

```

1  ## This is a sequenced set of routines to be followed for Fleet Modelling Purposes
2  ## This version has been amended to help with code surgery Coffee and Coding - 26th of June 2019
3  ## Import CAA movement data, IBA fleet and other lookup data - which is cleaned, merged and then processed (some steps are
4  ## There are review or QA steps at several points - please check working directory for spreadsheets that are produced during
   execution
5  ## For details on suitable use of the underlying movements data, please speak to the Stats team
6  ## Dharmender Tathgur AMA January 2019
7  ## This code was amended after MH carried out an (internal) QA process in April 2019
8
9  # Step 0 - Global declarations, file locations and libraries ---- Please double check locations and input year on a
   regular basis
10 ## worksheets may have been added, amended or removed from the lookup files so be aware
11
12 ## Global definition lists ----
13
14 OECD_countries = c("Australia", "Austria", "Belgium", "Canada", "Chile", "Czech
   Republic", "Denmark", "Estonia", "Finland", "France", "Germany", "Greece", "Hungary", "Iceland", "Ireland",
15 "Israel", "Italy", "Japan", "Korea, Republic of", "Luxembourg", "Mexico", "Netherlands", "New
   Zealand", "Norway", "Poland", "Portugal", "Slovakia", "Slovenia", "Spain",
16 "Sweden", "Switzerland", "Turkey", "United Kingdom", "United States")
17
18 UK_modelled_airports = c(461:487, 491:492) # For further detail see aviation model documentation - 599 is a
   NON MODELLED UK AIRPORT
19
20
21 # Working directory - not necessarily the location of datasets or lookup files
22 setwd("<INSERT DIR HERE>")
23
24 #Input year here
25 year <- c("<INSERT YEAR HERE>")
26
27 #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
28 ATM_file <- "<INSERT FILE HERE>"
29
30 #Input location of aviation model lookup files
31 IATA_ICAO_Aircraft_codes_file <- "<INSERT FILE HERE>"
32 WorldZoneCon_file <- "<INSERT FILE HERE>"
33
34 #Location of Aircraft inventory by registration code as of the beginning and end of input year to allow maximum coverage
35 #Download instructions and the order of columns are in the following spreadsheets
36 #Aircraft registrations change across a year and the following method is able to catch as much as it can
37 #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
38 Aircraft_inventory_file_Jan <- "<INSERT FILE HERE>"
39 Aircraft_inventory_file_Dec <- "<INSERT FILE HERE>"
40 Aircraft_inventory_file_Missing <- "<INSERT FILE HERE>"
41
42 # Location of Aircraft Backlog file as of end of input year again for maximum coverage
43 Aircraft_order_file <- "<INSERT FILE HERE>"
44
45 #Import following libraries to carry out relevant functions
46 library(tidyverse)
47 library(data.table)
48 library(openxlsx)
49 library(janitor)
50 library(readxl)
51
52 # Step 1 - Import files and lookups into R Environment -----
53
54 ## Step 1.1 Import aircraft inventory files downloaded from IBA fleets into memory and the missing data manually input
55 ## Please refer to the procedure document to find out the description of the IBA datasets below and how to extract it
   CORRECTLY
56
57 Aircraft_inventory_Jan <- read_excel(Aircraft_inventory_file_Jan, sheet = "Report",
58 col_types = c("text", "text", "text",
59 "text", "text", "text", "text", "numeric",
60 "text", "text", "text", "text", "text",
61 "numeric", "numeric", "text"),
62 na = "NA", skip = 9)
63
64 Aircraft_inventory_Dec <- read_excel(Aircraft_inventory_file_Dec, sheet = "Report",
65 col_types = c("text", "text", "text",
66 "text", "text", "text", "text", "numeric",
67 "text", "text", "text", "text", "text",
68 "numeric", "numeric", "text"),
69 na = "NA", skip = 9)
70
71 Aircraft_inventory_Missing <- read_excel(Aircraft_inventory_file_Missing, sheet = "Report",
72 col_types = c("text", "text", "text",
73 "text", "text", "text", "text", "numeric",
74 "text", "text", "text", "text", "text",
75 "numeric", "numeric", "text"),
76 na = "NA", skip = 9)
77
78 Aircraft_order <- read_excel(Aircraft_order_file, sheet = "Report", na = "", skip = 9)
79
80
81 ## Step 1.2 Import CAA movements file into memory
82 ## Read in the (caa) ATM data correctly formatted ~ Please see metadata in the source directory
83 ## The following method should also apply to all ATM files from CAA - for all years - Hence the temp suffix
84 ## In case of any discrepancy or error/warning thrown, Please speak to the stats team
85
86 ATM_temp <- readr::read_csv(ATM_file,
87 col_types = cols(
88 year = col_integer()
89 ,month = col_factor(levels = c("Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul",
   "Aug", "Sep", "Oct", "Nov", "Dec"), ordered = TRUE)
90 ,month_no = col_integer()
91 ,reporting_airport = col_character()
92 ,reporting_airport_code = col_character()
93 ,lastnext_airport = col_character()
94 ,lastnext_country = col_character()
95 ,origdest_airport = col_character()

```

```

96         ,origdest_country = col_character()
97         ,aircraft_name = col_character()
98         ,aircraft_reg = col_character()
99         ,aoc_holder_name = col_character()
100        ,consent_to_publish = col_character()
101        ,service_type = col_character()
102        ,operation_type = col_character()
103        ,direction_type = col_character()
104        ,atm_date = col_date(format = "%Y-%m-%d")
105        ,weekday1 = col_factor(levels = c("Sunday", "Monday", "Tuesday", "Wednesday",
106        "Thursday", "Friday", "Saturday"), ordered = TRUE)
107        ,weekday2 = col_factor(levels = c("Sun", "Mon", "Tue", "Wed", "Thu", "Fri",
108        "Sat"), ordered = TRUE)
109        ,time = col_time(format = "%H:%M:%S")
110        ,time_hour = col_integer()
111        ,time_band = col_character()
112        ,atm_count = col_integer()
113        ,lastnext_sectordistance_km = col_integer()
114        ,available_seats = col_integer()
115        ,pax_terminal_dom = col_integer()
116        ,pax_terminal_int = col_integer()
117        ,pax_terminal_total = col_integer()
118        ,pax_transit_total = col_integer()
119        ,freight_weight_kg = col_integer()
120        ,mail_weight_kg = col_integer()
121        ,cargo_weight_kg = col_integer()
122        ,freight_weight_tonnes = col_double()
123        ,mail_weight_tonnes = col_double()
124        ,cargo_weight_tonnes = col_double()
125      ),
126      trim_ws=TRUE
127    )
128
129 ## Step 1.3 Import aviation model lookup files into memory
130 SPASMZone <- read_excel(WorldZoneCon_file, sheet = "WorldZoneCon", range = cell_cols("A:B"))
131 CAAICAOIATA <- read_excel(IATA_ICAO_Aircraft_codes_file,sheet = "CAAicaoIATA",range = cell_cols("A:F"))
132
133 # Step 1.4 Clean - Remove records with blank or NA values - these incomplete, mostly legacy, records are of no use
134 # Remove records with blank or NA values for Registration in aircraft inventory
135 Aircraft_inventory_Jan <- Aircraft_inventory_Jan[!is.na(Aircraft_inventory_Jan$Registration),]
136 Aircraft_inventory_Dec <- Aircraft_inventory_Dec[!is.na(Aircraft_inventory_Dec$Registration),]
137 Aircraft_inventory_Missing <- Aircraft_inventory_Missing[!is.na(Aircraft_inventory_Missing$Registration),]
138
139 # Step 1.5 Rename specific cols in lookup files to prevent duplication and ease understanding
140 names(SPASMZone)[names(SPASMZone) == "Name"] <- "AP_name"
141 names(SPASMZone)[names(SPASMZone) == "SPASMZONE"] <- "SPASM_zone"
142
143 # Step 1.6 Text cleaning routines for consistency with CAA movements file
144 # the following commands ensure that ALL values in specific columns are CAPS & without trailing/leading spaces.
145 Aircraft_inventory_Jan$Registration <- trimws(toupper(c(Aircraft_inventory_Jan$Registration)))
146 Aircraft_inventory_Dec$Registration <- trimws(toupper(c(Aircraft_inventory_Dec$Registration)))
147 Aircraft_inventory_Missing$Registration <- trimws(toupper(c(Aircraft_inventory_Missing$Registration)))
148
149 ATM_temp$lastnext_airport <- trimws(toupper(c(ATM_temp$lastnext_airport)))
150 ATM_temp$origdest_airport <- trimws(toupper(c(ATM_temp$origdest_airport)))
151 ATM_temp$aircraft_name <- trimws(toupper(c(ATM_temp$aircraft_name)))
152 ATM_temp$aircraft_reg <- trimws(toupper(c(ATM_temp$aircraft_reg)))
153
154 SPASMZone$AP_name <- toupper(SPASMZone$AP_name)
155
156 # Remove hyphens and pluses from aircraft registrations in both to provide better merge later on
157
158 Aircraft_inventory_Jan$Registration <- gsub("-", "", Aircraft_inventory_Jan$Registration)
159 Aircraft_inventory_Jan$Registration <- gsub("\\+", "", Aircraft_inventory_Jan$Registration)
160 Aircraft_inventory_Dec$Registration <- gsub("-", "", Aircraft_inventory_Dec$Registration)
161 Aircraft_inventory_Dec$Registration <- gsub("\\+", "", Aircraft_inventory_Dec$Registration)
162 Aircraft_inventory_Missing$Registration <- gsub("-", "", Aircraft_inventory_Missing$Registration)
163 Aircraft_inventory_Missing$Registration <- gsub("\\+", "", Aircraft_inventory_Missing$Registration)
164
165 ATM_temp$aircraft_reg <- gsub("-", "", ATM_temp$aircraft_reg)
166
167 # Step 1.7 Harmonising IBA aircraft inventory - Create a new col called Age_midYear
168 # 1.7.1 Add half a year to January and subtract half year to Dec search ages to represent mid year age profile
169
170 Aircraft_inventory_Jan$Age_midYear <- Aircraft_inventory_Jan$`Search Age` + 0.5
171 Aircraft_inventory_Dec$Age_midYear <- Aircraft_inventory_Dec$`Search Age` - 0.5
172 Aircraft_inventory_Missing$Age_midYear <- Aircraft_inventory_Missing$`Search Age`
173
174 # 1.7.2 Consolidate Dec, Jan and Missing. Missing given the highest priority followed by Dec and then Jan
175 # rename age col appropriately and round to the nearest integer
176
177 Aircraft_inventory <- bind_rows(Aircraft_inventory_Dec, Aircraft_inventory_Jan[!(Aircraft_inventory_Jan$Registration
178 %in% Aircraft_inventory_Dec$Registration),])
179 Aircraft_inventory <- bind_rows(Aircraft_inventory_Missing, Aircraft_inventory[!(Aircraft_inventory$Registration
180 %in% Aircraft_inventory_Missing$Registration),])
181 Aircraft_inventory$Age_midYear <- round(Aircraft_inventory$Age_midYear,0)
182
183 #1.7.3 The resulting harmonised inventory may have a few duplicate values for Registration which means the following order of
184 importance should be carried out for distinct values
185
186 order_of_importance <- c("Active", "On Order", "Retained","Parted Out","Stored","Destroyed", "Retired","Parked","Damaged")
187
188 Aircraft_inventory <- Aircraft_inventory %>%
189   mutate(Status = factor(Status, order_of_importance)) %>%
190   arrange(Registration, Status) %>%
191   distinct(Registration, .keep_all = TRUE)
192
193 # Step 1.8 Set up factors - only for columns that are of cateogorical nature - speeds up processing later on and for
194 producing plots
195 Aircraft_inventory$Registration <- as.factor(Aircraft_inventory$Registration)
196 Aircraft_inventory$`Aircraft Class` <- as.factor(Aircraft_inventory$`Aircraft Class`)
197 Aircraft_inventory$`Aircraft Family` <- as.factor(Aircraft_inventory$`Aircraft Family`)
198 Aircraft_inventory$`Aircraft Manufacturer` <-as.factor(Aircraft_inventory$`Aircraft Manufacturer`)
199 Aircraft_inventory$`Aircraft Series` <-as.factor(Aircraft_inventory$`Aircraft Series`)

```

```

197 Aircraft_inventory$`Aircraft Type` <- as.factor(Aircraft_inventory$`Aircraft Type`)
198 Aircraft_inventory$`Operator Country` <-as.factor(Aircraft_inventory$`Operator Country`)
199 Aircraft_inventory$`Operator Region` <- as.factor(Aircraft_inventory$`Operator Region`)
200 Aircraft_inventory$`Operator Subregion`<- as.factor(Aircraft_inventory$`Operator Subregion`)
201 Aircraft_inventory$Operator<-as.factor(Aircraft_inventory$Operator)
202 Aircraft_inventory$Status<-as.factor(Aircraft_inventory$Status)
203
204 Aircraft_order$`Aircraft Family` <- as.factor(Aircraft_order$`Aircraft Family`)
205
206 ATM_temp$month <- as.factor(ATM_temp$month)
207 ATM_temp$month_no <- as.factor(ATM_temp$month_no)
208 ATM_temp$reporting_airport <- as.factor(ATM_temp$reporting_airport)
209 ATM_temp$reporting_airport_code <- as.factor(ATM_temp$reporting_airport_code)
210 ATM_temp$lastnext_airport <- as.factor(ATM_temp$lastnext_airport)
211 ATM_temp$lastnext_country <- as.factor(ATM_temp$lastnext_country)
212 ATM_temp$origdest_airport <- as.factor(ATM_temp$origdest_airport)
213 ATM_temp$origdest_country <- as.factor(ATM_temp$origdest_country)
214 ATM_temp$aoc_holder_name <- as.factor(ATM_temp$aoc_holder_name)
215 ATM_temp$service_type <- as.factor(ATM_temp$service_type)
216 ATM_temp$operation_type <- as.factor(ATM_temp$operation_type)
217 ATM_temp$direction_type <- as.factor(ATM_temp$direction_type)
218 ATM_temp$weekday1 <- as.factor(ATM_temp$weekday1)
219 ATM_temp$weekday2 <- as.factor(ATM_temp$weekday2)
220 ATM_temp$time_hour <- as.factor(ATM_temp$time_hour)
221 ATM_temp$aircraft_name <- as.factor(ATM_temp$aircraft_name )
222 ATM_temp$aircraft_reg <- as.factor(ATM_temp$aircraft_reg)
223 ATM_temp$consent_to_publish <- as.factor(ATM_temp$consent_to_publish) #All these flights form part
of our analysis
224
225
226 CAAICAOIATA$CAA_Acf_Name <- as.factor(CAAICAOIATA$CAA_Acf_Name)
227 CAAICAOIATA$`Aircraft ICAO Code` <- as.factor(CAAICAOIATA$`Aircraft ICAO Code`)
228 CAAICAOIATA$IATA <- as.factor(CAAICAOIATA$IATA)
229 CAAICAOIATA$Fmm_Haul <- as.factor(CAAICAOIATA$Fmm_Haul)
230 CAAICAOIATA$Aircraft_Manufacturer <- as.factor(CAAICAOIATA$Aircraft_Manufacturer)
231 CAAICAOIATA$In_Production <- as.factor(CAAICAOIATA$In_Production)
232
233 # Step 1.9 Remove any temporary files or locations
234
235 rm(Aircraft_inventory_file_Missing,Aircraft_inventory_file_Jan,Aircraft_inventory_file_Dec, Aircraft_order_file)
236
237
238
239
240 # Step 2 - Pre Processing Quality Assurance -----
241 ## Relevant data from input files in previous step are consolidated into dataframes and findings are saved in single .xlsx
file for review
242
243 #2.1.1 QA211 - High level integrity check - Get number of records and variables from CAA ATM movements file
244 no_of_records <- nrow(ATM_temp)
245 no_of_cols <- ncol(ATM_temp)
246 pre_proc_row_col_total <- rbind(no_of_cols,no_of_records)
247 colnames(pre_proc_row_col_total) <- "no_obs"
248
249 # 2.1.2 QA212 - Aggregating service_type and operation_type from CAA ATM movements file
250 pre_proc_QA_serv_oper <- ATM_temp %>%
251   group_by(service_type,operation_type) %>%
252   summarise("no_obs" = n()) %>%
253   arrange(desc(no_obs)) %>%
254   adorn_totals(where = "row")
255
256 # 2.1.3 QA213 - Aggregating aircraft_name from CAA ATM movements file
257 pre_proc_QA_aircraft_name <- ATM_temp %>%
258   group_by(aircraft_name) %>%
259   summarise("no_obs"=n(),"max_distance flown" = round(quantile(lastnext_sectordistance_km,0.95),0)) %>%
260   mutate(Haul = ifelse(max_distance_flown<3750, "Short", "Both")) %>%
261   arrange(desc(no_obs)) %>%
262   adorn_totals(where = "row")
263
264 # 2.1.4 QA214 - Aggregating year, month and month_no from CAA ATM movements file
265 pre_proc_QA_year_month <- ATM_temp %>%
266   group_by(year,month, month_no) %>%
267   summarise("no_obs" = n()) %>%
268   arrange(desc(no_obs)) %>%
269   adorn_totals(where = "row")
270
271 # 2.1.5 QA215 - Aggregating reporting_airport_code, reporting_airport and direction_type from CAA ATM movements file
272 pre_proc_QA_reporting_airport_direction <- ATM_temp %>%
273   group_by(reporting_airport_code, reporting_airport, direction_type) %>%
274   summarise("no_obs" = n()) %>%
275   arrange(desc(no_obs)) %>%
276   adorn_totals(where = "row")
277
278 # 2.1.6 QA216 - Aggregating aircraft registrations with available seats from CAA ATM movements file includes zero
available_seats and UNKNOWN reg
279 # Multiple occurrences of aircraft_reg AND aircraft_name are expected especially since available_seats are recorded
differently either by airports or operations
280 pre_proc_QA_reg_name_seat <- ATM_temp %>%
281   group_by(aircraft_reg, aircraft_name, available_seats) %>%
282   summarise("no_obs" = n()) %>%
283   arrange(desc(no_obs)) %>%
284   adorn_totals(where = "row")
285
286 # 2.1.7 QA217- Aggregating aircraft reg AND aircraft_name into unique record combinations summarised by seat info from CAA
ATM movements file
287 pre_proc_QA_reg_name_seat_analysis <- ATM_temp %>%
288   filter(available_seats>0 & aircraft_reg != "UNKNOWN") %>%
289   group_by(aircraft_reg, aircraft_name) %>%
290   summarise("min_seat" = min(available_seats),
291     "max_seat" = max(available_seats),
292     "gap_seat_values" = max_seat - min_seat ,
293     "mean_seat" = round(mean(available_seats),0),
294     "no_obs" = n()) %>%
295   arrange(desc(gap_seat_values)) %>%
296   adorn_totals(where = "row")

```



```

297
298
299 #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)
300
301
302 pre_proc_QA_unique_reg_name_seat_quartile_analysis <- ATM temp %>%
303   filter(available_seats>0 & aircraft_reg != "UNKNOWN") %>%
304   group_by(aircraft_reg) %>%
305   summarise( "lower quartile" = quantile(available_seats,0.25),
306             "upper quartile" = quantile(available_seats,0.75),
307             "difference quartile" = `upper quartile` - `lower quartile`,
308             "no_obs" = n(),
309             "order variable - product" = ifelse(`difference quartile` > 20 | `no_obs` > 1, `difference
quartile`*`no_obs`, 0),
310             "aircraft names" = paste(sort(unique(aircraft_name)), collapse=" , "),
311             "no aircraft names" = n_distinct(aircraft_name)
312           ) %>%
313   arrange(desc(`no aircraft names`)) %>%
314   adorn_totals(where = "row")
315
316 #2.2.1 QA221- Fleet by operator from harmonised IBA Aircraft inventory
317 pre_proc_fleet_by_oper <- Aircraft_inventory %>%
318   filter(Status == "Active") %>%
319   group_by(Operator) %>%
320   summarise("no_obs" = n()) %>%
321   arrange(desc(no_obs)) %>%
322   adorn_totals(where = "row")
323
324 #2.2.2 QA222- Fleet by type from harmonised IBA Aircraft inventory
325 pre_proc_fleet_by_type <- Aircraft_inventory %>%
326   filter(Status == "Active") %>%
327   group_by(`Operator Region`, `Aircraft Type`) %>%
328   summarise("no_obs" = n()) %>%
329   arrange(desc(no_obs)) %>%
330   adorn_totals(where = "row")
331
332 #2.2.3 QA223 - Global Fleet with aircraft type and basic age analysis from harmonised IBA Aircraft inventory
333 pre_proc_Global_fleet_by_oper_type <- Aircraft_inventory %>%
334   filter(Status == "Active") %>%
335   group_by(`Aircraft Type`) %>%
336   summarise("mean age (mid year)" = round(mean(Age_midYear),0),
337             "median age (mid year)" = round(median(Age_midYear),0),
338             "90th percentile age (mid year)" = round(quantile(Age_midYear, 0.90),0),
339             "95th percentile age (mid year)" = round(quantile(Age_midYear, 0.95),0),
340             "no_obs" = n() ) %>%
341   arrange(desc(no_obs)) %>%
342   adorn_totals(where = "row")
343
344 #2.2.4 QA224 - UK Fleet with aircraft type and basic age analysis from harmonised IBA Aircraft inventory
345 pre_proc_UK_fleet_by_oper_type <- Aircraft_inventory %>%
346   filter(Status == "Active" & `Operator Country` == "United Kingdom") %>%
347   group_by(`Aircraft Type`) %>%
348   summarise("mean age (mid year)" = round(mean(Age_midYear),0),
349             "median age (mid year)" = round(median(Age_midYear),0),
350             "90th percentile age (mid year)" = round(quantile(Age_midYear, 0.90),0),
351             "95th percentile age (mid year)" = round(quantile(Age_midYear, 0.95),0),
352             "no_obs" = n() ) %>%
353   arrange(desc(no_obs)) %>%
354   adorn_totals(where = "row")
355
356 #2.2.5 QA225 - WE Fleet with aircraft type and basic age analysis from harmonised IBA Aircraft inventory
357 pre_proc_WE_fleet_by_oper_type <- Aircraft_inventory %>%
358   filter(Status == "Active" & `Operator Subregion` == "Western Europe") %>%
359   group_by(`Aircraft Type`) %>%
360   summarise("mean age (mid year)" = round(mean(Age_midYear),0),
361             "median age (mid year)" = round(median(Age_midYear),0),
362             "90th percentile age (mid year)" = round(quantile(Age_midYear, 0.90),0),
363             "95th percentile age (mid year)" = round(quantile(Age_midYear, 0.95),0),
364             "no_obs" = n() ) %>%
365   arrange(desc(no_obs)) %>%
366   adorn_totals(where = "row")
367
368 #2.2.6 QA226 - OECD Fleet with aircraft type and basic age analysis from harmonised IBA Aircraft inventory
369 pre_proc_OECD_fleet_by_oper_type <- Aircraft_inventory %>%
370   filter(Status == "Active" & `Operator Country` %in% OECD_countries) %>%
371   group_by(`Aircraft Type`) %>%
372   summarise("mean age (mid year)" = round(mean(Age_midYear),0),
373             "median age (mid year)" = round(median(Age_midYear),0),
374             "90th percentile age (mid year)" = round(quantile(Age_midYear, 0.90),0),
375             "95th percentile age (mid year)" = round(quantile(Age_midYear, 0.95),0),
376             "no_obs" = n() ) %>%
377   arrange(desc(no_obs)) %>%
378   adorn_totals(where = "row")
379
380 #2.2.7 QA227 - Basic in-year Global retirement analysis from harmonised IBA Aircraft inventory
381 pre_proc_age_at_Global_retirement <- Aircraft_inventory %>%
382   filter(Status == "Retired") %>%
383   group_by(`Aircraft Family`) %>%
384   summarise ("median age" = round(median(Age_midYear),0)
385             , "min age" = round(min(Age_midYear),0),
386             "max age" = round(max(Age_midYear),0),
387             "no_obs" = n() ) %>%
388   arrange(`Aircraft Family`)
389
390 #2.2.8 QA228- Basic in-year UK retirement analysis from harmonised IBA Aircraft inventory
391 pre_proc_age_at_UK_retirement <- Aircraft_inventory %>%
392   filter(Status == "Retired" & `Operator Country` == "United Kingdom") %>%
393   group_by(`Aircraft Family`) %>%
394   summarise ("median age" = round(median(Age_midYear),0)
395             , "min age" = round(min(Age_midYear),0),
396             "max age" = round(max(Age_midYear),0),
397             "no_obs" = n() ) %>%
398   arrange(`Aircraft Family`)
399

```

```

400 #2.2.9 QA229- Basic in-year WE retirement analysis from harmonised IBA Aircraft inventory
401 pre_proc_age_at_WE_retirement <- Aircraft_inventory %>%
402   filter(Status == "Retired" & `Operator Subregion` == "Western Europe") %>%
403   group_by(`Aircraft Family`) %>%
404   summarise ("median age" = round(median(Age_midYear),0) ,
405             "min age" = round(min(Age_midYear),0),
406             "max age" = round(max(Age_midYear),0),
407             "no_obs" = n() ) %>%
408   arrange(`Aircraft Family`)
409
410 #2.2.10 QA2210- Basic in-year OECD retirement analysis from harmonised IBA Aircraft inventory
411 pre_proc_age_at_OECD_retirement <- Aircraft_inventory %>%
412   filter(Status == "Retired" & `Operator Country` %in% OECD_countries) %>%
413   group_by(`Aircraft Family`) %>%
414   summarise ("median age" = round(median(Age_midYear),0) ,
415             "min age" = round(min(Age_midYear),0),
416             "max age" = round(max(Age_midYear),0),
417             "no_obs" = n() ) %>%
418   arrange(`Aircraft Family`)
419
420
421
422
423 # 2.3.1 QA231- Global aircraft orders absolute values by years using IBA Aircraft order
424 pre_proc_Global_order_count <- Aircraft_order %>%
425   filter(`Build Year`>2016) %>%
426   group_by(`Aircraft Family`, `Aircraft Model`, `Build Year`) %>%
427   summarise(no_obs = n()) %>%
428   spread(`Build Year`, no_obs, fill=0) %>%
429   adorn_totals(where = c("row", "col"))
430
431 # 2.3.2 QA232 - Global aircraft orders percentages by years using IBA Aircraft order
432 pre_proc_Global_order_percentage <- Aircraft_order %>%
433   filter(`Build Year`>2016) %>%
434   group_by(`Aircraft Family`, `Aircraft Model`, `Build Year`) %>%
435   summarise(no_obs = n()) %>%
436   spread(`Build Year`, no_obs, fill=0) %>%
437   adorn_percentages(denominator = "col") %>%
438   mutate_if(is.numeric, round, 2)
439
440 # 2.3.3 QA233- UK aircraft orders absolute values by years using IBA Aircraft order
441 pre_proc_UK_order_count <- Aircraft_order %>%
442   filter(`Build Year`>2016 & `Operator Country` == "United Kingdom") %>%
443   group_by(`Aircraft Family`, `Aircraft Model`, `Build Year`) %>%
444   summarise(no_obs = n()) %>%
445   spread(`Build Year`, no_obs, fill=0) %>%
446   adorn_totals(where = c("row", "col"))
447
448 # 2.3.4 QA234 - UK aircraft orders percentages by years using IBA Aircraft order
449 pre_proc_UK_order_percentage <- Aircraft_order %>%
450   filter(`Build Year`>2016 & `Operator Country` == "United Kingdom") %>%
451   group_by(`Aircraft Family`, `Aircraft Model`, `Build Year`) %>%
452   summarise(no_obs = n()) %>%
453   spread(`Build Year`, no_obs, fill=0) %>%
454   adorn_percentages(denominator = "col") %>%
455   mutate_if(is.numeric, round, 2)
456
457 # 2.3.5 QA235- WE aircraft orders absolute values by years using IBA Aircraft order
458 pre_proc_WE_order_count <- Aircraft_order %>%
459   filter(`Build Year`>2016 & `Operator Subregion` == "Western Europe") %>%
460   group_by(`Aircraft Family`, `Aircraft Model`, `Build Year`) %>%
461   summarise(no_obs = n()) %>%
462   spread(`Build Year`, no_obs, fill=0) %>%
463   adorn_totals(where = c("row", "col"))
464
465 # 2.3.6 QA236 - WE aircraft orders percentages by years using IBA Aircraft order
466 pre_proc_WE_order_percentage <- Aircraft_order %>%
467   filter(`Build Year`>2016 & `Operator Subregion` == "Western Europe") %>%
468   group_by(`Aircraft Family`, `Aircraft Model`, `Build Year`) %>%
469   summarise(no_obs = n()) %>%
470   spread(`Build Year`, no_obs, fill=0) %>%
471   adorn_percentages(denominator = "col") %>%
472   mutate_if(is.numeric, round, 2)
473
474 # 2.3.7 QA237- OECD aircraft orders absolute values by years using IBA Aircraft order
475 pre_proc_OECD_order_count <- Aircraft_order %>%
476   filter(`Build Year`>2016 & `Operator Country` %in% OECD_countries) %>%
477   group_by(`Aircraft Family`, `Aircraft Model`, `Build Year`) %>%
478   summarise(no_obs = n()) %>%
479   spread(`Build Year`, no_obs, fill=0) %>%
480   adorn_totals(where = c("row", "col"))
481
482 # 2.3.8 QA238 - OECD aircraft orders percentages by years using IBA Aircraft order
483 pre_proc_OECD_order_percentage <- Aircraft_order %>%
484   filter(`Build Year`>2016 & `Operator Country` %in% OECD_countries) %>%
485   group_by(`Aircraft Family`, `Aircraft Model`, `Build Year`) %>%
486   summarise(no_obs = n()) %>%
487   spread(`Build Year`, no_obs, fill=0) %>%
488   adorn_percentages(denominator = "col") %>%
489   mutate_if(is.numeric, round, 2)
490
491
492
493
494
495 # 2.2 Write PRE PROCESSING Review excel file with dataframes above
496
497 # declare a Review workbook object
498 wb_qa <- createWorkbook(creator = "dtathgur", title = paste0("Review Pre Processing ", year))
499
500 # Add worksheets for each dataframe including three coversheets
501
502 addWorksheet(wb_qa, paste0("CAA_ATMS>>>"),
503             tabColour = "#00bfff" ) #210
504 addWorksheet(wb_qa, paste0("Topsheet_", year),

```

```

505         tabColour = "#00bfff" )           #211
506 addWorksheet(wb_qa, paste0("Serv_oper_",year),
507         tabColour = "#00bfff" )           #212
508 addWorksheet(wb_qa, paste0("Ac_name_",year),
509         tabColour = "#00bfff" )           #213
510 addWorksheet(wb_qa, paste0("Yr_mon_",year) ,
511         tabColour = "#00bfff")           #214
512 addWorksheet(wb_qa, paste0("Rep_arpt_dire_",year),
513         tabColour = "#00bfff" )           #215
514 addWorksheet(wb_qa, paste0("Reg_name_seat_",year),
515         tabColour = "#00bfff" )           #216
516 addWorksheet(wb_qa, paste0("Seat_analysis_",year),
517         tabColour = "#00bfff" )           #217
518 addWorksheet(wb_qa, paste0("Seat_quar_analysis_",year),
519         tabColour = "#00bfff" )           #218
520
521
522 addWorksheet(wb_qa, paste0("IBA_FLEETS>>>"),
523         tabColour = "#ff0040")           #220
524 addWorksheet(wb_qa, paste0("Fleet_by_operator_",year), tabColour = "#ff0040")           #221
525 addWorksheet(wb_qa, paste0("Fleet_by_type_",year), tabColour = "#ff0040")           #222
526 addWorksheet(wb_qa, paste0("Global_Fleet_",year), tabColour = "#ff0040")           #223
527 addWorksheet(wb_qa, paste0("UK_Fleet_",year), tabColour = "#ff0040")           #224
528 addWorksheet(wb_qa, paste0("WE_Fleet_",year), tabColour = "#ff0040")           #225
529 addWorksheet(wb_qa, paste0("OECD_Fleet_",year), tabColour = "#ff0040")           #226
530 addWorksheet(wb_qa, paste0("Global_retirements_",year), tabColour = "#ff0040")           #227
531 addWorksheet(wb_qa, paste0("UK_retirements_",year), tabColour = "#ff0040")           #228
532 addWorksheet(wb_qa, paste0("WE_retirements_",year), tabColour = "#ff0040")           #229
533 addWorksheet(wb_qa, paste0("OECD_retirements_",year), tabColour = "#ff0040")           #2210
534
535
536 addWorksheet(wb_qa, paste0("IBA_ORDERS>>>"),
537         tabColour = "#90ee90")           #230
538 addWorksheet(wb_qa, paste0("Global_order_count_",year), tabColour = "#90ee90")           # QA231
539 addWorksheet(wb_qa, paste0("Global_order_percentage_",year), tabColour = "#90ee90")           # QA232
540 addWorksheet(wb_qa, paste0("UK_order_count_",year), tabColour = "#90ee90")           # QA233
541 addWorksheet(wb_qa, paste0("UK_order_percentage_",year), tabColour = "#90ee90")           # QA234
542 addWorksheet(wb_qa, paste0("WE_order_count_",year), tabColour = "#90ee90")           # QA235
543 addWorksheet(wb_qa, paste0("WE_order_percentage_",year), tabColour = "#90ee90")           # QA236
544 addWorksheet(wb_qa, paste0("OECD_order_count_",year), tabColour = "#90ee90")           # QA237
545 addWorksheet(wb_qa, paste0("OECD_order_percentage_",year), tabColour = "#90ee90")           # QA238
546
547
548
549
550
551 # Add datatable to each worksheet
552
553
554
555
556 writeData(wb_qa, paste0("Topsheet_",year),c(paste("CAA ATM Data "," - ",date())," No. of records and variables"), #211
557         startCol = 1, startRow = 1, xy = NULL,
558         borders = "rows",
559         borderColour = getOption("openxlsx.borderColour", "black"),
560         borderStyle = getOption("openxlsx.borderStyle", "thick")
561     )
562
563 writeData(wb_qa, paste0("Serv_oper_",year) , c(paste("CAA ATM Data ","- ",date())," service_type and operation_type"), #212
564         startCol = 1, startRow = 1, xy = NULL,
565         borders = "rows",
566         borderColour = getOption("openxlsx.borderColour", "black"),
567         borderStyle = getOption("openxlsx.borderStyle", "thick")
568     )
569
570 writeData(wb_qa, paste0("Ac_name_",year) , c(paste("CAA ATM Data "," - ",date())," aircraft_name, distance(s) flown and haul
571 calc (i.e. if max distance flown <3750 then short otherwise both) "),#213
572         startCol = 1, startRow = 1, xy = NULL,
573         borders = "rows",
574         borderColour = getOption("openxlsx.borderColour", "black"),
575         borderStyle = getOption("openxlsx.borderStyle", "thick")
576     )
577
578 writeData(wb_qa, paste0("Yr_mon_",year), c(paste("CAA ATM Data "," - ",date()),"year, month and month_no"), #214
579         startCol = 1, startRow = 1, xy = NULL,
580         borders = "rows",
581         borderColour = getOption("openxlsx.borderColour", "black"),
582         borderStyle = getOption("openxlsx.borderStyle", "thick")
583     )
584
585 writeData(wb_qa, paste0("Rep_arpt_dire_",year) , c(paste("CAA ATM Data "," - ",date())," reporting_airport_code, name and
586 direction_type"), #215
587         startCol = 1, startRow = 1, xy = NULL,
588         borders = "rows",
589         borderColour = getOption("openxlsx.borderColour", "black"),
590         borderStyle = getOption("openxlsx.borderStyle", "thick")
591     )
592
593 writeData(wb_qa, paste0("Reg_name_seat_",year), c(paste("CAA ATM Data "," - ",date())," ALL aircraft_reg and _name
594 combinations (inc. UNKNOWN and zeroes)
595 with available seats"), #216
596         startCol = 1, startRow = 1, xy = NULL,
597         borders = "rows",
598         borderColour = getOption("openxlsx.borderColour", "black"),
599         borderStyle = getOption("openxlsx.borderStyle", "thick")
600     )
601
602 writeData(wb_qa, paste0("Seat_analysis_",year) , c(paste("CAA ATM Data "," - ",date())," Basic seat analysis on aircraft_reg
603 and _name (exc. UNKNOWN and zeroes)
604 combinations "), #217
605         startCol = 1, startRow = 1, xy = NULL,
606         borders = "rows",
607         borderColour = getOption("openxlsx.borderColour", "black"),
608         borderStyle = getOption("openxlsx.borderStyle", "thick")
609     )
610

```



```

606
607 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)

608
609 startCol = 1, startRow = 1, xy = NULL,
610 borders = "rows",
611 borderColour = getOption("openxlsx.borderColour", "black"),
612 borderStyle = getOption("openxlsx.borderStyle", "thick")
613 )
614
615
616
617
618
619
620
621
622
623 writeData(wb_qa, paste0("Fleet_by_operator_",year) , c(paste("IBA Fleets "," - ",date())," Active global fleet by
operator"), #221
624 startCol = 1, startRow = 1, xy = NULL,
625 borders = "rows",
626 borderColour = getOption("openxlsx.borderColour", "black"),
627 borderStyle = getOption("openxlsx.borderStyle", "thick")
628 )
629
630 writeData(wb_qa, paste0("Fleet_by_type_",year) , c(paste("IBA Fleets "," - ",date())," Active global fleet by operator
region and aircraft type"), #222
631 startCol = 1, startRow = 1, xy = NULL,
632 borders = "rows",
633 borderColour = getOption("openxlsx.borderColour", "black"),
634 borderStyle = getOption("openxlsx.borderStyle", "thick")
635 )
636
637 writeData(wb_qa, paste0("Global_Fleet_",year) , c(paste("IBA Fleets "," - ",date())," Active global operators fleet by age
summaries as of mid-year"), #223
638 startCol = 1, startRow = 1, xy = NULL,
639 borders = "rows",
640 borderColour = getOption("openxlsx.borderColour", "black"),
641 borderStyle = getOption("openxlsx.borderStyle", "thick")
642 )
643
644 writeData(wb_qa, paste0("UK_Fleet_",year) , c(paste("IBA Fleets "," - ",date())," Active UK operators fleet by age summaries
as of mid-year"), #224
645 startCol = 1, startRow = 1, xy = NULL,
646 borders = "rows",
647 borderColour = getOption("openxlsx.borderColour", "black"),
648 borderStyle = getOption("openxlsx.borderStyle", "thick")
649 )
650
651 writeData(wb_qa, paste0("WE_Fleet_",year) , c(paste("IBA Fleets "," - ",date())," Active WE operators fleet by age summaries
as of mid-year"), #225
652 startCol = 1, startRow = 1, xy = NULL,
653 borders = "rows",
654 borderColour = getOption("openxlsx.borderColour", "black"),
655 borderStyle = getOption("openxlsx.borderStyle", "thick")
656 )
657
658 writeData(wb_qa, paste0("OECD_Fleet_",year) , c(paste("IBA Fleets "," - ",date())," Active OECD operators fleet by age
summaries as of mid-year"), #226
659 startCol = 1, startRow = 1, xy = NULL,
660 borders = "rows",
661 borderColour = getOption("openxlsx.borderColour", "black"),
662 borderStyle = getOption("openxlsx.borderStyle", "thick")
663 )
664
665 writeData(wb_qa, paste0("Global_retirements_",year) , c(paste("IBA Fleets "," - ",date()),"Basic global retirement
analysis"), #227
666 startCol = 1, startRow = 1, xy = NULL,
667 borders = "rows",
668 borderColour = getOption("openxlsx.borderColour", "black"),
669 borderStyle = getOption("openxlsx.borderStyle", "thick")
670 )
671
672 writeData(wb_qa, paste0("UK_retirements_",year) , c(paste("IBA Fleets "," - ",date()),"Basic UK retirement analysis"), #228
673 startCol = 1, startRow = 1, xy = NULL,
674 borders = "rows",
675 borderColour = getOption("openxlsx.borderColour", "black"),
676 borderStyle = getOption("openxlsx.borderStyle", "thick")
677 )
678
679 writeData(wb_qa, paste0("WE_retirements_",year) , c(paste("IBA Fleets "," - ",date())," Basic WE retirement analysis"), #229
680 startCol = 1, startRow = 1, xy = NULL,
681 borders = "rows",
682 borderColour = getOption("openxlsx.borderColour", "black"),
683 borderStyle = getOption("openxlsx.borderStyle", "thick")
684 )
685
686 writeData(wb_qa, paste0("OECD_retirements_",year) , c(paste("IBA Fleets "," - ",date())," Basic OECD retirement analysis"),
#2210
687 startCol = 1, startRow = 1, xy = NULL,
688 borders = "rows",
689 borderColour = getOption("openxlsx.borderColour", "black"),
690 borderStyle = getOption("openxlsx.borderStyle", "thick")
691 )
692
693 writeData(wb_qa, paste0("Global_order_count_",year) , c(paste("IBA Orders "," - ",date())," Global aircraft orders (no.) by
aircraft family and model"), #231
694 startCol = 1, startRow = 1, xy = NULL,
695 borders = "rows",
696 borderColour = getOption("openxlsx.borderColour", "black"),
697 borderStyle = getOption("openxlsx.borderStyle", "thick")
698 )
699
700 writeData(wb_qa, paste0("Global_order_percentage_",year) , c(paste("IBA Orders "," - ",date())," Global aircraft orders (pc)

```

```

by aircraft family and model"), #232
701     startCol = 1, startRow = 1, xy = NULL,
702     borders = "rows",
703     borderColour = getOption("openxlsx.borderColour", "black"),
704     borderStyle = getOption("openxlsx.borderStyle", "thick")
705 )
706
707 writeData(wb_qa, paste0("UK_order_count_",year) , c(paste("IBA Orders "," - ",date()), " UK operators aircraft orders (no.)
by aircraft family and model"),
708     startCol = 1, startRow = 1, xy = NULL,
709     borders = "rows",
710     borderColour = getOption("openxlsx.borderColour", "black"),
711     borderStyle = getOption("openxlsx.borderStyle", "thick")
712 )
713
714 writeData(wb_qa, paste0("UK_order_percentage_",year) , c(paste("IBA Orders "," - ",date()), " UK operators aircraft orders
(pc) by aircraft family and model"),
715     startCol = 1, startRow = 1, xy = NULL,
716     borders = "rows",
717     borderColour = getOption("openxlsx.borderColour", "black"),
718     borderStyle = getOption("openxlsx.borderStyle", "thick")
719 )
720
721 writeData(wb_qa, paste0("WE_order_count_",year) , c(paste("IBA Orders "," - ",date()), " WE operators aircraft orders (no.)
by aircraft family and model"),
722     startCol = 1, startRow = 1, xy = NULL,
723     borders = "rows",
724     borderColour = getOption("openxlsx.borderColour", "black"),
725     borderStyle = getOption("openxlsx.borderStyle", "thick")
726 )
727
728 writeData(wb_qa, paste0("WE_order_percentage_",year) , c(paste("IBA Orders "," - ",date()), " WE operators aircraft orders
(pc) by aircraft family and model"),
729     startCol = 1, startRow = 1, xy = NULL,
730     borders = "rows",
731     borderColour = getOption("openxlsx.borderColour", "black"),
732     borderStyle = getOption("openxlsx.borderStyle", "thick")
733 )
734
735 writeData(wb_qa, paste0("OECD_order_count_",year) , c(paste("IBA Orders "," - ",date()), " OECD operators aircraft orders
(no.) by aircraft family and model"),
736     startCol = 1, startRow = 1, xy = NULL,
737     borders = "rows",
738     borderColour = getOption("openxlsx.borderColour", "black"),
739     borderStyle = getOption("openxlsx.borderStyle", "thick")
740 )
741
742 writeData(wb_qa, paste0("OECD_order_percentage_",year), c(paste("IBA Orders "," - ",date()), " OECD operators aircraft orders
(pc) by aircraft family and model"),
743     startCol = 1, startRow = 1, xy = NULL,
744     borders = "rows",
745     borderColour = getOption("openxlsx.borderColour", "black"),
746     borderStyle = getOption("openxlsx.borderStyle", "thick")
747 )
748
749 # Add datatable to each worksheet
750
751 writeDataTable(wb_qa, paste0("Topsheet_",year),data.frame(pre_proc_row_col_total),
752     startCol = 1, startRow = 3, xy = NULL,
753     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium3",
754     tableName = NULL, headerStyle = NULL, withFilter = TRUE,
755     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
756     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
757
758 writeDataTable(wb_qa, paste0("Serv_oper_",year) , pre_proc_QA_serv_oper,
759     startCol = 1, startRow = 3, xy = NULL,
760     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium2",
761     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
762     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
763     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
764
765 writeDataTable(wb_qa, paste0("Ac_name_",year) , pre_proc_QA_aircraft_name, startCol = 1, startRow = 3, xy = NULL,
766     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium2",
767     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
768     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
769     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
770
771 writeDataTable(wb_qa, paste0("Yr_mon_",year), pre_proc_QA_year_month, startCol = 1, startRow = 3, xy = NULL,
772     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium2",
773     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
774     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
775     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
776
777 writeDataTable(wb_qa, paste0("Rep_arpt_dire_",year) , pre_proc_QA_reporting_airport_direction, startCol = 1, startRow = 3,
xy = NULL,
778     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium2",
779     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
780     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
781     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
782
783 writeDataTable(wb_qa, paste0("Reg_name_seat_",year), pre_proc_QA_reg_name_seat, startCol = 1, startRow = 3, xy = NULL,
784     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium2",
785     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
786     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
787     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
788
789 writeDataTable(wb_qa, paste0("Seat_analysis_",year) , pre_proc_QA_reg_name_seat_analysis , startCol = 1, startRow = 3, xy =
NULL,
790     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium2",
791     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
792     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
793     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
794
795
796 writeDataTable(wb_qa, paste0("Seat_quar_analysis_",year) , pre_proc_QA_unique_reg_name_seat_quartile_analysis , startCol = 1,

```



```

797     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium2",
798     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
799     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
800     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
801
802
803
804 writeDataTable(wb_qa, paste0("Fleet_by_operator_",year) , pre_proc_fleet_by_oper, startCol = 1, startRow = 3, xy = NULL,
805     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium3",
806     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
807     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
808     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
809
810 writeDataTable(wb_qa, paste0("Fleet_by_type_",year) , pre_proc_fleet_by_type, startCol = 1, startRow = 3, xy = NULL,
811     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
812     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
813     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
814     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
815
816 writeDataTable(wb_qa, paste0("Global_Fleet_",year) , pre_proc_Global_fleet_by_oper_type, startCol = 1, startRow = 3, xy = NULL,
817     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
818     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
819     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
820     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
821
822 writeDataTable(wb_qa, paste0("UK_Fleet_",year) , pre_proc_UK_fleet_by_oper_type, startCol = 1, startRow = 3, xy = NULL,
823     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
824     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
825     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
826     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
827
828 writeDataTable(wb_qa, paste0("WE_Fleet_",year) , pre_proc_WE_fleet_by_oper_type, startCol = 1, startRow = 3, xy = NULL,
829     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
830     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
831     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
832     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
833
834 writeDataTable(wb_qa, paste0("OECD_Fleet_",year) , pre_proc_OECD_fleet_by_oper_type, startCol = 1, startRow = 3, xy = NULL,
835     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
836     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
837     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
838     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
839
840 writeDataTable(wb_qa, paste0("Global_retirements_",year) , pre_proc_age_at_Global_retirement, startCol = 1, startRow = 3, xy
= NULL,
841     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
842     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
843     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
844     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
845
846 writeDataTable(wb_qa, paste0("UK_retirements_",year) ,pre_proc_age_at_UK_retirement, startCol = 1, startRow = 3, xy = NULL,
847     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
848     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
849     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
850     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
851
852 writeDataTable(wb_qa, paste0("WE_retirements_",year) ,pre_proc_age_at_WE_retirement, startCol = 1, startRow = 3, xy = NULL,
853     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
854     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
855     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
856     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
857
858 writeDataTable(wb_qa, paste0("OECD_retirements_",year) ,pre_proc_age_at_OECD_retirement, startCol = 1, startRow = 3, xy = NULL,
859     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
860     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
861     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
862     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
863
864 writeDataTable(wb_qa, paste0("Global_order_count_",year) , pre_proc_Global_order_count , startCol = 1, startRow = 3, xy = NULL,
865     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
866     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
867     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
868     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
869
870 writeDataTable(wb_qa, paste0("Global_order_percentage_",year) , pre_proc_Global_order_percentage , startCol = 1, startRow =
3, xy = NULL,
871     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium4",
872     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
873     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
874     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
875
876 writeDataTable(wb_qa, paste0("UK_order_count_",year) , pre_proc_UK_order_count, startCol = 1, startRow = 3, xy = NULL,
877     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
878     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
879     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
880     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
881
882 writeDataTable(wb_qa, paste0("UK_order_percentage_",year) ,pre_proc_UK_order_percentage, startCol = 1, startRow = 3, xy = NULL,
883     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
884     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
885     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
886     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
887
888 writeDataTable(wb_qa, paste0("WE_order_count_",year) , pre_proc_WE_order_count, startCol = 1, startRow = 3, xy = NULL,
889     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
890     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
891     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
892     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
893
894 writeDataTable(wb_qa, paste0("WE_order_percentage_",year) ,pre_proc_WE_order_percentage, startCol = 1, startRow = 3, xy = NULL,
895     colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
896     tableName = NULL, headerStyle = NULL, withFilter =TRUE,
897     keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
898     lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)

```

```

899
900 writeDataTable(wb_qa, paste0("OECD_order_count_",year) , pre_proc_OECD_order_count, startCol = 1, startRow = 3, xy = NULL,
901               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
902               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
903               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
904               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
905
906 writeDataTable(wb_qa, paste0("OECD_order_percentage_",year) ,pre_proc_OECD_order_percentage, startCol = 1, startRow = 3, xy =
NULL,
907               colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
908               tableName = NULL, headerStyle = NULL, withFilter =TRUE,
909               keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
910               lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
911
912
913 # Save into a workbook and increase size of columns for easy read
914
915 saveWorkbook(wb_qa, file = paste0("Pre_processing_review_",year,".xlsx") , overwrite = TRUE)
916 for(sheetindex in c(1:length(getSheetNames(paste0("Pre_processing_review_",year,".xlsx")))) )
917 {setColWidths(wb_qa, sheet = sheetindex, cols = 1:7, widths = 20)
918 }
919 saveWorkbook(wb_qa, file = paste0("Pre_processing_review_",year,".xlsx") , overwrite = TRUE)
920
921 # Remove pre_proc dataframes as no longer needed
922 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,
923    pre_proc_QA_year_month, pre_proc_QA_reporting_airport_direction, pre_proc_QA_reg_name_seat,
924    pre_proc_QA_reg_name_seat_analysis, pre_proc_QA_unique_reg_name_seat_quartile_analysis,
925    pre_proc_fleet_by_oper,pre_proc_fleet_by_type, pre_proc_Global_fleet_by_oper_type, pre_proc_UK_fleet_by_oper_type,
926    pre_proc_WE_fleet_by_oper_type, pre_proc_OECD_fleet_by_oper_type,
927    pre_proc_age_at_Global_retirement, pre_proc_age_at_UK_retirement, pre_proc_age_at_WE_retirement,
928    pre_proc_age_at_OECD_retirement, pre_proc_Global_order_count ,
929    pre_proc_Global_order_percentage, pre_proc_UK_order_count, pre_proc_UK_order_percentage, pre_proc_WE_order_count,
930    pre_proc_WE_order_percentage,
931    pre_proc_OECD_order_count, pre_proc_OECD_order_percentage)
932
933
934 # Step 3 - Process data by merging, cutting and applying heuristics -----
935
936 # 3.1 For all ATMs - merging reporting and lastnext airport names for SPASM zones
937 ATM_temp <- merge(ATM_temp,SPASMZone , by.x="reporting_airport", by.y ="AP_name", all.x=TRUE )
938 names(ATM_temp)[names(ATM_temp) == "SPASM_zone"] <- "reporting_spasm_zone"
939 ATM_temp <- merge(ATM_temp,SPASMZone , by.x="lastnext_airport", by.y ="AP_name", all.x=TRUE )
940 names(ATM_temp)[names(ATM_temp) == "SPASM_zone"] <- "lastnext_spasm_zone"
941
942 # 3.2 Merging further FMM aircraft info from CAAICAOIATA (lookup file for aircraft)
943 ATM_temp <- merge(ATM_temp, CAAICAOIATA, by.x ="aircraft_name" , by.y = "CAA_Acf_Name" , all.x = TRUE)
944
945 # 3.3 Merging CAA ATM movements data Registration with harmonised IBA aircraft inventory
946 # The previous version had an unused bit of code which has been cleaned out after QA by MH - Apr 2019.
947 # Upon discussion DT decided this unused bit of code wasnt necessary and better functionality was provided by a code in
948 Section 1.7
949
950 ATM_temp <- merge(ATM_temp, Aircraft_inventory, by.x = "aircraft_reg", by.y = "Registration", all.x = TRUE)
951
952 # 3.3.1 Creating a merge report for to see successful and unsuccessful matches -
953 # unsuccessful matches will we appropriately handled i.e. removed for fmm purposes
954
955 # Match success reports by registration
956 IBA_match_status <- ATM_temp %>%
957   group_by(aircraft_reg) %>%
958   summarise(no_obs = n()) %>%
959   mutate(harmonised_match_success = aircraft_reg %in% Aircraft_inventory$Registration,
960          jan_match_success = aircraft_reg %in% Aircraft_inventory_Jan$Registration,
961          dec_match_success = aircraft_reg %in% Aircraft_inventory_Dec$Registration,
962          missing_match_success = aircraft_reg %in% Aircraft_inventory_Missing$Registration ) %>%
963   arrange(desc(no_obs))
964
965 # Match success reports by TRUE and FALSE
966 IBA_match_success <- IBA_match_status %>%
967   gather(key, value, harmonised_match_success:missing_match_success ) %>%
968   group_by(value, key) %>%
969   summarise(observations = sum(no_obs)) %>%
970   spread(key, observations)
971
972 IBA_match_status <- IBA_match_status %>%
973   adorn_totals(where = "row")
974
975 # Aircraft reg not matched
976 IBA_negative_match <- ATM_temp %>%
977   filter(ATM_temp$aircraft_reg %in% IBA_match_status$aircraft_reg[IBA_match_status$harmonised_match_success == FALSE]) %>%
978   group_by(aircraft_reg,aoc_holder_name, aircraft_name) %>%
979   summarise(no_obs = n(), terminal_pax = sum(pax_terminal_total), cargo_weight_kg = sum(cargo_weight_kg)) %>%
980   arrange(desc(no_obs)) %>%
981   adorn_totals(where = "row")
982
983 # 3.3.2 Records where merge has been unsuccessful, change N/A in Seating Capacity to -1 for cutting into seat class zero
984 # These records will use CAA ATM movement's data available seats for FMM seat classing
985 # Records where both the above approaches cannot be performed are removed
986
987 ATM_temp$`Seating Capacity`[is.na(ATM_temp$`Seating Capacity`)] <- -1
988
989 #3.4 Allocating flight movements to DfT carrier type as Defined by the Department Aviation Model
990 # Schedule(Sch), Charter(Ch), Low Cost Carrier(LCC), Freight(F) and Other(O) - Simplified as S,C,L,F,O for notation
991 # This step applies SCLFO classification based on the following rules to the relevant ATM file
992 # Note - (CAA) service_type has three possible values of "Cargo Only", "Passenger" & "Transit Cargo" - CAA terminology NOT
993 DfT Aviation Model
994 # Note - (CAA) operation_type has three possible values of "Charter", "Government Charter" & "Scheduled" - CAA terminology
995 NOT DfT Aviation Model
996 # Classification begins by creation of a new columnn called SCLFO

```

```
996 ATM_temp$SCLFO = as.character("",nrow(ATM_temp))
997
998 # Rules commence as below and subsequent rules override the previous one - where applicable
999
1000 # Rule 1 - If service_type is "Passenger", the movement is classed as S (Schedule)
1001
1002 ATM_temp$SCLFO[ATM_temp$service_type %in% c("Passenger")] = "S"
1003
1004 # Rule 2 - If the aoc_holder_name is one of - "RYANAIR", "EASYJET AIRLINE COMPANY LTD", "EASYJET SWITZERLAND", "JET2.COM LTD"
or "TUI AIRWAYS LTD" - Then movement is reclassified as L (Low Cost)
1005
1006 ATM_temp$SCLFO[ATM_temp$aoc_holder_name %in% c("RYANAIR", "EASYJET AIRLINE COMPANY LTD", "EASYJET SWITZERLAND", "JET2.COM
LTD")] = "L"
1007
1008 # Rule 3 - if operation_type is "Charter" , then movement is reclassified as C (Charter)
1009
1010 ATM_temp$SCLFO[ATM_temp$operation_type %in% c("Charter")] = "C"
1011
1012 # Rule 4 - if service_type is either "Cargo Only" or "Transit Cargo" , then movement is reclassified as F (Freighter)
1013
1014 ATM_temp$SCLFO[ATM_temp$service_type %in% c("Cargo Only","Transit Cargo")] = "F"
1015
1016 # Rule 5 - if operation_type is "Government Charter" , then movement is reclassified as O (Other)
1017
1018 ATM_temp$SCLFO[ATM_temp$operation_type %in% c("Government Charter")] = "O"
1019
1020 # Rule 6 - if pax_terminal_total & pax_transit_total & cargo_weight_kg are all zero, then movement is reclassified as O (Other)
1021
1022 ATM_temp$SCLFO[ATM_temp$pax_terminal_total==0 & ATM_temp$pax_transit_total==0 & ATM_temp$cargo_weight_kg==0 ] = "O"
1023
1024 # Rule 7- if the lastnext airport is in the UK and Chartered, it is classed as O
1025
1026 ATM_temp$SCLFO[ATM_temp$lastnext_spasm_zone %in% UK_modelled_airports & ATM_temp$SCLFO %in% c("C")] = "O"
1027 ATM_temp$SCLFO[ATM_temp$lastnext_spasm_zone %in% c(599) & ATM_temp$SCLFO %in% c("C")] = "O"
1028
1029 # Rule 8 - if the flight movement is S or L and has either reporting or last nextairport SPASM zone as 599, it is classed as O
1030
1031 ATM_temp$SCLFO[ATM_temp$SCLFO %in% c("S","L") & ATM_temp$reporting_spasm_zone == 599] = "O"
1032 ATM_temp$SCLFO[ATM_temp$SCLFO %in% c("S","L") & ATM_temp$lastnext_spasm_zone == 599] = "O"
1033
1034 # Now convert SCLFO values to factors to speed up processing
1035
1036 ATM_temp$SCLFO <- as.factor(ATM_temp$SCLFO)
1037
1038 # Step 3.4.1 SPASM Match for reference
1039
1040 SPASM_Match_Airport_SCLFO <- ATM_temp %>%
1041   filter(reporting_spasm_zone != 599) %>%
1042   group_by(reporting_spasm_zone,reporting_airport,SCLFO) %>%
1043   summarise(no_obs = n()) %>%
1044   spread(SCLFO, no_obs, fill=0) %>%
1045   mutate(SCL=S+C+L,
1046          SCLFO=SCL+F+O,
1047          ` ` =') %>%
1048   select(reporting_spasm_zone, reporting_airport, S, L, C , ` ` , SCL, F, O, SCLFO) %>%
1049   arrange() %>%
1050   adorn_totals(where = "row")
1051
1052
1053 #3.5 Assigning seat classes to aircraft movement using Seating capacity(IBA) first and then available_seats(CAA) for the
missing ones
1054
1055 ATM_temp$FMM_seatclass <- cut(ATM_temp$`Seating Capacity`,
c(-Inf,1,70,150,250,350,500,Inf),labels=c("0","1","2","3","4","5","6"))
1056
1057 ATM_temp$FMM_seatclass[ATM_temp$FMM_seatclass==0] <- cut(ATM_temp$available_seats[ATM_temp$FMM_seatclass==0],
c(-Inf,1,70,150,250,350,500,Inf),
1058                                labels=c("0","1","2","3","4","5","6"))
1059
1060
1061 # 3.6 Cleaning out non-relevant information for FMM purposes
1062
1063
1064
1065 # TRUNCATE ATM records only for modelled airports and SCL
1066 ATM_temp <- ATM_temp[ATM_temp$reporting_spasm_zone %in% UK_modelled_airports , ]
1067 ATM_temp <- ATM_temp[ATM_temp$SCLFO %in% c("S","C","L"),]
1068
1069 # Clean out "Unknown" aircraft_reg , "Unspecified" aircraft_name, "Unknown" lastnext_airport and NA Aircraft ICAO Code
1070 ATM_temp <- ATM_temp[ATM_temp$aircraft_reg!= "UNKNOWN" ,]
1071 ATM_temp <- ATM_temp[ATM_temp$aircraft_name!= "UNSPECIFIED",]
1072 ATM_temp <- ATM_temp[ATM_temp$lastnext_airport!= "UNKNOWN",]
1073 ATM_temp <- ATM_temp[ATM_temp$`Aircraft ICAO Code` != "Unknown",]
1074
1075 ATM_temp <- ATM_temp %>% drop_na(c(aircraft_reg, aircraft_name, lastnext_airport, reporting_airport, `Aircraft ICAO Code`))
1076
1077
1078 # 3.7 Cleaning based on items merged from Aircraft Inventory
1079 # cleaning out items that have zero seat class
1080 ATM_temp <- ATM_temp[ATM_temp$FMM_seatclass!= 0,]
1081
1082 # cleaning out items that have NA Age_midYear field
1083 ATM_temp <- ATM_temp %>% drop_na(Age_midYear)
1084
1085 # For proportionality any IATA code that occurs less than 100 times in the year is removed
1086
1087 ATM_temp <- ATM_temp[ATM_temp$IATA %in% names(which(table(ATM_temp$IATA) > 100)),]
1088
1089 # 3.8 Create a new column representing FMM service type - Sch, Ch and NFC (FMM terminology for low cost)
1090 ATM_temp$fmm_service_type = as.character("",nrow(ATM_temp))
1091
1092 ATM_temp$fmm_service_type[ATM_temp$SCLFO %in% c("S")] = "Sch"
1093 ATM_temp$fmm_service_type[ATM_temp$SCLFO %in% c("C")] = "Ch"
1094 ATM_temp$fmm_service_type[ATM_temp$SCLFO %in% c("L")] = "NFC"
1095
```



```
1096 # Create a new column representing FMM segment i.e. model worksheets in the fmm
1097
1098
1099 ATM_temp$fmm_segment <- paste0("c",ATM_temp$FMM_seatclass,ATM_temp$fmm_service_type)
1100
1101
1102
1103 # Threshold to at least 200 atms per year for that IATA code within the segment to remove noisy data
1104 ATM_temp$combinedrule <- with(ATM_temp, interaction(IATA,fmm_segment))
1105 ATM_temp <- ATM_temp[!table(ATM_temp$combinedrule)[ATM_temp$combinedrule] < 200,]
1106
1107 #cleaning levels that have no observations
1108 ATM_temp <- droplevels(ATM_temp)
1109
1110 # Removing intermediate versions of flight inventories
1111
1112 rm(Aircraft_inventory_Dec,Aircraft_inventory_Jan,Aircraft_inventory_Missing)
1113
1114
1115 # Step 4 - Post processing Quality Assurance prior to feeding into FMM ----
1116 ## Relevant data in previous step are consolidated into dataframes and findings are saved in single .xlsx file for review
1117
1118 #4.1.1 QA411 - High level integrity check - Get number of records for resultant ATMs file
1119
1120 post_proc_ATM_temp_no_of_records <- nrow(ATM_temp)
1121 post_proc_ATM_temp_no_of_cols <- ncol(ATM_temp)
1122 post_proc_row_col_total <- rbind(post_proc_ATM_temp_no_of_cols,post_proc_ATM_temp_no_of_records)
1123 colnames(post_proc_row_col_total) <- "no_obs"
1124
1125 #4.1.2 QA412 - Dataframe with disaggregation of resultant ATMs by SCLFO
1126 post_proc_QA_SCLFO <- ATM_temp %>%
1127   group_by(SCLFO,direction_type)%>%
1128   summarise(no_obs = n()) %>%
1129   arrange(desc(no_obs)) %>%
1130   adorn_totals(where = "row")
1131
1132 #4.1.3 QA413 - Dataframe with aircraft registrations comparing IBA's seating capacity and CAA's available seats arranged in
1133 descending order
1134
1135 ATM_temp$IBA_seatclass <- cut(ATM_temp$`Seating Capacity`,
1136   c(-Inf,1,70,150,250,350,500,Inf),labels=c("0","1","2","3","4","5","6"))
1137 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"))
1138
1139 post_proc_QA_AC_IATA_seats <- ATM_temp %>%
1140   group_by(aircraft_reg,IATA,IBA_seatclass,CAA_seatclass, FMM_seatclass) %>%
1141   summarise(no_obs = n())
1142
1143 post_proc_QA_AC_IATA_seats$difference <- as.numeric(post_proc_QA_AC_IATA_seats$IBA_seatclass) -
1144 as.numeric(post_proc_QA_AC_IATA_seats$CAA_seatclass)
1145 post_proc_QA_AC_IATA_seats$difference <- abs(post_proc_QA_AC_IATA_seats$difference)
1146 post_proc_QA_AC_IATA_seats$missing_IBA <- ifelse(post_proc_QA_AC_IATA_seats$IBA_seatclass==0, "Missing IBA", "IBA present")
1147 post_proc_QA_AC_IATA_seats$missing_CAA <- ifelse(post_proc_QA_AC_IATA_seats$CAA_seatclass==0, "Missing CAA", "CAA present")
1148
1149 ATM_temp$IBA_seatclass <- NULL
1150 ATM_temp$CAA_seatclass <- NULL
1151
1152 #4.1.4 QA414 - Dataframe with disaggregation of resultant ATMs by FMM seat class
1153 post_proc_QA_SeatClass <- ATM_temp %>%
1154   group_by(FMM_seatclass, SCLFO) %>%
1155   summarise(no_obs = n()) %>%
1156   spread(SCLFO,no_obs, fill = 0) %>%
1157   select(FMM_seatclass,S, L, C) %>%
1158   adorn_totals(where = "row")
1159
1160 #4.1.5 QA415 - Dataframe with contingency table for airports vs SCLFO disaggregation
1161 post_proc_QA_Airport_SCLFO <- ATM_temp %>%
1162   group_by(reporting_spasm_zone,reporting_airport,SCLFO) %>%
1163   summarise(no_obs = n()) %>%
1164   spread(SCLFO,no_obs, fill = 0) %>%
1165   select(reporting_spasm_zone,reporting_airport, S, L, C) %>%
1166   adorn_totals(where = c("col","row"))
1167
1168 #4.1.6 QA416 - Dataframe with contingency table by aircraft code, type and total
1169 post_proc_QA_AC_IATA_SCLFO <- ATM_temp %>%
1170   group_by(IATA, aircraft_name, `Aircraft ICAO Code`,In_Production,Fmm_Haul, FMM_seatclass,SCLFO) %>%
1171   summarise(no_obs = n()) %>%
1172   spread(SCLFO,no_obs, fill = 0) %>%
1173   select(IATA, aircraft_name, `Aircraft ICAO Code`,In_Production,Fmm_Haul, FMM_seatclass, S, L, C) %>%
1174   adorn_totals(where = "col")
1175
1176 #4.1.7 QA417 - Dataframe with contingency table by in production IATA code (only) , seat class, totals and percents
1177 post_proc_QA_AC_IATA_SCLFO_InProd <- ATM_temp %>% filter(In_Production == "1") %>%
1178   group_by(FMM_seatclass, IATA, SCLFO) %>%
1179   summarise(no_obs = n()) %>%
1180   spread(SCLFO,no_obs, fill = 0) %>%
1181   select(IATA,FMM_seatclass, S, L, C) %>%
1182   arrange(FMM_seatclass) %>%
1183   ungroup() %>%
1184   adorn_totals(where = "col") %>%
1185   group_by(FMM_seatclass) %>%
1186   mutate("S pc within seatclass" = round(S/sum(S),2),
1187     "L pc within seatclass" = round(L/sum(L),2) ,
1188     "C pc within seatclass" = round(C/sum(C),2)
1189   )
1190
1191 # 4.1.8 QA418 - Global aircraft orders without UK Low Cost Carriers - absolute values by years using IBA Aircraft order
1192 post_proc_Global_family_order_count_excUKLCC <- Aircraft_order %>%
1193   filter(`Build Year`>2016 & !Operator %in% c("easyJet","Ryanair","Jet2.com")) %>%
1194   group_by(`Aircraft Family`,`Build Year`) %>%
1195   summarise(no_obs = n()) %>%
1196   spread(`Build Year`,no_obs,fill=0) %>%
1197   adorn_totals(where = c("row", "col"))
1198
1199 # 4.1.9 QA419- Global aircraft orders percentages without UK Low Cost Carriers by years using IBA Aircraft order
1200 post_proc_Global_family_order_percentage_excUKLCC <- Aircraft_order %>%
```

```
1198     filter(`Build Year`>2016 & !Operator %in% c("easyJet","Ryanair","Jet2.com")) %>%
1199     group_by(`Aircraft Family`, `Build Year`) %>%
1200     summarise(no_obs = n()) %>%
1201     spread(`Build Year`,no_obs,fill=0) %>%
1202     adorn_percentages(denominator = "col") %>%
1203     mutate_if(is.numeric, round, 2)
1204
1205
1206 # 4.1.10 QA4110- Aircraft orders for Ryanair, EasyJet and Jet2 - absolute values by years using IBA Aircraft order
1207 post_proc_UK_low_cost_order_count <- Aircraft_order %>%
1208     filter(`Build Year`>2016 & Operator %in% c("easyJet","Ryanair","Jet2.com")) %>%
1209     group_by(`Aircraft Family`, `Aircraft Model`, `Build Year`) %>%
1210     summarise(no_obs = n()) %>%
1211     spread(`Build Year`,no_obs,fill=0) %>%
1212     adorn_totals(where = c("row", "col"))
1213
1214
1215 # 4.1.11 QA4111 - Aircraft orders for Ryanair, EasyJet and Jet2 percentages by years using IBA Aircraft order
1216 post_proc_UK_low_cost_order_percentage <- Aircraft_order %>%
1217     filter(`Build Year`>2016 & Operator %in% c("easyJet","Ryanair","Jet2.com")) %>%
1218     group_by(`Aircraft Family`, `Aircraft Model`, `Build Year`) %>%
1219     summarise(no_obs = n()) %>%
1220     spread(`Build Year`,no_obs,fill=0) %>%
1221     adorn_percentages(denominator = "col") %>%
1222     mutate_if(is.numeric, round, 2)
1223
1224
1225 # 4.2 Write POST PROCESSING QA excel file with dataframes above
1226
1227 # 4.2.1 declare a QA workbook object
1228 wb_qa_post <- createWorkbook(creator = "dtathgur", title = paste("Quality Assurance Post",year))
1229
1230 # Add worksheets for each dataframe
1231 addWorksheet(wb_qa_post, paste0("PostProcessedData>>>"),
1232             tabColour = "#00ffbb" ) #410
1233 addWorksheet(wb_qa_post, paste0("Topsheet_",year),
1234             tabColour = "#00ffbb" ) #411
1235 addWorksheet(wb_qa_post, paste0("SCLFO_",year),
1236             tabColour = "#00ffbb") #412
1237 addWorksheet(wb_qa_post, paste0("SeatClass_Compare_",year) ,
1238             tabColour = "#00ffbb") #413
1239 addWorksheet(wb_qa_post, paste0("SCLFO_SeatClass_",year) ,
1240             tabColour = "#00ffbb") #414
1241 addWorksheet(wb_qa_post, paste0("SCLFO_Airport_",year),
1242             tabColour = "#00ffbb" ) #415
1243 addWorksheet(wb_qa_post, paste0("SCLFO_IATA_",year) ,
1244             tabColour = "#00ffbb") #416
1245 addWorksheet(wb_qa_post, paste0("SCLFO_IATA_InProd_",year) ,
1246             tabColour = "#00ffbb") #417
1247 addWorksheet(wb_qa_post, paste0("Order_count_Exc_UKLCC_",year) ,
1248             tabColour = "#00ffbb") #418
1249 addWorksheet(wb_qa_post, paste0("Order_PC_Exc_UKLCC_",year) ,
1250             tabColour = "#00ffbb") #419
1251 addWorksheet(wb_qa_post, paste0("UK_LCC_Order_count_",year) ,
1252             tabColour = "#00ffbb") #420
1253 addWorksheet(wb_qa_post, paste0("UK_LCC_Order_PC_",year) ,
1254             tabColour = "#00ffbb") #421
1255
1256 addWorksheet(wb_qa_post, paste0("DIAGNOSTICS>>>"),
1257             tabColour = "#2d4423" ) #Match topsheet
1258
1259 addWorksheet(wb_qa_post, paste0("IBA_match_status_",year),
1260             tabColour = "#2d4423" ) #422 data frame called IBA_match_status
1261 addWorksheet(wb_qa_post, paste0("IBA_negative_match_",year),
1262             tabColour = "#2d4423" ) #423 data frame called IBA_negative_match
1263 addWorksheet(wb_qa_post, paste0("SPASM_match_",year) ,
1264             tabColour = "#2d4423" ) #424 data frame called SPASM Match
1265
1266
1267 # Add description to each worksheet
1268
1269 writeData(wb_qa_post, paste0("Topsheet_",year),c(paste("Post Processed "," - ",date())," No. of records and variables"), #411
1270          startCol = 1, startRow = 1, xy = NULL,
1271          borders = "rows",
1272          borderColour = getOption("openxlsx.borderColour", "black"),
1273          borderStyle = getOption("openxlsx.borderStyle", "thick")
1274          )
1275
1276
1277 writeData(wb_qa_post, paste0("SCLFO_",year),c(paste("Post Processed "," - ",date()),"SCLFO and direction_type"), #412
1278          startCol = 1, startRow = 1, xy = NULL,
1279          borders = "rows",
1280          borderColour = getOption("openxlsx.borderColour", "black"),
1281          borderStyle = getOption("openxlsx.borderStyle", "thick")
1282          )
1283 writeData(wb_qa_post, paste0("SeatClass_Compare_",year),c(paste("Post Processed "," - ",date())," Aircraft Reg and seat class
1284 approaches for all records"), #413
1285          startCol = 1, startRow = 1, xy = NULL,
1286          borders = "rows",
1287          borderColour = getOption("openxlsx.borderColour", "black"),
1288          borderStyle = getOption("openxlsx.borderStyle", "thick")
1289          )
1290
1291 writeData(wb_qa_post, paste0("SCLFO_SeatClass_",year),c(paste("Post Processed "," - ",date())," FMM seat class by SCLFO"), #414
1292          startCol = 1, startRow = 1, xy = NULL,
1293          borders = "rows",
1294          borderColour = getOption("openxlsx.borderColour", "black"),
1295          borderStyle = getOption("openxlsx.borderStyle", "thick")
1296          )
1297 writeData(wb_qa_post, paste0("SCLFO_Airport_",year),c(paste("Post Processed "," - ",date())," Reporting Airports by SCLFO"),
1298          #415
1299          startCol = 1, startRow = 1, xy = NULL,
1300          borders = "rows",
1301          borderColour = getOption("openxlsx.borderColour", "black"),
```

```
1301         borderStyle = getOption("openxlsx.borderStyle", "thick")
1302     )
1303
1304
1305 writeData(wb_qa_post, paste0("SCLFO_IATA_",year),c(paste("Post Processed "," - ",date())," Aircraft details by IATA and
SCLFO"), #416
1306         startCol = 1, startRow = 1, xy = NULL,
1307         borders = "rows",
1308         borderColour = getOption("openxlsx.borderColour", "black"),
1309         borderStyle = getOption("openxlsx.borderStyle", "thick")
1310     )
1311
1312 writeData(wb_qa_post, paste0("SCLFO_IATA_InProd_",year),c(paste("Post Processed "," - ",date())," In Production Aircraft
details by IATA codes and SCLFO"), #417
1313         startCol = 1, startRow = 1, xy = NULL,
1314         borders = "rows",
1315         borderColour = getOption("openxlsx.borderColour", "black"),
1316         borderStyle = getOption("openxlsx.borderStyle", "thick")
1317     )
1318
1319 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
1320         startCol = 1, startRow = 1, xy = NULL,
1321         borders = "rows",
1322         borderColour = getOption("openxlsx.borderColour", "black"),
1323         borderStyle = getOption("openxlsx.borderStyle", "thick")
1324     )
1325
1326 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
1327         startCol = 1, startRow = 1, xy = NULL,
1328         borders = "rows",
1329         borderColour = getOption("openxlsx.borderColour", "black"),
1330         borderStyle = getOption("openxlsx.borderStyle", "thick")
1331     )
1332
1333
1334
1335 writeData(wb_qa_post, paste0("UK_LCC_Order_count_",year),c(paste("Post Processed "," - ",date())," UK Low Cost Carriers order
(from IBA) values"), #420
1336         startCol = 1, startRow = 1, xy = NULL,
1337         borders = "rows",
1338         borderColour = getOption("openxlsx.borderColour", "black"),
1339         borderStyle = getOption("openxlsx.borderStyle", "thick")
1340     )
1341
1342 writeData(wb_qa_post, paste0("UK_LCC_Order_PC_",year),c(paste("Post Processed "," - ",date())," UK Low Cost Carriers order
(from IBA) percentages "), #421
1343         startCol = 1, startRow = 1, xy = NULL,
1344         borders = "rows",
1345         borderColour = getOption("openxlsx.borderColour", "black"),
1346         borderStyle = getOption("openxlsx.borderStyle", "thick")
1347     )
1348
1349
1350
1351
1352 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
1353         startCol = 1, startRow = 1, xy = NULL,
1354         borders = "rows",
1355         borderColour = getOption("openxlsx.borderColour", "black"),
1356         borderStyle = getOption("openxlsx.borderStyle", "thick")
1357     )
1358
1359
1360 writeData(wb_qa_post, paste0("IBA_negative_match_",year),c(paste("Post Processed "," - ",date())," Aircraft reg not
matched"), #423
1361         startCol = 1, startRow = 1, xy = NULL,
1362         borders = "rows",
1363         borderColour = getOption("openxlsx.borderColour", "black"),
1364         borderStyle = getOption("openxlsx.borderStyle", "thick")
1365     )
1366 writeData(wb_qa_post, paste0("SPASM_match_",year),c(paste("Post Processed "," - ",date())," SPASM Match file"), #424
1367         startCol = 1, startRow = 1, xy = NULL,
1368         borders = "rows",
1369         borderColour = getOption("openxlsx.borderColour", "black"),
1370         borderStyle = getOption("openxlsx.borderStyle", "thick")
1371     )
1372
1373
1374
1375 # Add data to each worksheet
1376
1377 writeDataTable(wb_qa_post, paste0("Topsheet_",year), data.frame(post_proc_row_col_total), startCol = 1, startRow = 3, xy =
NULL,
1378         colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium3",
1379         tableName = NULL, headerStyle = NULL, withFilter = TRUE,
1380         keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
1381         lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
1382
1383 writeDataTable(wb_qa_post, paste0("SCLFO_",year), post_proc_QA_SCLFO, startCol = 1, startRow = 3, xy = NULL,
1384         colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium2",
1385         tableName = NULL, headerStyle = NULL, withFilter = TRUE,
1386         keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
1387         lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
1388
1389 writeDataTable(wb_qa_post, paste0("SeatClass_Compare_",year), post_proc_QA_AC_IATA_seats, startCol = 1, startRow = 3, xy =
NULL,
1390         colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium5",
1391         tableName = NULL, headerStyle = NULL, withFilter = TRUE,
1392         keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
1393         lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
1394
1395 writeDataTable(wb_qa_post, paste0("SCLFO_SeatClass_",year), post_proc_QA_SeatClass, startCol = 1, startRow = 3, xy = NULL,
```



```
1396         colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium3",
1397         tableName = NULL, headerStyle = NULL, withFilter = TRUE,
1398         keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
1399         lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
1400
1401 writeDataTable(wb_qa_post, paste0("SCLFO_Airport_",year), post_proc_QA_Airport_SCLFO, startCol = 1, startRow = 3, xy = NULL,
1402         colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium3",
1403         tableName = NULL, headerStyle = NULL, withFilter = TRUE,
1404         keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
1405         lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
1406
1407 writeDataTable(wb_qa_post, paste0("SCLFO_IATA_",year), post_proc_QA_AC_IATA_SCLFO, startCol = 1, startRow = 3, xy = NULL,
1408         colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium4",
1409         tableName = NULL, headerStyle = NULL, withFilter = TRUE,
1410         keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
1411         lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
1412
1413 writeDataTable(wb_qa_post, paste0("SCLFO_IATA_InProd_",year), post_proc_QA_AC_IATA_SCLFO_InProd, startCol = 1, startRow = 3,
1414 xy = NULL,
1415         colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium4",
1416         tableName = NULL, headerStyle = NULL, withFilter = TRUE,
1417         keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
1418         lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
1419
1420 writeDataTable(wb_qa_post, paste0("Order_count_Exc_UKLCC_",year), post_proc_Global_family_order_count_excUKLCC , startCol =
1421 1, startRow = 3, xy = NULL,
1422         colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium4",
1423         tableName = NULL, headerStyle = NULL, withFilter = TRUE,
1424         keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
1425         lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
1426
1427 writeDataTable(wb_qa_post, paste0("Order_PC_Exc_UKLCC_",year), post_proc_Global_family_order_percentage_excUKLCC , startCol =
1428 1, startRow = 3, xy = NULL,
1429         colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium4",
1430         tableName = NULL, headerStyle = NULL, withFilter = TRUE,
1431         keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
1432         lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
1433
1434 writeDataTable(wb_qa_post, paste0("UK_LCC_Order_count_",year), post_proc_UK_low_cost_order_count, startCol = 1, startRow = 3,
1435 xy = NULL,
1436         colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium4",
1437         tableName = NULL, headerStyle = NULL, withFilter = TRUE,
1438         keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
1439         lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
1440
1441 writeDataTable(wb_qa_post, paste0("UK_LCC_Order_PC_",year), post_proc_UK_low_cost_order_percentage, startCol = 1, startRow =
1442 3, xy = NULL,
1443         colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium4",
1444         tableName = NULL, headerStyle = NULL, withFilter = TRUE,
1445         keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
1446         lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
1447
1448 writeDataTable(wb_qa_post, paste0("IBA_match_status_",year),IBA_match_status, startCol = 1, startRow = 3, xy = NULL,
1449         colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium3",
1450         tableName = NULL, headerStyle = NULL, withFilter = TRUE,
1451         keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
1452         lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
1453
1454 writeDataTable(wb_qa_post, paste0("IBA_match_status_",year),IBA_match_success, startCol = 9, startRow = 5, xy = NULL,
1455         colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium2",
1456         tableName = NULL, headerStyle = NULL, withFilter = TRUE,
1457         keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
1458         lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
1459
1460 writeDataTable(wb_qa_post, paste0("IBA_negative_match_",year), IBA_negative_match, startCol = 1, startRow = 3, xy = NULL,
1461         colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium6",
1462         tableName = NULL, headerStyle = NULL, withFilter = TRUE,
1463         keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
1464         lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
1465
1466 writeDataTable(wb_qa_post, paste0("SPASM_match_",year), SPASM_Match_Airport_SCLFO, startCol = 1, startRow = 3, xy = NULL,
1467         colNames = TRUE, rowNames = TRUE, tableStyle = "TableStyleMedium4",
1468         tableName = NULL, headerStyle = NULL, withFilter = TRUE,
1469         keepNA = TRUE, sep = ", ", stack = FALSE, firstColumn = FALSE,
1470         lastColumn = TRUE, bandedRows = TRUE, bandedCols = TRUE)
1471
1472 # Save into a workbook and increase size of columns for easy read
1473
1474 saveWorkbook(wb_qa_post, file = paste0("Post_Processing_Review_",year,".xlsx") , overwrite = TRUE)
1475 for(sheetindex in c(1:length(getSheetNames(paste0("Post_Processing_Review_",year,".xlsx")))) )
1476 {setColWidths(wb_qa_post, sheet = sheetindex, cols = 1:7, widths = 20)
1477 }
1478 saveWorkbook(wb_qa_post, file = paste0("Post_Processing_Review_",year,".xlsx") , overwrite = TRUE)
1479
1480 # remove post processing QA files
1481 rm(IBA_match_status,IBA_match_success,IBA_negative_match, post_proc_ATM_temp_no_of_cols,post_proc_ATM_temp_no_of_records,
1482 post_proc_Global_family_order_count_excUKLCC,
1483 post_proc_Global_family_order_percentage_excUKLCC, post_proc_QA_AC_IATA_SCLFO, post_proc_QA_AC_IATA_SCLFO_InProd,
1484 post_proc_QA_AC_IATA_seats, post_proc_QA_Airport_SCLFO,
1485 post_proc_QA_SCLFO, post_proc_QA_SeatClass, post_proc_row_col_total, post_proc_UK_low_cost_order_count,
1486 post_proc_UK_low_cost_order_percentage,SPASM_Match_Airport_SCLFO)
1487
1488 # Step 5 - Prepare data for FMM and output into CSV -----
1489
1490 # 5.1 Create a contingency table cross tabulating unique aircraft types with fmm segments
1491
1492 # Ensure they are ordered as per fmm
1493 ATM_temp$fmm_segment<- factor(ATM_temp$fmm_segment, levels=c("c1Sch","c2Sch", "c3Sch", "c4Sch" ,"c5Sch" ,"c6Sch",
1494 "c1Ch","c2Ch","c3Ch","c4Ch","c5Ch","c6Ch",
1495 "c1NFC","c2NFC","c3NFC","c4NFC","c5NFC","c6NFC"))
1496
1497 fmm_table <- table(ATM_temp$IATA,ATM_temp$fmm_segment)
1498 fmm_table <- fmm_table[rowSums(fmm_table)>=1,]
1499 fmm_table <- as.data.frame.matrix(fmm_table)
1500
```

```
1493
1494 # 5.4.1 Create ATM age distributions for all service type andthen for Sch, Ch and NFC individually
1495
1496 ATM_temp$Age_midYear_fct <- as.factor(ATM_temp$Age_midYear)
1497
1498 # All service type ATM age distribution
1499
1500 ATM_age_dist_All <- ATM_temp %>% group_by(IATA, Age_midYear_fct) %>%
1501   summarise(n=n()) %>% spread(Age_midYear_fct,n, drop = FALSE, fill = 0)
1502
1503 # Sch population ATM age distribution
1504 ATM_age_dist_Sch <- ATM_temp %>% filter(fmm_service_type == "Sch") %>% group_by(IATA, Age_midYear_fct) %>%
1505   summarise(n=n()) %>% spread(Age_midYear_fct, n, drop = FALSE, fill = 0)
1506
1507 # Ch population ATM age distribution
1508 ATM_age_dist_Ch <- ATM_temp %>% filter(fmm_service_type == "Ch") %>% group_by(IATA, Age_midYear_fct) %>%
1509   summarise(n=n()) %>% spread(Age_midYear_fct, n, drop = FALSE, fill = 0)
1510
1511 # NFC population ATM age distribution
1512 ATM_age_dist_NFC <- ATM_temp %>% filter(fmm_service_type == "NFC") %>% group_by(IATA, Age_midYear_fct) %>%
1513   summarise(n=n()) %>% spread(Age_midYear_fct, n, drop = FALSE, fill = 0)
1514
1515 # 5.4.2 Create Aircraft age distributions by registration code for all service type and then for Sch, Ch and NFC individually
1516
1517 # All service type aircraft age distribution by type and registration
1518
1519 Reg_age_dist_All <- ATM_temp %>%
1520   group_by(IATA,Age_midYear_fct) %>%
1521   summarise(n=n_distinct(aircraft_reg)) %>%
1522   spread(Age_midYear_fct,n, drop = FALSE, fill = 0)
1523
1524 # Sch service type aircraft age distribution by type and registration
1525 Reg_age_dist_Sch <- ATM_temp %>% filter(fmm_service_type == "Sch") %>%
1526   group_by(IATA,Age_midYear_fct) %>%
1527   summarise(n=n_distinct(aircraft_reg)) %>%
1528   spread(Age_midYear_fct,n, drop = FALSE, fill = 0)
1529
1530
1531 # Ch service type aircraft age distribution by type and registration
1532 Reg_age_dist_Ch <- ATM temp %>% filter(fmm_service_type == "Ch") %>%
1533   group_by(IATA,Age_midYear_fct) %>%
1534   summarise(n=n_distinct(aircraft_reg)) %>%
1535   spread(Age_midYear_fct,n, drop = FALSE, fill = 0)
1536
1537
1538 # Sch service type aircraft age distribution by type and registration
1539 Reg_age_dist_NFC <- ATM temp %>% filter(fmm_service_type == "NFC") %>%
1540   group_by(IATA,Age_midYear_fct) %>%
1541   summarise(n=n_distinct(aircraft_reg)) %>%
1542   spread(Age_midYear_fct,n, drop = FALSE, fill = 0)
1543
1544
1545
1546
1547 # 5.6 create workbooks for fmm parameters
1548
1549 wb_fmm_table <- createWorkbook(creator = "dtathgur", title = paste0("FMM_Inputs_",year))
1550 # add fmm contingency table
1551 addWorksheet(wb_fmm_table, paste0("FMM_",year) )
1552
1553 #add ATM age distributions
1554 addWorksheet(wb_fmm_table, paste0("ATM_age_dist_All_",year) )
1555 addWorksheet(wb_fmm_table, paste0("ATM_age_dist_Ch_",year) )
1556 addWorksheet(wb_fmm_table, paste0("ATM_age_dist_NFC_",year) )
1557 addWorksheet(wb_fmm_table, paste0("ATM_age_dist_Sch_",year) )
1558
1559 #add Reg age distributions
1560 addWorksheet(wb_fmm_table, paste0("Reg_age_dist_All_",year) )
1561 addWorksheet(wb_fmm_table, paste0("Reg_age_dist_Ch_",year) )
1562 addWorksheet(wb_fmm_table, paste0("Reg_age_dist_NFC_",year) )
1563 addWorksheet(wb_fmm_table, paste0("Reg_age_dist_Sch_",year) )
1564
1565 writeData(wb_fmm_table, paste0("FMM_",year),fmm_table , startCol = 1, startRow = 3, xy = NULL,
1566   colNames = TRUE, rowNames = TRUE, headerStyle = NULL,
1567   keepNA = TRUE, sep = ", ")
1568
1569 writeData(wb_fmm_table, paste0("ATM_age_dist_All_",year), ATM_age_dist_All , startCol = 1, startRow = 3, xy = NULL,
1570   colNames = TRUE, rowNames = TRUE, headerStyle = NULL,
1571   keepNA = TRUE, sep = ", ")
1572
1573 writeData(wb_fmm_table, paste0("ATM_age_dist_Ch_",year), ATM_age_dist_Ch , startCol = 1, startRow = 3, xy = NULL,
1574   colNames = TRUE, rowNames = TRUE, headerStyle = NULL,
1575   keepNA = TRUE, sep = ", ")
1576
1577 writeData(wb_fmm_table, paste0("ATM_age_dist_NFC_",year), ATM_age_dist_NFC , startCol = 1, startRow = 3, xy = NULL,
1578   colNames = TRUE, rowNames = TRUE, headerStyle = NULL,
1579   keepNA = TRUE, sep = ", ")
1580
1581 writeData(wb_fmm_table, paste0("ATM_age_dist_Sch_",year), ATM_age_dist_Sch , startCol = 1, startRow = 3, xy = NULL,
1582   colNames = TRUE, rowNames = TRUE, headerStyle = NULL,
1583   keepNA = TRUE, sep = ", ")
1584
1585 writeData(wb_fmm_table, paste0("Reg_age_dist_All_",year), Reg_age_dist_All , startCol = 1, startRow = 3, xy = NULL,
1586   colNames = TRUE, rowNames = TRUE, headerStyle = NULL,
1587   keepNA = TRUE, sep = ", ")
1588
1589 writeData(wb_fmm_table, paste0("Reg_age_dist_Ch_",year), Reg_age_dist_Ch , startCol = 1, startRow = 3, xy = NULL,
1590   colNames = TRUE, rowNames = TRUE, headerStyle = NULL,
1591   keepNA = TRUE, sep = ", ")
1592
1593 writeData(wb_fmm_table, paste0("Reg_age_dist_NFC_",year), Reg_age_dist_NFC , startCol = 1, startRow = 3, xy = NULL,
1594   colNames = TRUE, rowNames = TRUE, headerStyle = NULL,
1595   keepNA = TRUE, sep = ", ")
1596
1597 writeData(wb_fmm_table, paste0("Reg_age_dist_Sch_",year), Reg_age_dist_Sch , startCol = 1, startRow = 3, xy = NULL,
```

```
1598         colNames = TRUE, rowNames = TRUE, headerStyle = NULL,
1599         keepNA = TRUE, sep = ", ")
1600
1601     # save workbooks
1602
1603     saveWorkbook(wb_fmm_table, file = paste0("FMM_Inputs_",year,".xlsx") , overwrite = TRUE)
1604
1605     # remove temporary files
1606     rm(Reg_age_dist_All,Reg_age_dist_Ch,Reg_age_dist_NFC,Reg_age_dist_Sch, ATM_retirement_All, ATM_retirement_Ch,
1607        ATM_retirement_NFC, ATM_retirement_Sch,
1608        fmm_table, OECD_countries, UK_modelled_airports)
1609
1610
```