp$time_hour <- as.factor(ATM_temp$time_hour)
ATM_temp$aircraft_name <- as.factor(ATM_temp$aircraft_name )
ATM_temp$aircraft_reg <- as.factor(ATM_temp$aircraft_reg)
ATM_temp$consent_to_publish <- as.factor(ATM_temp$consent_to_publish)                           #All these flights form part
of our analysis


CAAICAOIATA$CAA_Acf_Name <- as.factor(CAAICAOIATA$CAA_Acf_Name)
CAAICAOIATA$`Aircraft ICAO Code` <- as.factor(CAAICAOIATA$`Aircraft ICAO Code`)
CAAICAOIATA$IATA <- as.factor(CAAICAOIATA$IATA)
CAAICAOIATA$Fmm_Haul <- as.factor(CAAICAOIATA$Fmm_Haul)
CAAICAOIATA$Aircraft_Manufacturer <- as.factor(CAAICAOIATA$Aircraft_Manufacturer)
CAAICAOIATA$In_Production <- as.factor(CAAICAOIATA$In_Production)

# Step 1.9  Remove any temporary files or locations

rm(Aircraft_inventory_file_Missing,Aircraft_inventory_file_Jan,Aircraft_inventory_file_Dec, Aircraft_order_file)




# Step 2 - Pre Processing Quality Assurance -------------------------------
## Relevant data from input files in previous step are consolidated into dataframes and findings are saved in single .xlsx
file for review

#2.1.1 QA211 - High level integrity check  - Get number of records and variables from CAA ATM movements file
no_of_records <- nrow(ATM_temp)
no_of_cols <- ncol(ATM_temp)
pre_proc_row_col_total <- rbind(no_of_cols,no_of_records)
colnames(pre_proc_row_col_total) <- "no_obs"

# 2.1.2 QA212 - Aggregating service_type and operation_type from CAA ATM movements file
pre_proc_QA_serv_oper <- ATM_temp %>%
  group_by(service_type,operation_type)  %>%
  summarise("no_obs" = n()) %>%
  arrange(desc(no_obs))  %>%
  adorn_totals(where = "row")

# 2.1.3 QA213 - Aggregating aircraft_name from CAA ATM movements file
pre_proc_QA_aircraft_name <- ATM_temp %>%
  group_by(aircraft_name) %>%
  summarise("no_obs"=n(),"max_distance_flown" = round(quantile(lastnext_sectordistance_km,0.95),0)) %>%
  mutate(Haul = ifelse(max_distance_flown<3750, "Short", "Both")) %>%
  arrange(desc(no_obs))  %>%
  adorn_totals(where = "row")

# 2.1.4 QA214 - Aggregating year, month and month_no from CAA ATM movements file
pre_proc_QA_year_month <- ATM_temp %>%
  group_by(year,month, month_no)  %>%
  summarise("no_obs" = n()) %>%
  arrange(desc(no_obs))  %>%
  adorn_totals(where = "row")

# 2.1.5 QA215 - Aggregating reporting_airport_code, reporting_airport and direction_type from CAA ATM movements file
pre_proc_QA_reporting_airport_direction <- ATM_temp %>%
  group_by(reporting_airport_code, reporting_airport, direction_type)  %>%
  summarise("no_obs" = n()) %>%
  arrange(desc(no_obs))  %>%
  adorn_totals(where = "row")

# 2.1.6 QA216 - Aggregating aircraft registrations with available seats from CAA ATM movements file includes zero
available_seats and UNKNOWN reg
# Multiple occurances of aircraft_reg AND aircraft_name are expected especially since available_seats are recorded
differently either by airports or operations
pre_proc_QA_reg_name_seat  <- ATM_temp %>%
  group_by(aircraft_reg,  aircraft_name, available_seats)  %>%
  summarise("no_obs" = n()) %>%
  arrange(desc(no_obs))  %>%
  adorn_totals(where = "row")

# 2.1.7 QA217- Aggregating aircraft reg AND aircraft_name into unique record combinations summarised by seat info from CAA
ATM movements file
pre_proc_QA_reg_name_seat_analysis  <- ATM_temp %>%
  filter (available_seats>0 & aircraft_reg != "UNKNOWN") %>%
  group_by(aircraft_reg, aircraft_name)  %>%
  summarise("min_seat" = min(available_seats),
            "max_seat" = max(available_seats),
            "gap_seat_values" = max_seat - min_seat ,
            "mean_seat" = round(mean(available_seats),0),
            "no_obs" = n()) %>%
  arrange(desc(gap_seat_values))  %>%
  adorn_totals(where = "row")
```

```r


#2.1.8 QA218 Unique aircraft_reg, aircraft name(s) associated with that reg and summary of seat quantiles from CAA ATM movements file
#importance assumed to be correlated to difference quartile)

pre_proc_QA_unique_reg_name_seat_quartile_analysis <-  ATM_temp %>%
  filter (available_seats>0 & aircraft_reg != "UNKNOWN") %>%
  group_by(aircraft_reg)  %>%
  summarise( "lower quartile" = quantile(available_seats,0.25),
             "upper quartile" = quantile(available_seats,0.75),
             "difference quartile" = `upper quartile` - `lower quartile`,
             "no_obs" = n(),
             "order variable - product" = ifelse(`difference quartile` > 20  |   `no_obs` > 1 , `difference quartile`*`no_obs`, 0 ),
             "aircraft names" = paste(sort(unique(aircraft_name)), collapse=" , "),
             "no aircraft names" = n_distinct(aircraft_name)
             ) %>%
  arrange(desc(`no aircraft names`))  %>%
  adorn_totals(where = "row")

#2.2.1 QA221- Fleet by operator from harmonised IBA Aircraft inventory
pre_proc_fleet_by_oper <- Aircraft_inventory  %>%
  filter(Status == "Active" ) %>%
  group_by(Operator)  %>%
  summarise( "no_obs" = n()) %>%
  arrange(desc(no_obs))  %>%
  adorn_totals(where = "row")

#2.2.2 QA222- Fleet by type from harmonised IBA Aircraft inventory
pre_proc_fleet_by_type <- Aircraft_inventory  %>%
  filter(Status == "Active" )%>%
  group_by(`Operator Region`,`Aircraft Type`)  %>%
  summarise( "no_obs" = n())  %>%
  arrange(desc(no_obs))  %>%
  adorn_totals(where = "row")

#2.2.3 QA223 - Global Fleet with aircraft type and basic age analysis from harmonised IBA Aircraft inventory
pre_proc_Global_fleet_by_oper_type <- Aircraft_inventory  %>%
  filter(Status == "Active" ) %>%
  group_by(`Aircraft Type`) %>%
  summarise("mean age (mid year)" = round(mean(Age_midYear),0),
            "median age (mid year)" = round(median(Age_midYear),0),
            "90th percentile age (mid year)" = round(quantile(Age_midYear, 0.90),0),
            "95th percentile age (mid year)" = round(quantile(Age_midYear, 0.95),0),
            "no_obs" = n() ) %>%
  arrange(desc(no_obs))  %>%
  adorn_totals(where = "row")

#2.2.4 QA224 - UK Fleet with aircraft type and basic age analysis from harmonised IBA Aircraft inventory
pre_proc_UK_fleet_by_oper_type <- Aircraft_inventory  %>%
  filter(Status == "Active" & `Operator Country` == "United Kingdom") %>%
  group_by(`Aircraft Type`) %>%
  summarise("mean age (mid year)" = round(mean(Age_midYear),0),
            "median age (mid year)" = round(median(Age_midYear),0),
            "90th percentile age (mid year)" = round(quantile(Age_midYear, 0.90),0),
            "95th percentile age (mid year)" = round(quantile(Age_midYear, 0.95),0),
            "no_obs" = n() ) %>%
  arrange(desc(no_obs))  %>%
  adorn_totals(where = "row")

#2.2.5 QA225 - WE Fleet with aircraft type and basic age analysis from harmonised IBA Aircraft inventory
pre_proc_WE_fleet_by_oper_type <- Aircraft_inventory  %>%
  filter(Status == "Active" & `Operator Subregion` == "Western Europe") %>%
  group_by(`Aircraft Type`) %>%
  summarise("mean age (mid year)" = round(mean(Age_midYear),0),
            "median age (mid year)" = round(median(Age_midYear),0),
            "90th percentile age (mid year)" = round(quantile(Age_midYear, 0.90),0),
            "95th percentile age (mid year)" = round(quantile(Age_midYear, 0.95),0),
            "no_obs" = n() )  %>%
  arrange(desc(no_obs))  %>%
  adorn_totals(where = "row")

#2.2.6 QA226 - OECD Fleet with aircraft type and basic age analysis from harmonised IBA Aircraft inventory
pre_proc_OECD_fleet_by_oper_type <- Aircraft_inventory  %>%
  filter(Status == "Active" & `Operator Country` %in% OECD_countries) %>%
  group_by(`Aircraft Type`) %>%
  summarise("mean age (mid year)" = round(mean(Age_midYear),0),
            "median age (mid year)" = round(median(Age_midYear),0),
            "90th percentile age (mid year)" = round(quantile(Age_midYear, 0.90),0),
            "95th percentile age (mid year)" = round(quantile(Age_midYear, 0.95),0),
            "no_obs" = n() ) %>%
  arrange(desc(no_obs))  %>%
  adorn_totals(where = "row")

#2.2.7 QA227 - Basic in-year Global retirement analysis from harmonised IBA Aircraft inventory
pre_proc_age_at_Global_retirement <- Aircraft_inventory  %>%
  filter(Status == "Retired") %>%
  group_by(`Aircraft Family`) %>%
  summarise  ("median age" = round(median(Age_midYear),0)
             ,"min age" = round(min(Age_midYear),0),
             "max age" = round(max(Age_midYear),0),
             "no_obs" = n() ) %>%
  arrange(`Aircraft Family`)

#2.2.8 QA228- Basic in-year UK retirement analysis from harmonised IBA Aircraft inventory
pre_proc_age_at_UK_retirement <- Aircraft_inventory  %>%
  filter(Status == "Retired" & `Operator Country` == "United Kingdom") %>%
  group_by(`Aircraft Family`) %>%
  summarise  ("median age" = round(median(Age_midYear),0)    ,
             "min age" = round(min(Age_midYear),0),
             "max age" = round(max(Age_midYear),0),
             "no_obs" = n() ) %>%
  arrange(`Aircraft Family`)

```

```r
#2.2.9 QA229- Basic in-year WE retirement analysis from harmonised IBA Aircraft inventory
pre_proc_age_at_WE_retirement <- Aircraft_inventory  %>%
  filter(Status == "Retired" & `Operator Subregion` == "Western Europe") %>%
  group_by(`Aircraft Family`) %>%
  summarise ("median age" = round(median(Age_midYear),0)    ,
             "min age" = round(min(Age_midYear),0),
             "max age" = round(max(Age_midYear),0),
             "no_obs" = n() ) %>%
  arrange(`Aircraft Family`)

#2.2.10 QA2210- Basic in-year OECD retirement analysis from harmonised IBA Aircraft inventory
pre_proc_age_at_OECD_retirement <- Aircraft_inventory  %>%
  filter(Status == "Retired" & `Operator Country` %in% OECD_countries) %>%
  group_by(`Aircraft Family`) %>%
  summarise ("median age" = round(median(Age_midYear),0)    ,
             "min age" = round(min(Age_midYear),0),
             "max age" = round(max(Age_midYear),0),
             "no_obs" = n() ) %>%
  arrange(`Aircraft Family`)




# 2.3.1 QA231- Global aircraft orders absolute values by years using IBA Aircraft order
pre_proc_Global_order_count <- Aircraft_order %>%
  filter(`Build Year`>2016) %>%
  group_by(`Aircraft Family`,`Aircraft Model`,`Build Year`) %>%
  summarise(no_obs = n()) %>%
  spread(`Build Year`,no_obs,fill=0) %>%
  adorn_totals(where =  c("row", "col"))

# 2.3.2 QA232 - Global aircraft orders percentages by years using IBA Aircraft order
pre_proc_Global_order_percentage <- Aircraft_order %>%
  filter(`Build Year`>2016) %>%
  group_by(`Aircraft Family`,`Aircraft Model`,`Build Year`) %>%
  summarise(no_obs = n()) %>%
  spread(`Build Year`,no_obs,fill=0)  %>%
  adorn_percentages(denominator = "col") %>%
  mutate_if(is.numeric, round, 2)

# 2.3.3 QA233- UK aircraft orders absolute values by years using IBA Aircraft order
pre_proc_UK_order_count <- Aircraft_order %>%
  filter(`Build Year`>2016 & `Operator Country` == "United Kingdom") %>%
  group_by(`Aircraft Family`,`Aircraft Model`,`Build Year`) %>%
  summarise(no_obs = n()) %>%
  spread(`Build Year`,no_obs,fill=0)  %>%
  adorn_totals(where =  c("row", "col"))

# 2.3.4 QA234 - UK aircraft orders percentages by years using IBA Aircraft order
pre_proc_UK_order_percentage <- Aircraft_order %>%
  filter(`Build Year`>2016 & `Operator Country` == "United Kingdom") %>%
  group_by(`Aircraft Family`,`Aircraft Model`,`Build Year`) %>%
  summarise(no_obs = n()) %>%
  spread(`Build Year`,no_obs,fill=0)  %>%
  adorn_percentages(denominator = "col") %>%
  mutate_if(is.numeric, round, 2)

# 2.3.5 QA235- WE aircraft orders absolute values by years using IBA Aircraft order
pre_proc_WE_order_count <- Aircraft_order %>%
  filter(`Build Year`>2016 & `Operator Subregion` == "Western Europe") %>%
  group_by(`Aircraft Family`,`Aircraft Model`,`Build Year`) %>%
  summarise(no_obs = n()) %>%
  spread(`Build Year`,no_obs,fill=0)  %>%
  adorn_totals(where =  c("row", "col"))

# 2.3.6 QA236 - WE aircraft orders percentages by years using IBA Aircraft order
pre_proc_WE_order_percentage <- Aircraft_order %>%
  filter(`Build Year`>2016 & `Operator Subregion` == "Western Europe") %>%
  group_by(`Aircraft Family`,`Aircraft Model`,`Build Year`) %>%
  summarise(no_obs = n()) %>%
  spread(`Build Year`,no_obs,fill=0)  %>%
  adorn_percentages(denominator = "col") %>%
  mutate_if(is.numeric, round, 2)

# 2.3.7 QA237- OECD aircraft orders absolute values by years using IBA Aircraft order
pre_proc_OECD_order_count <- Aircraft_order %>%
  filter(`Build Year`>2016 & `Operator Country` %in% OECD_countries) %>%
  group_by(`Aircraft Family`,`Aircraft Model`,`Build Year`) %>%
  summarise(no_obs = n()) %>%
  spread(`Build Year`,no_obs,fill=0)  %>%
  adorn_totals(where =  c("row", "col"))

# 2.3.8 QA238 - OECD aircraft orders percentages by years using IBA Aircraft order
pre_proc_OECD_order_percentage <- Aircraft_order %>%
  filter(`Build Year`>2016 & `Operator Country` %in% OECD_countries) %>%
  group_by(`Aircraft Family`,`Aircraft Model`,`Build Year`) %>%
  summarise(no_obs = n()) %>%
  spread(`Build Year`,no_obs,fill=0)  %>%
  adorn_percentages(denominator = "col") %>%
  mutate_if(is.numeric, round, 2)





# 2.2 Write PRE PROCESSING Review excel file with dataframes above

# declare a Review workbook object
wb_qa <- createWorkbook(creator = "dtathgur", title = paste0("Review Pre Processing ",year))

# Add worksheets for each dataframe including three coversheets

addWorksheet(wb_qa, paste0("CAA_ATMS>>>"),
             tabColour = "#00bfff" )       #210
addWorksheet(wb_qa, paste0("Topsheet_",year),
             tabColour = "#00bfff" )       #211
addWorksheet(wb_qa, paste0("Serv_oper_",year),
             tabColour = "#00bfff" )      #212
addWorksheet(wb_qa, paste0("Ac_name_",year),
             tabColour = "#00bfff" )        #213
addWorksheet(wb_qa, paste0("Yr_mon_",year) ,
             tabColour = "#00bfff")         #214
addWorksheet(wb_qa, paste0("Rep_arpt_dire_",year),
             tabColour = "#00bfff" )  #215
addWorksheet(wb_qa, paste0("Reg_name_seat_",year),
             tabColour = "#00bfff" )      #216
addWorksheet(wb_qa, paste0("Seat_analysis_",year),
             tabColour = "#00bfff" )  #217
addWorksheet(wb_qa, paste0("Seat_quar_analysis_",year),
             tabColour = "#00bfff" )  #218


addWorksheet(wb_qa, paste0("IBA_FLEETS>>>"),
             tabColour = "#ff0040")       #220
addWorksheet(wb_qa, paste0("Fleet_by_operator_",year), tabColour = "#ff0040")     #221
addWorksheet(wb_qa, paste0("Fleet_by_type_",year), tabColour = "#ff0040")         #222
addWorksheet(wb_qa, paste0("Global_Fleet_",year), tabColour = "#ff0040")         #223
addWorksheet(wb_qa, paste0("UK_Fleet_",year), tabColour = "#ff0040")             #224
addWorksheet(wb_qa, paste0("WE_Fleet_",year), tabColour = "#ff0040")             #225
addWorksheet(wb_qa, paste0("OECD_Fleet_",year), tabColour = "#ff0040")           #226
addWorksheet(wb_qa, paste0("Global_retirements_",year), tabColour = "#ff0040")   #227
addWorksheet(wb_qa, paste0("UK_retirements_",year), tabColour = "#ff0040")       #228
addWorksheet(wb_qa, paste0("WE_retirements_",year), tabColour = "#ff0040")       #229
addWorksheet(wb_qa, paste0("OECD_retirements_",year), tabColour = "#ff0040")     #2210


addWorksheet(wb_qa, paste0("IBA_ORDERS>>>"),
             tabColour = "#90ee90")       #230
addWorksheet(wb_qa, paste0("Global_order_count_",year), tabColour = "#90ee90")      # QA231
addWorksheet(wb_qa, paste0("Global_order_percentage_",year), tabColour = "#90ee90") # QA232
addWorksheet(wb_qa, paste0("UK_order_count_",year), tabColour = "#90ee90")          # QA233
addWorksheet(wb_qa, paste0("UK_order_percentage_",year), tabColour = "#90ee90")     # QA234
addWorksheet(wb_qa, paste0("WE_order_count_",year), tabColour = "#90ee90")         # QA235
addWorksheet(wb_qa, paste0("WE_order_percentage_",year), tabColour = "#90ee90")    # QA236
addWorksheet(wb_qa, paste0("OECD_order_count_",year), tabColour = "#90ee90")       # QA237
addWorksheet(wb_qa, paste0("OECD_order_percentage_",year), tabColour = "#90ee90")  # QA238





# Add datatable to each worksheet




writeData(wb_qa, paste0("Topsheet_",year),c(paste("CAA ATM Data "," - ",date())," No. of records and variables"), #211
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
          )

writeData(wb_qa, paste0("Serv_oper_",year) , c(paste("CAA ATM Data ","- ",date())," service_type and operation_type"), #212
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
            )

writeData(wb_qa, paste0("Ac_name_",year) , c(paste("CAA ATM Data "," - ",date())," aircraft_name, distance(s) flown and haul
calc (i.e. if max distance flown <3750 then short otherwise both) "),#213
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
             )

writeData(wb_qa,  paste0("Yr_mon_",year), c(paste("CAA ATM Data "," - ",date()),"year, month and month_no"), #214
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
               )

writeData(wb_qa,  paste0("Rep_arpt_dire_",year) ,  c(paste("CAA ATM Data "," - ",date())," reporting_airport_code, name and
direction_type"), #215
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
            )

writeData(wb_qa, paste0("Reg_name_seat_",year),  c(paste("CAA ATM Data "," - ",date())," ALL aircraft_reg and _name
combinations (inc. UNKNOWN and zeroes)
                                                   with available seats"), #216
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
              )

writeData(wb_qa, paste0("Seat_analysis_",year) , c(paste("CAA ATM Data "," - ",date())," Basic seat analysis on aircraft_reg
and _name (exc. UNKNOWN and zeroes)
                                                           combinations "), #217
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
               )
```

```r

writeData(wb_qa, paste0("Seat_quar_analysis_",year) , c(paste("CAA ATM Data "," - ",date())," Quartile based seat analysis on unique aircraft_reg (exc. UNKNOWN and zeroes)
                                                                          and recorded _names  "), #218
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
)









writeData(wb_qa, paste0("Fleet_by_operator_",year) ,  c(paste("IBA Fleets "," - ",date())," Active global fleet by operator"), #221
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
         )

writeData(wb_qa, paste0("Fleet_by_type_",year) ,  c(paste("IBA Fleets "," - ",date())," Active global fleet by operator region and aircraft type"), #222
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
               )

writeData(wb_qa, paste0("Global_Fleet_",year) ,  c(paste("IBA Fleets "," - ",date())," Active global operators fleet by age summaries as of mid-year"), #223
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
         )

writeData(wb_qa, paste0("UK_Fleet_",year) ,  c(paste("IBA Fleets "," - ",date())," Active UK operators fleet by age summaries as of mid-year"), #224
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
             )

writeData(wb_qa, paste0("WE_Fleet_",year) ,  c(paste("IBA Fleets "," - ",date())," Active WE operators fleet by age summaries as of mid-year"), #225
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
    )

writeData(wb_qa, paste0("OECD_Fleet_",year) ,  c(paste("IBA Fleets "," - ",date())," Active OECD operators fleet by age summaries as of mid-year"), #226
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
             )

writeData(wb_qa, paste0("Global_retirements_",year) ,  c(paste("IBA Fleets "," - ",date()),"Basic global retirement analysis"), #227
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
               )

writeData(wb_qa, paste0("UK_retirements_",year) ,  c(paste("IBA Fleets "," - ",date()),"Basic UK retirement analysis"), #228
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
            )

writeData(wb_qa, paste0("WE_retirements_",year) , c(paste("IBA Fleets "," - ",date())," Basic WE retirement analysis"), #229
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
               )

writeData(wb_qa, paste0("OECD_retirements_",year) , c(paste("IBA Fleets "," - ",date())," Basic OECD retirement analysis"), #2210
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
            )

writeData(wb_qa, paste0("Global_order_count_",year) ,  c(paste("IBA Orders "," - ",date())," Global aircraft orders (no.) by aircraft family and model"), #231
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
           )

writeData(wb_qa, paste0("Global_order_percentage_",year) ,  c(paste("IBA Orders "," - ",date())," Global aircraft orders (pc)
by aircraft family and model"), #232
           startCol = 1, startRow = 1, xy = NULL,
           borders = "rows",
           borderColour = getOption("openxlsx.borderColour", "black"),
           borderStyle = getOption("openxlsx.borderStyle", "thick")
        )

writeData(wb_qa, paste0("UK_order_count_",year) ,  c(paste("IBA Orders "," - ",date())," UK operators aircraft orders (no.)
by aircraft family and model"),
           startCol = 1, startRow = 1, xy = NULL,
           borders = "rows",
           borderColour = getOption("openxlsx.borderColour", "black"),
           borderStyle = getOption("openxlsx.borderStyle", "thick")
               )

writeData(wb_qa, paste0("UK_order_percentage_",year) ,  c(paste("IBA Orders "," - ",date())," UK operators aircraft orders
(pc) by aircraft family and model"),
           startCol = 1, startRow = 1, xy = NULL,
           borders = "rows",
           borderColour = getOption("openxlsx.borderColour", "black"),
           borderStyle = getOption("openxlsx.borderStyle", "thick")
          )

writeData(wb_qa, paste0("WE_order_count_",year) ,  c(paste("IBA Orders "," - ",date())," WE operators aircraft orders (no.)
by aircraft family and model"),
           startCol = 1, startRow = 1, xy = NULL,
           borders = "rows",
           borderColour = getOption("openxlsx.borderColour", "black"),
           borderStyle = getOption("openxlsx.borderStyle", "thick")
          )

writeData(wb_qa, paste0("WE_order_percentage_",year) ,  c(paste("IBA Orders "," - ",date())," WE operators aircraft orders
(pc) by aircraft family and model"),
           startCol = 1, startRow = 1, xy = NULL,
           borders = "rows",
           borderColour = getOption("openxlsx.borderColour", "black"),
           borderStyle = getOption("openxlsx.borderStyle", "thick")
             )

writeData(wb_qa, paste0("OECD_order_count_",year) , c(paste("IBA Orders "," - ",date())," OECD operators aircraft orders
(no.) by aircraft family and model"),
           startCol = 1, startRow = 1, xy = NULL,
           borders = "rows",
           borderColour = getOption("openxlsx.borderColour", "black"),
           borderStyle = getOption("openxlsx.borderStyle", "thick")
         )

writeData(wb_qa, paste0("OECD_order_percentage_",year), c(paste("IBA Orders "," - ",date())," OECD operators aircraft orders
(pc) by aircraft family and model"),
           startCol = 1, startRow = 1, xy = NULL,
           borders = "rows",
           borderColour = getOption("openxlsx.borderColour", "black"),
           borderStyle = getOption("openxlsx.borderStyle", "thick")
               )

# Add datatable to each worksheet

writeDataTable(wb_qa, paste0("Topsheet_",year),data.frame(pre_proc_row_col_total),
               startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium3",
               tableName = NULL, headerStyle = NULL, withFilter = TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa, paste0("Serv_oper_",year) , pre_proc_QA_serv_oper,
               startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium2",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa, paste0("Ac_name_",year) , pre_proc_QA_aircraft_name, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium2",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa,  paste0("Yr_mon_",year), pre_proc_QA_year_month, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium2",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa,  paste0("Rep_arpt_dire_",year) , pre_proc_QA_reporting_airport_direction, startCol = 1, startRow = 3,
xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium2",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa, paste0("Reg_name_seat_",year), pre_proc_QA_reg_name_seat, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium2",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa, paste0("Seat_analysis_",year) , pre_proc_QA_reg_name_seat_analysis , startCol = 1, startRow = 3, xy =
NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium2",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)


writeDataTable(wb_qa, paste0("Seat_quar_analysis_",year) , pre_proc_QA_unique_reg_name_seat_quartile_analysis , startCol = 1,
startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium2",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)



writeDataTable(wb_qa, paste0("Fleet_by_operator_",year) , pre_proc_fleet_by_oper, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium3",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa, paste0("Fleet_by_type_",year) , pre_proc_fleet_by_type, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa, paste0("Global_Fleet_",year) , pre_proc_Global_fleet_by_oper_type, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa, paste0("UK_Fleet_",year) , pre_proc_UK_fleet_by_oper_type, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa, paste0("WE_Fleet_",year) , pre_proc_WE_fleet_by_oper_type, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa, paste0("OECD_Fleet_",year) , pre_proc_OECD_fleet_by_oper_type, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa, paste0("Global_retirements_",year) , pre_proc_age_at_Global_retirement, startCol = 1, startRow = 3, xy
= NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa, paste0("UK_retirements_",year) ,pre_proc_age_at_UK_retirement, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa, paste0("WE_retirements_",year) ,pre_proc_age_at_WE_retirement, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa, paste0("OECD_retirements_",year) ,pre_proc_age_at_OECD_retirement, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa, paste0("Global_order_count_",year) , pre_proc_Global_order_count , startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa, paste0("Global_order_percentage_",year) , pre_proc_Global_order_percentage , startCol = 1, startRow =
3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium4",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa, paste0("UK_order_count_",year) , pre_proc_UK_order_count, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa, paste0("UK_order_percentage_",year) ,pre_proc_UK_order_percentage, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa, paste0("WE_order_count_",year) , pre_proc_WE_order_count, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa, paste0("WE_order_percentage_",year) ,pre_proc_WE_order_percentage, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
```

```r

writeDataTable(wb_qa, paste0("OECD_order_count_",year) , pre_proc_OECD_order_count, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa, paste0("OECD_order_percentage_",year) ,pre_proc_OECD_order_percentage, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)


# Save into a workbook and increase size of columns for easy read

saveWorkbook(wb_qa, file = paste0("Pre_processing_review_",year,".xlsx") , overwrite = TRUE)
for(sheetindex in c(1:length(getSheetNames(paste0("Pre_processing_review_",year,".xlsx"))))  )
{setColWidths(wb_qa, sheet = sheetindex, cols = 1:7, widths = 20)
}
saveWorkbook(wb_qa, file = paste0("Pre_processing_review_",year,".xlsx") , overwrite = TRUE)

# Remove pre_proc dataframes as no longer needed
rm(wb_qa, sheetindex, no_of_records, no_of_cols, pre_proc_row_col_total, pre_proc_QA_serv_oper, pre_proc_QA_aircraft_name,
   pre_proc_QA_year_month, pre_proc_QA_reporting_airport_direction, pre_proc_QA_reg_name_seat, pre_proc_QA_reg_name_seat_analysis, pre_proc_QA_unique_reg_name_seat_quartile_analysis,
   pre_proc_fleet_by_oper,pre_proc_fleet_by_type, pre_proc_Global_fleet_by_oper_type, pre_proc_UK_fleet_by_oper_type, pre_proc_WE_fleet_by_oper_type, pre_proc_OECD_fleet_by_oper_type,
   pre_proc_age_at_Global_retirement, pre_proc_age_at_UK_retirement, pre_proc_age_at_WE_retirement, pre_proc_age_at_OECD_retirement, pre_proc_Global_order_count ,
   pre_proc_Global_order_percentage, pre_proc_UK_order_count, pre_proc_UK_order_percentage, pre_proc_WE_order_count, pre_proc_WE_order_percentage,
   pre_proc_OECD_order_count, pre_proc_OECD_order_percentage)




# Step 3 - Process data by merging, cutting and applying heuristics -----

# 3.1 For all ATMs - merging reporting and lastnext airport names for SPASM zones
ATM_temp <- merge(ATM_temp,SPASMZone , by.x="reporting_airport", by.y ="AP_name", all.x=TRUE )
names(ATM_temp)[names(ATM_temp) == "SPASM_zone"] <- "reporting_spasm_zone"
ATM_temp <- merge(ATM_temp,SPASMZone , by.x="lastnext_airport", by.y ="AP_name", all.x=TRUE )
names(ATM_temp)[names(ATM_temp) == "SPASM_zone"] <- "lastnext_spasm_zone"

# 3.2 Merging further FMM aircraft info from CAAICAOIATA (lookup file for aircraft)
ATM_temp <- merge(ATM_temp, CAAICAOIATA, by.x ="aircraft_name" , by.y = "CAA_Acf_Name" , all.x = TRUE)


# 3.3 Merging CAA ATM movements data Registration with harmonised IBA aircraft inventory
# The previous version had an unused bit of code which has been cleaned out after QA by MH - Apr 2019.
# Upon discussion DT decided this unused bit of code wasnt necessary and better functionality was provided by a code in Section 1.7

ATM_temp <- merge(ATM_temp, Aircraft_inventory, by.x = "aircraft_reg", by.y = "Registration", all.x = TRUE)

# 3.3.1 Creating a merge report for to see successful and unsuccessful matches -
# unsuccessful matches will we appropriately handled i.e. removed for fmm purposes

# Match success reports by registration
IBA_match_status <- ATM_temp %>%
  group_by(aircraft_reg) %>%
  summarise(no_obs = n()) %>%
  mutate(harmonised_match_success = aircraft_reg %in% Aircraft_inventory$Registration,
         jan_match_success = aircraft_reg %in% Aircraft_inventory_Jan$Registration,
         dec_match_success = aircraft_reg %in% Aircraft_inventory_Dec$Registration,
         missing_match_success = aircraft_reg %in% Aircraft_inventory_Missing$Registration ) %>%
  arrange(desc(no_obs))

  # Match success reports by TRUE and FALSE
IBA_match_success <- IBA_match_status %>%
  gather(key, value, harmonised_match_success:missing_match_success ) %>%
  group_by(value, key) %>%
  summarise(observations = sum(no_obs))  %>%
  spread(key, observations)

IBA_match_status <- IBA_match_status %>%
  adorn_totals(where = "row")

    # Aircraft reg not matched
IBA_negative_match <- ATM_temp %>%
  filter(ATM_temp$aircraft_reg %in% IBA_match_status$aircraft_reg[IBA_match_status$harmonised_match_success == FALSE]) %>%
  group_by(aircraft_reg,aoc_holder_name, aircraft_name) %>%
  summarise(no_obs = n(), terminal_pax = sum(pax_terminal_total), cargo_weight_kg = sum(cargo_weight_kg)) %>%
  arrange(desc(no_obs))  %>%
  adorn_totals(where = "row")



# 3.3.2 Records where merge has been unsuccesful, change N/A in Seating Capacity to -1 for cutting into seat class zero
# These records will use CAA ATM movement's data available seats for FMM seat classing
# Records where both the above approaches cannot be performed are removed

ATM_temp$`Seating Capacity`[is.na(ATM_temp$`Seating Capacity`)] <- -1

#3.4 Allocating flight movements to DFT carrier type as Defined by the Department Aviation Model
# Schedule(Sch), Charter(Ch), Low Cost Carrier(LCC), Freight(F) and Other(O) - Simplified as S,C,L,F,O for notation
# This step applies SCLFO classification based on the following rules to the relevant ATM file
# Note - (CAA) service_type has three possible values of "Cargo Only", "Passenger" & "Transit Cargo" - CAA terminology NOT DfT Aviation Model
# Note - (CAA) operation_type has three possible values of "Charter", "Government Charter" & "Scheduled" - CAA terminology NOT DfT Aviation Model
# Classification begings by creation of a new columnn called SCLFO

```

```r
ATM_temp$SCLFO = as.character("",nrow(ATM_temp))

# Rules commence as below and subsequent rules override the previous one - where applicable

# Rule 1 - If service_type is "Passenger", the movement is classed as S (Schedule)

ATM_temp$SCLFO[ATM_temp$service_type %in% c("Passenger")] = "S"

# Rule 2 - If the aoc_holder_name is one of - "RYANAIR", "EASYJET AIRLINE COMPANY LTD", "EASYJET SWITZERLAND", "JET2.COM LTD" or "TUI AIRWAYS LTD" - Then movement is reclassed as L (Low Cost)

ATM_temp$SCLFO[ATM_temp$aoc_holder_name %in% c("RYANAIR", "EASYJET AIRLINE COMPANY LTD", "EASYJET SWITZERLAND", "JET2.COM LTD")] = "L"

# Rule 3 - if operation_type is "Charter" , then movement is reclassed as C (Charter)

ATM_temp$SCLFO[ATM_temp$operation_type %in% c("Charter")] = "C"

# Rule 4 - if service_type is either "Cargo Only" or "Transit Cargo" , then movement is reclassed as F (Freighter)

ATM_temp$SCLFO[ATM_temp$service_type %in% c("Cargo Only","Transit Cargo")] = "F"

# Rule 5 - if operation_type is "Government Charter" , then movement is reclassed as O (Other)

ATM_temp$SCLFO[ATM_temp$operation_type %in% c("Government Charter")] = "O"

# Rule 6 - if pax_terminal_total & pax_transit_total & cargo_weight_kg are all zero, then movement is reclassed as O (Other)

ATM_temp$SCLFO[ATM_temp$pax_terminal_total==0 & ATM_temp$pax_transit_total==0 & ATM_temp$cargo_weight_kg==0 ] = "O"

# Rule 7- if the lastnext airport is in the UK and Chartered, it is classed as O

ATM_temp$SCLFO[ATM_temp$lastnext_spasm_zone %in% UK_modelled_airports & ATM_temp$SCLFO %in% c("C")] = "O"
ATM_temp$SCLFO[ATM_temp$lastnext_spasm_zone %in% c(599) & ATM_temp$SCLFO %in% c("C")] = "O"

# Rule 8 - if the flight movement is S or L and has either reporting or last nextairport SPASM zone as 599, it is classed as O

ATM_temp$SCLFO[ATM_temp$SCLFO %in% c("S","L") & ATM_temp$reporting_spasm_zone == 599] = "O"
ATM_temp$SCLFO[ATM_temp$SCLFO %in% c("S","L") & ATM_temp$lastnext_spasm_zone == 599] = "O"

# Now convert SCLFO values to factors to speed up processing

ATM_temp$SCLFO <- as.factor(ATM_temp$SCLFO)

# Step 3.4.1 SPASM Match for reference

SPASM_Match_Airport_SCLFO  <- ATM_temp %>%
  filter(reporting_spasm_zone != 599) %>%
  group_by(reporting_spasm_zone,reporting_airport,SCLFO) %>%
  summarise(no_obs = n()) %>%
  spread(SCLFO, no_obs, fill=0) %>%
  mutate(SCL=S+C+L,
         SCLFO=SCL+F+O,
         ` ` ='') %>%
  select(reporting_spasm_zone, reporting_airport, S, L, C , ` `, SCL, F, O, SCLFO) %>%
  arrange()  %>%
  adorn_totals(where = "row")


#3.5 Assigning seat classes to aircraft movement using Seating capacity(IBA) first and then available_seats(CAA) for the missing ones

ATM_temp$FMM_seatclass <- cut(ATM_temp$`Seating Capacity`, c(-Inf,1,70,150,250,350,500,Inf),labels=c("0","1","2","3","4","5","6"))

ATM_temp$FMM_seatclass[ATM_temp$FMM_seatclass==0] <- cut(ATM_temp$available_seats[ATM_temp$FMM_seatclass==0], c(-Inf,1,70,150,250,350,500,Inf),
                                                         labels=c("0","1","2","3","4","5","6"))


# 3.6 Cleaning out non-relevant information for FMM purposes



# TRUNCATE ATM records only for modelled airports and SCL
ATM_temp <- ATM_temp[ATM_temp$reporting_spasm_zone %in% UK_modelled_airports , ]
ATM_temp <- ATM_temp[ATM_temp$SCLFO %in% c("S","C","L"),]

# Clean out "Unknown" aircraft_reg , "Unspecified" aircraft_name, "Unknown" lastnext_airport and NA Aircraft ICAO Code
ATM_temp <- ATM_temp[ATM_temp$aircraft_reg!= "UNKNOWN" ,]
ATM_temp <- ATM_temp[ATM_temp$aircraft_name!= "UNSPECIFIED",]
ATM_temp <- ATM_temp[ATM_temp$lastnext_airport!= "UNKNOWN",]
ATM_temp <- ATM_temp[ATM_temp$`Aircraft ICAO Code` != "Unknown",]

ATM_temp <- ATM_temp %>% drop_na(c(aircraft_reg, aircraft_name, lastnext_airport, reporting_airport, `Aircraft ICAO Code`))


# 3.7 Cleaning based on items merged from Aircraft Inventory
# cleaning out items that have zero seat class
ATM_temp <- ATM_temp[ATM_temp$FMM_seatclass!= 0,]

# cleaning out items that have NA Age_midYear field
ATM_temp <- ATM_temp %>% drop_na(Age_midYear)

# For proportionality any IATA code that occurs less than 100 times in the year is removed

ATM_temp <- ATM_temp[ATM_temp$IATA %in% names(which(table(ATM_temp$IATA) > 100)),]

# 3.8 Create a new column representing FMM service type - Sch, Ch and NFC (FMM terminology for low cost)
ATM_temp$fmm_service_type = as.character("",nrow(ATM_temp))

ATM_temp$fmm_service_type[ATM_temp$SCLFO %in% c("S")] = "Sch"
ATM_temp$fmm_service_type[ATM_temp$SCLFO %in% c("C")] = "Ch"
ATM_temp$fmm_service_type[ATM_temp$SCLFO %in% c("L")] = "NFC"

```

```r
# Create a new column representing FMM segment i.e. model worksheets in the fmm


ATM_temp$fmm_segment  <- paste0("c",ATM_temp$FMM_seatclass,ATM_temp$fmm_service_type)



# Threshold to at least 200 atms per year for that IATA code within the segment to remove noisy data
ATM_temp$combinedrule <- with(ATM_temp, interaction(IATA,fmm_segment))
ATM_temp <- ATM_temp[!table(ATM_temp$combinedrule)[ATM_temp$combinedrule] < 200,]

#cleaning levels that have no observations
ATM_temp <- droplevels(ATM_temp)

# Removing intermediate versions of flight inventories

rm(Aircraft_inventory_Dec,Aircraft_inventory_Jan,Aircraft_inventory_Missing)


# Step 4 - Post processing Quality Assurance prior to feeding into FMM ----
## Relevant data in previous step are consolidated into dataframes and findings are saved in single .xlsx file for review

#4.1.1 QA411 - High level integrity check  - Get number of records for resultant ATMs file

post_proc_ATM_temp_no_of_records <- nrow(ATM_temp)
post_proc_ATM_temp_no_of_cols <- ncol(ATM_temp)
post_proc_row_col_total <- rbind(post_proc_ATM_temp_no_of_cols,post_proc_ATM_temp_no_of_records)
colnames(post_proc_row_col_total) <- "no_obs"

#4.1.2 QA412 - Dataframe with disaggregation of resultant ATMs by SCLFO
post_proc_QA_SCLFO <- ATM_temp %>%
  group_by(SCLFO,direction_type)%>%
  summarise(no_obs = n()) %>%
 arrange(desc(no_obs))  %>%
  adorn_totals(where = "row")

#4.1.3 QA413 - Dataframe with aircraft registrations comparing IBA's seating capacity and CAA's available seats arranged in
descending order

ATM_temp$IBA_seatclass <- cut(ATM_temp$`Seating Capacity`,
c(-Inf,1,70,150,250,350,500,Inf),labels=c("0","1","2","3","4","5","6"))
ATM_temp$CAA_seatclass <- cut(ATM_temp$available_seats, c(-Inf,1,70,150,250,350,500,Inf),labels=c("0","1","2","3","4","5","6"))

post_proc_QA_AC_IATA_seats <- ATM_temp %>%
  group_by(aircraft_reg,IATA,IBA_seatclass,CAA_seatclass, FMM_seatclass) %>%
  summarise(no_obs = n())

post_proc_QA_AC_IATA_seats$difference <- as.numeric(post_proc_QA_AC_IATA_seats$IBA_seatclass) -
as.numeric(post_proc_QA_AC_IATA_seats$CAA_seatclass)
post_proc_QA_AC_IATA_seats$difference <- abs(post_proc_QA_AC_IATA_seats$difference)
post_proc_QA_AC_IATA_seats$missing_IBA <- ifelse(post_proc_QA_AC_IATA_seats$IBA_seatclass==0, "Missing IBA", "IBA present")
post_proc_QA_AC_IATA_seats$missing_CAA <- ifelse(post_proc_QA_AC_IATA_seats$CAA_seatclass==0, "Missing CAA", "CAA present")

ATM_temp$IBA_seatclass <- NULL
ATM_temp$CAA_seatclass <- NULL

#4.1.4 QA414 - Dataframe with disaggregation of resultant ATMs by FMM seat class
post_proc_QA_SeatClass <- ATM_temp %>%
  group_by(FMM_seatclass, SCLFO) %>%
  summarise(no_obs = n()) %>%
  spread(SCLFO,no_obs, fill = 0) %>%
  select(FMM_seatclass,S, L, C) %>%
  adorn_totals(where = "row")

#4.1.5 QA415 - Dataframe with contingency table for airports vs SCLFO disaggregation
post_proc_QA_Airport_SCLFO <- ATM_temp %>%
  group_by(reporting_spasm_zone,reporting_airport,SCLFO) %>%
  summarise(no_obs = n()) %>%
  spread(SCLFO,no_obs, fill = 0) %>%
  select(reporting_spasm_zone,reporting_airport, S, L, C) %>%
  adorn_totals(where = c("col","row"))

#4.1.6 QA416 - Dataframe with contingency table by aircraft code, type and total
post_proc_QA_AC_IATA_SCLFO <- ATM_temp %>%
  group_by(IATA, aircraft_name, `Aircraft ICAO Code`,In_Production,Fmm_Haul, FMM_seatclass,SCLFO) %>%
  summarise(no_obs = n()) %>%
  spread(SCLFO,no_obs, fill = 0) %>%
  select(IATA, aircraft_name, `Aircraft ICAO Code`,In_Production,Fmm_Haul, FMM_seatclass, S, L, C) %>%
  adorn_totals(where = "col")

#4.1.7 QA417 - Dataframe with contingency table by in production IATA code (only) , seat class, totals and percents
post_proc_QA_AC_IATA_SCLFO_InProd <- ATM_temp %>% filter(In_Production == "1") %>%
  group_by(FMM_seatclass, IATA, SCLFO) %>%
  summarise(no_obs = n()) %>%
  spread(SCLFO,no_obs, fill = 0) %>%
  select(IATA,FMM_seatclass, S, L, C) %>%
  arrange(FMM_seatclass) %>%
  ungroup()%>%
  adorn_totals(where = "col") %>%
  group_by(FMM_seatclass) %>%
  mutate("S pc within seatclass" = round(S/sum(S),2),
         "L pc within seatclass" = round(L/sum(L),2) ,
         "C pc within seatclass" = round(C/sum(C),2)
        )

# 4.1.8 QA418 - Global aircraft orders without UK Low Cost Carriers - absolute values by years using IBA Aircraft order
post_proc_Global_family_order_count_excUKLCC <- Aircraft_order %>%
  filter(`Build Year`>2016 & !Operator %in% c("easyJet","Ryanair","Jet2.com")) %>%
  group_by(`Aircraft Family`,`Build Year`) %>%
  summarise(no_obs = n()) %>%
  spread(`Build Year`,no_obs,fill=0) %>%
  adorn_totals(where =  c("row", "col"))

# 4.1.9 QA419 - Global aircraft orders percentages without UK Low Cost Carriers by years using IBA Aircraft order
post_proc_Global_family_order_percentage_excUKLCC <- Aircraft_order %>%
  filter(`Build Year`>2016 & !Operator %in% c("easyJet","Ryanair","Jet2.com") ) %>%
  group_by(`Aircraft Family`,`Build Year`) %>%
  summarise(no_obs = n()) %>%
  spread(`Build Year`,no_obs,fill=0)  %>%
  adorn_percentages(denominator = "col") %>%
  mutate_if(is.numeric, round, 2)


# 4.1.10 QA4110- Aircraft orders for Ryanair, EasyJet and Jet2 - absolute values by years using IBA Aircraft order
post_proc_UK_low_cost_order_count <- Aircraft_order %>%
  filter(`Build Year`>2016 & Operator %in% c("easyJet","Ryanair","Jet2.com")) %>%
  group_by(`Aircraft Family`,`Aircraft Model`,`Build Year`) %>%
  summarise(no_obs = n()) %>%
  spread(`Build Year`,no_obs,fill=0) %>%
  adorn_totals(where =  c("row", "col"))


# 4.1.11 QA4111 - Aircraft orders for Ryanair, EasyJet and Jet2  percentages by years using IBA Aircraft order
post_proc_UK_low_cost_order_percentage <- Aircraft_order %>%
  filter(`Build Year`>2016 & Operator %in% c("easyJet","Ryanair","Jet2.com")) %>%
  group_by(`Aircraft Family`,`Aircraft Model`,`Build Year`) %>%
  summarise(no_obs = n()) %>%
  spread(`Build Year`,no_obs,fill=0)  %>%
  adorn_percentages(denominator = "col") %>%
  mutate_if(is.numeric, round, 2)


# 4.2 Write POST PROCESSING QA excel file with dataframes above

# 4.2.1  declare a QA workbook object
wb_qa_post <- createWorkbook(creator = "dtathgur", title = paste("Quality Assurance Post",year))

# Add worksheets for each dataframe
addWorksheet(wb_qa_post, paste0("PostProcessedData>>>"),
             tabColour = "#00ffbb" )       #410
addWorksheet(wb_qa_post, paste0("Topsheet_",year),
             tabColour = "#00ffbb" )               #411
addWorksheet(wb_qa_post, paste0("SCLFO_",year),
             tabColour = "#00ffbb")                  #412
addWorksheet(wb_qa_post, paste0("SeatClass_Compare_",year) ,
             tabColour = "#00ffbb")    #413
addWorksheet(wb_qa_post, paste0("SCLFO_SeatClass_",year) ,
             tabColour = "#00ffbb")      #414
addWorksheet(wb_qa_post, paste0("SCLFO_Airport_",year),
             tabColour = "#00ffbb" )        #415
addWorksheet(wb_qa_post, paste0("SCLFO_IATA_",year) ,
             tabColour = "#00ffbb")           #416
addWorksheet(wb_qa_post, paste0("SCLFO_IATA_InProd_",year) ,
             tabColour = "#00ffbb")           #417
addWorksheet(wb_qa_post, paste0("Order_count_Exc_UKLCC_",year) ,
             tabColour = "#00ffbb")           #418
addWorksheet(wb_qa_post, paste0("Order_PC_Exc_UKLCC_",year) ,
             tabColour = "#00ffbb")           #419
addWorksheet(wb_qa_post, paste0("UK_LCC_Order_count_",year) ,
             tabColour = "#00ffbb")           #420
addWorksheet(wb_qa_post, paste0("UK_LCC_Order_PC_",year) ,
             tabColour = "#00ffbb")           #421

addWorksheet(wb_qa_post, paste0("DIAGNOSTICS>>>"),
             tabColour = "#2d4423" )       #Match topsheet

addWorksheet(wb_qa_post, paste0("IBA_match_status_",year),
             tabColour = "#2d4423" )      #422 data frame called IBA_match_status
addWorksheet(wb_qa_post, paste0("IBA_negative_match_",year),
             tabColour = "#2d4423"  )   #423 data frame called IBA_negative_match
addWorksheet(wb_qa_post, paste0("SPASM_match_",year) ,
             tabColour = "#2d4423" )          #424 data frame called SPASM Match


# Add description to each worksheet

writeData(wb_qa_post, paste0("Topsheet_",year),c(paste("Post Processed "," - ",date())," No. of records and variables"), #411
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
          )


writeData(wb_qa_post, paste0("SCLFO_",year),c(paste("Post Processed "," - ",date()),"SCLFO and direction_type"), #412
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
          )
writeData(wb_qa_post, paste0("SeatClass_Compare_",year),c(paste("Post Processed "," - ",date())," Aircraft Reg and seat class approaches for all records"), #413
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
)


writeData(wb_qa_post, paste0("SCLFO_SeatClass_",year),c(paste("Post Processed "," - ",date())," FMM seat class by SCLFO"), #414
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
)
writeData(wb_qa_post, paste0("SCLFO_Airport_",year),c(paste("Post Processed "," - ",date())," Reporting Airports by SCLFO"), #415
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
)


writeData(wb_qa_post, paste0("SCLFO_IATA_",year),c(paste("Post Processed "," - ",date())," Aircraft details by IATA and SCLFO"), #416
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
)

writeData(wb_qa_post, paste0("SCLFO_IATA_InProd_",year),c(paste("Post Processed "," - ",date())," In Production Aircraft details by IATA codes and SCLFO"), #417
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
)

writeData(wb_qa_post, paste0("Order_count_Exc_UKLCC_",year),c(paste("Post Processed "," - ",date())," Global Orders (from IBA) values by aircraft family excl UK LCC"), #418
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
)

writeData(wb_qa_post, paste0("Order_PC_Exc_UKLCC_",year),c(paste("Post Processed "," - ",date())," Global Orders (from IBA) percentages by aircraft familyexc UK LCC"), #419
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
)



writeData(wb_qa_post, paste0("UK_LCC_Order_count_",year),c(paste("Post Processed "," - ",date())," UK Low Cost Carriers order (from IBA) values"), #420
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
)

writeData(wb_qa_post, paste0("UK_LCC_Order_PC_",year),c(paste("Post Processed "," - ",date())," UK Low Cost Carriers order (from IBA) percentages "), #421
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
)




writeData(wb_qa_post, paste0("IBA_match_status_",year),c(paste("Post Processed "," - ",date())," Aircraft reg match in inventory for Harmonised, Jan, Dec and Missing versions"), #422
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
)


writeData(wb_qa_post, paste0("IBA_negative_match_",year),c(paste("Post Processed "," - ",date())," Aircraft reg not matched"), #423
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
)
writeData(wb_qa_post, paste0("SPASM_match_",year),c(paste("Post Processed "," - ",date())," SPASM Match file"), #424
          startCol = 1, startRow = 1, xy = NULL,
          borders = "rows",
          borderColour = getOption("openxlsx.borderColour", "black"),
          borderStyle = getOption("openxlsx.borderStyle", "thick")
)



# Add data to each worksheet

writeDataTable(wb_qa_post, paste0("Topsheet_",year), data.frame(post_proc_row_col_total), startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium3",
               tableName = NULL, headerStyle = NULL, withFilter = TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa_post, paste0("SCLFO_",year), post_proc_QA_SCLFO, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium2",
               tableName = NULL, headerStyle = NULL, withFilter = TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa_post, paste0("SeatClass_Compare_",year), post_proc_QA_AC_IATA_seats, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
               tableName = NULL, headerStyle = NULL, withFilter = TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa_post, paste0("SCLFO_SeatClass_",year), post_proc_QA_SeatClass, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium3",
               tableName = NULL, headerStyle = NULL, withFilter = TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa_post, paste0("SCLFO_Airport_",year), post_proc_QA_Airport_SCLFO, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium3",
               tableName = NULL, headerStyle = NULL, withFilter = TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa_post, paste0("SCLFO_IATA_",year), post_proc_QA_AC_IATA_SCLFO, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium4",
               tableName = NULL, headerStyle = NULL, withFilter = TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa_post, paste0("SCLFO_IATA_InProd_",year), post_proc_QA_AC_IATA_SCLFO_InProd, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium4",
               tableName = NULL, headerStyle = NULL, withFilter = TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa_post, paste0("Order_count_Exc_UKLCC_",year), post_proc_Global_family_order_count_excUKLCC , startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium4",
               tableName = NULL, headerStyle = NULL, withFilter = TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa_post, paste0("Order_PC_Exc_UKLCC_",year), post_proc_Global_family_order_percentage_excUKLCC , startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium4",
               tableName = NULL, headerStyle = NULL, withFilter = TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa_post, paste0("UK_LCC_Order_count_",year), post_proc_UK_low_cost_order_count, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium4",
               tableName = NULL, headerStyle = NULL, withFilter = TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa_post, paste0("UK_LCC_Order_PC_",year), post_proc_UK_low_cost_order_percentage, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium4",
               tableName = NULL, headerStyle = NULL, withFilter = TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa_post, paste0("IBA_match_status_",year),IBA_match_status, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium3",
               tableName = NULL, headerStyle = NULL, withFilter = TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa_post, paste0("IBA_match_status_",year),IBA_match_success, startCol = 9, startRow = 5, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium2",
               tableName = NULL, headerStyle = NULL, withFilter = TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa_post, paste0("IBA_negative_match_",year), IBA_negative_match, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium6",
               tableName = NULL, headerStyle = NULL, withFilter = TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

writeDataTable(wb_qa_post, paste0("SPASM_match_",year), SPASM_Match_Airport_SCLFO, startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium4",
               tableName = NULL, headerStyle = NULL, withFilter = TRUE,
               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

# Save into a workbook and increase size of columns for easy read

saveWorkbook(wb_qa_post, file = paste0("Post_Processing_Review_",year,".xlsx") , overwrite = TRUE)
for(sheetindex in c(1:length(getSheetNames(paste0("Post_Processing_Review_",year,".xlsx"))))  )
{setColWidths(wb_qa_post, sheet = sheetindex, cols = 1:7, widths = 20)
}
saveWorkbook(wb_qa_post, file = paste0("Post_Processing_Review_",year,".xlsx") , overwrite = TRUE)

# remove post processing QA files
rm(IBA_match_status,IBA_match_success,IBA_negative_match, post_proc_ATM_temp_no_of_cols,post_proc_ATM_temp_no_of_records, post_proc_Global_family_order_count_excUKLCC,
   post_proc_Global_family_order_percentage_excUKLCC, post_proc_QA_AC_IATA_SCLFO, post_proc_QA_AC_IATA_SCLFO_InProd, post_proc_QA_AC_IATA_seats, post_proc_QA_Airport_SCLFO,
   post_proc_QA_SCLFO, post_proc_QA_SeatClass, post_proc_row_col_total, post_proc_UK_low_cost_order_count, post_proc_UK_low_cost_order_percentage,SPASM_Match_Airport_SCLFO)

# Step 5 - Prepare data for FMM and output into CSV -----

# 5.1 Create a contingency table cross tabulating unique aircraft types with fmm segments

# Ensure they are ordered as per fmm
ATM_temp$fmm_segment<- factor(ATM_temp$fmm_segment, levels=c("c1Sch","c2Sch",   "c3Sch",    "c4Sch" ,"c5Sch"    ,"c6Sch",
                                                             "c1Ch","c2Ch","c3Ch","c4Ch","c5Ch","c6Ch",
                                                             "c1NFC","c2NFC","c3NFC","c4NFC","c5NFC","c6NFC"))

fmm_table <- table(ATM_temp$IATA,ATM_temp$fmm_segment)
fmm_table <- fmm_table[rowSums(fmm_table)>=1,]
fmm_table <- as.data.frame.matrix(fmm_table)
```

```r

# 5.4.1 Create ATM age distributions for all service type and then for Sch, Ch and NFC individually

ATM_temp$Age_midYear_fct <- as.factor(ATM_temp$Age_midYear)

# All service type ATM age distribution

ATM_age_dist_All <- ATM_temp %>% group_by(IATA, Age_midYear_fct) %>%
  summarise(n=n()) %>% spread(Age_midYear_fct,n, drop = FALSE, fill = 0)

# Sch population ATM age distribution
ATM_age_dist_Sch <- ATM_temp %>% filter(fmm_service_type == "Sch") %>% group_by(IATA, Age_midYear_fct) %>%
summarise(n=n()) %>% spread(Age_midYear_fct, n, drop = FALSE, fill = 0)

# Ch population ATM age distribution
ATM_age_dist_Ch <- ATM_temp %>% filter(fmm_service_type == "Ch") %>% group_by(IATA, Age_midYear_fct) %>%
  summarise(n=n()) %>% spread(Age_midYear_fct, n, drop = FALSE, fill = 0)

# NFC population ATM age distribution
ATM_age_dist_NFC <- ATM_temp %>% filter(fmm_service_type == "NFC") %>% group_by(IATA, Age_midYear_fct) %>%
  summarise(n=n()) %>% spread(Age_midYear_fct, n, drop = FALSE, fill = 0)

# 5.4.2 Create Aircraft age distributions by registration code for all service type and then for Sch, Ch and NFC individually

# All service type aircraft age distribution by type and registration

Reg_age_dist_All <- ATM_temp %>%
  group_by(IATA,Age_midYear_fct) %>%
  summarise(n=n_distinct(aircraft_reg)) %>%
  spread(Age_midYear_fct,n, drop = FALSE, fill = 0)

# Sch service type aircraft age distribution by type and registration
Reg_age_dist_Sch <- ATM_temp %>% filter(fmm_service_type == "Sch") %>%
  group_by(IATA,Age_midYear_fct) %>%
  summarise(n=n_distinct(aircraft_reg)) %>%
  spread(Age_midYear_fct,n, drop = FALSE, fill = 0)


# Ch service type aircraft age distribution by type and registration
Reg_age_dist_Ch <- ATM_temp %>% filter(fmm_service_type == "Ch") %>%
  group_by(IATA,Age_midYear_fct) %>%
  summarise(n=n_distinct(aircraft_reg)) %>%
  spread(Age_midYear_fct,n, drop = FALSE, fill = 0)


# Sch service type aircraft age distribution by type and registration
Reg_age_dist_NFC <- ATM_temp %>% filter(fmm_service_type == "NFC") %>%
  group_by(IATA,Age_midYear_fct) %>%
  summarise(n=n_distinct(aircraft_reg)) %>%
  spread(Age_midYear_fct,n, drop = FALSE, fill = 0)




# 5.6 create workbooks for fmm parameters

wb_fmm_table <- createWorkbook(creator = "dtathgur", title = paste0("FMM_Inputs_",year))
# add fmm contingency table
addWorksheet(wb_fmm_table, paste0("FMM_",year) )

#add ATM age distributions
addWorksheet(wb_fmm_table, paste0("ATM_age_dist_All_",year) )
addWorksheet(wb_fmm_table, paste0("ATM_age_dist_Ch_",year) )
addWorksheet(wb_fmm_table, paste0("ATM_age_dist_NFC_",year) )
addWorksheet(wb_fmm_table, paste0("ATM_age_dist_Sch_",year) )

#add Reg age distributions
addWorksheet(wb_fmm_table, paste0("Reg_age_dist_All_",year) )
addWorksheet(wb_fmm_table, paste0("Reg_age_dist_Ch_",year) )
addWorksheet(wb_fmm_table, paste0("Reg_age_dist_NFC_",year) )
addWorksheet(wb_fmm_table, paste0("Reg_age_dist_Sch_",year) )

writeData(wb_fmm_table, paste0("FMM_",year),fmm_table ,  startCol = 1, startRow = 3, xy = NULL,
               colNames = TRUE, rowNames = TRUE, headerStyle = NULL,
               keepNA = TRUE, sep = ", ")

writeData(wb_fmm_table, paste0("ATM_age_dist_All_",year), ATM_age_dist_All ,  startCol = 1, startRow = 3, xy = NULL,
          colNames = TRUE, rowNames = TRUE, headerStyle = NULL,
          keepNA = TRUE, sep = ", ")

writeData(wb_fmm_table, paste0("ATM_age_dist_Ch_",year), ATM_age_dist_Ch ,  startCol = 1, startRow = 3, xy = NULL,
          colNames = TRUE, rowNames = TRUE, headerStyle = NULL,
          keepNA = TRUE, sep = ", ")

writeData(wb_fmm_table, paste0("ATM_age_dist_NFC_",year), ATM_age_dist_NFC ,  startCol = 1, startRow = 3, xy = NULL,
          colNames = TRUE, rowNames = TRUE, headerStyle = NULL,
          keepNA = TRUE, sep = ", ")

writeData(wb_fmm_table, paste0("ATM_age_dist_Sch_",year), ATM_age_dist_Sch ,  startCol = 1, startRow = 3, xy = NULL,
          colNames = TRUE, rowNames = TRUE, headerStyle = NULL,
          keepNA = TRUE, sep = ", ")

writeData(wb_fmm_table, paste0("Reg_age_dist_All_",year), Reg_age_dist_All ,  startCol = 1, startRow = 3, xy = NULL,
          colNames = TRUE, rowNames = TRUE, headerStyle = NULL,
          keepNA = TRUE, sep = ", ")

writeData(wb_fmm_table, paste0("Reg_age_dist_Ch_",year), Reg_age_dist_Ch ,  startCol = 1, startRow = 3, xy = NULL,
          colNames = TRUE, rowNames = TRUE, headerStyle = NULL,
          keepNA = TRUE, sep = ", ")

writeData(wb_fmm_table, paste0("Reg_age_dist_NFC_",year), Reg_age_dist_NFC ,  startCol = 1, startRow = 3, xy = NULL,
          colNames = TRUE, rowNames = TRUE, headerStyle = NULL,
          keepNA = TRUE, sep = ", ")

writeData(wb_fmm_table, paste0("Reg_age_dist_Sch_",year), Reg_age_dist_Sch ,  startCol = 1, startRow = 3, xy = NULL,
          colNames = TRUE, rowNames = TRUE, headerStyle = NULL,
          keepNA = TRUE, sep = ", ")

# save workbooks

saveWorkbook(wb_fmm_table, file = paste0("FMM_Inputs_",year,".xlsx") , overwrite = TRUE)

# remove temporary files
rm(Reg_age_dist_All,Reg_age_dist_Ch,Reg_age_dist_NFC,Reg_age_dist_Sch, ATM_retirement_All, ATM_retirement_Ch,
ATM_retirement_NFC, ATM_retirement_Sch,
   fmm_table, OECD_countries, UK_modelled_airports)
```