```sas
/* Problem Statement 1 SAS Code */

/* Import data */
FILENAME REFFILE '/folders/myfolders/project/vehicles19.xlsx';

PROC IMPORT DATAFILE=REFFILE
    DBMS=XLSX
    OUT=WORK.VEHICLE;
    GETNAMES=YES;
RUN;

PROC CONTENTS DATA=WORK.VEHICLE; RUN;

/* Drop unused variables */
data work.vehicle; set work.vehicle;
    keep collision_id crash_date crash_time vehicle_type;
run;

/* Get crash weekday, crash month, crash hour */
data work.vehicle; set work.vehicle;
    crash_weekday = weekday(crash_date);
    crash_month = month(crash_date);
    crash_hour = hour(crash_time);
    drop crash_date crash_time;
    rename crash_hour=crash_time;
run;

/* Process vehicle type */
/* Find crash frequency of vehicle types */
PROC SQL;
CREATE TABLE WORK.FREQUENCY
AS
SELECT COUNT(VEHICLE.COLLISION_ID)
AS Frequency, VEHICLE.VEHICLE_TYPE
FROM WORK.VEHICLE VEHICLE
GROUP BY VEHICLE.VEHICLE_TYPE;
QUIT;

/* Subset vehicle types with frequency higher than 100 */
/* Insert frequency into main dataset */
data work.vehicle2; set work.vehicle;
run;

PROC SQL;
CREATE TABLE WORK.VEHICLE
AS
SELECT VEHICLE2.COLLISION_ID, VEHICLE2.VEHICLE_TYPE, VEHICLE2.crash_weekday, VEHICLE2.crash_month, VEHICLE2.c
FROM WORK.VEHICLE2 VEHICLE2
INNER JOIN WORK.FREQUENCY FREQUENCY
ON
   ( VEHICLE2.VEHICLE_TYPE = FREQUENCY.VEHICLE_TYPE ) ;
QUIT;

/* Subset */
data work.vehicle; set work.vehicle;
    where frequency >=100;
run;

data work.vehicle; set work.vehicle;
    where not missing (vehicle_type);
run;

/* Reclassify vehicle types */
data work.vehicle; set work.vehicle;
    length classify $38;
    If vehicle_type = "Box Truck" then classify = "Truck";
    else if vehicle_type = "Tractor Truck Diesel" then classify = "Truck";
    else if vehicle_type = "Flat Bed" then classify = "Truck";
    else if vehicle_type = "Dump" then classify = "Truck";
    else if vehicle_type = "Station Wagon/Sport Utility Vehicle" then classify = "SUV";
    else if vehicle_type = "Garbage or Refuse" then classify = "Truck";
    else if vehicle_type = "PK" then classify = "Pick-up Truck";
    else if vehicle_type = "4 dr sedan" then classify = "Sedan";
    else classify = vehicle_type;
    drop vehicle_type;
    rename classify = vehicle_type;
run;

/* Subset 10 vehicle types with highest crash frequency */
/* Check frequency */

```

```
81  PROC SQL;
82  CREATE TABLE WORK.FREQUENCY
83  AS
84  SELECT COUNT(VEHICLE.COLLISION_ID)
85  AS Frequency, VEHICLE.VEHICLE_TYPE
86  FROM WORK.VEHICLE VEHICLE
87  GROUP BY VEHICLE.VEHICLE_TYPE;
88  QUIT;
89
90  /* Insert frequency to main dataset */
91  data work.vehicle2; set work.vehicle;
92  run;
93
94  PROC SQL;
95  CREATE TABLE WORK.VEHICLE
96  AS
97  SELECT VEHICLE2.COLLISION_ID, VEHICLE2.VEHICLE_TYPE, VEHICLE2.crash_weekday, VEHICLE2.crash_month, VEHICLE2.c
98  FROM WORK.VEHICLE2 VEHICLE2
99  INNER JOIN WORK.FREQUENCY FREQUENCY
100 ON
101    ( VEHICLE2.VEHICLE_TYPE = FREQUENCY.VEHICLE_TYPE ) ;
102 QUIT;
103
104 /* Subset */
105 data work.vehicle; set work.vehicle;
106     if frequency > 1000;
107 run;
108
109 PROC SQL;
110 CREATE TABLE WORK.FREQUENCY
111 AS
112 SELECT COUNT(VEHICLE.COLLISION_ID)
113 AS Frequency, VEHICLE.VEHICLE_TYPE
114 FROM WORK.VEHICLE VEHICLE
115 GROUP BY VEHICLE.VEHICLE_TYPE;
116 QUIT;
117
118 /* Plot graph */
119 /* SUV & Sedan */
120 /* Time */
121 ods graphics / reset width=6.4in height=4.8in imagemap;
122
123 proc sgplot data=WORK.VEHICLE (where=(vehicle_type="Sedan" or
124         vehicle_type="SUV"));
125     vline crash_time / group=vehicle_type curvelabel;
126     yaxis grid;
127 run;
128
129 ods graphics / reset;
130
131 /* Weekday */
132 ods graphics / reset width=6.4in height=4.8in imagemap;
133
134 proc sgplot data=WORK.VEHICLE (where=(vehicle_type="Sedan" or
135         vehicle_type="SUV"));
136     vline crash_weekday / group=vehicle_type curvelabel;
137     yaxis grid;
138 run;
139
140 ods graphics / reset;
141
142 /* Month */
143 ods graphics / reset width=6.4in height=4.8in imagemap;
144
145 proc sgplot data=WORK.VEHICLE (where=(vehicle_type="Sedan" or
146         vehicle_type="SUV"));
147     vline crash_month / group=vehicle_type curvelabel;
148     yaxis grid;
149 run;
150
151 ods graphics / reset;
152
153 /* Others */
154 /* Time */
155 ods graphics / reset width=6.4in height=4.8in imagemap;
156
157 proc sgplot data=WORK.VEHICLE (where=(vehicle_type ^="Sedan" and
158         vehicle_type ^="SUV"));
159     vline crash_time / group=vehicle_type curvelabel;
160     yaxis grid;
161 run;
```

```
162
163 ods graphics / reset;
164
165 /* Weekday */
166 ods graphics / reset width=6.4in height=4.8in imagemap;
167
168 proc sgplot data=WORK.VEHICLE (where=(vehicle_type ^="Sedan" and
169         vehicle_type ^="SUV"));
170     vline crash_weekday / group=vehicle_type curvelabel;
171     yaxis grid;
172 run;
173
174 ods graphics / reset;
175
176 /* Month */
177 ods graphics / reset width=6.4in height=4.8in imagemap;
178
179 proc sgplot data=WORK.VEHICLE (where=(vehicle_type ^="Sedan" and
180         vehicle_type ^="SUV"));
181     vline crash_month / group=vehicle_type curvelabel;
182     yaxis grid;
183 run;
184
```

```
/* Problem Statement 2 SAS Code*/


/* Pre-Processing */
/* import data 'crashes' */

filename MVCc
    '/folders/myfolders/group project/Motor Vehicle Collisions - Crashes.json';
libname MVCc JSON fileref=MVCc;
libname gp '/folders/myfolders/group project';

data gp.crash; set MVCc.data;
run;



/* run every time open the file */

libname gp '/folders/myfolders/group project';
data gp.preprocess_crash ; set gp.crash;
run;


data gp.try_crash; set gp.preprocess_crash;
run;


/* drop the columns*/
data gp.try_crash; set gp.try_crash;
    drop ordinal_root ordinal_data element1 element2 element3 element4 element5 element6 element7 element8
    element12 element15 element16 element17 element18;
run;

/* change the form of the colomns and rename */
data gp.try_crash; set gp.try_crash;
    format time time. ;
    time = input(element10,time.);
    drop element10;

    format collision_ID 8.;
    collision_ID = input(element32,8.);
    drop element32;

    format ppl_injured ppl_killed pedes_injured pedes_killed cyc_injured
    cyc_killed motor_injured motor_killed 4.;
    ppl_injured = input(element19, 4.);
    ppl_killed = input(element20, 4.);
    pedes_injured = input(element21, 4.);
    pedes_killed= input(element22, 4.);
    cyc_injured= input(element23, 4.);
    cyc_killed = input(element24, 4.);
    motor_injured= input(element25, 4.);
    motor_killed= input(element26, 4.);
    drop element19 element20 element21 element22 element23 element24 element25 element26 ;

    rename element9=date element11=BOROUGH element13=LATITUDE element14=LONGITUDE
    element27=contributing_vehicle_1 element28=contributing_vehicle_2 element29=contributing_vehicle_3
    element30=contributing_vehicle_4 element31=contributing_vehicle_5
    element33=vehicle_type_1 element34=vehicle_type_2 element35=vehicle_type_3
    element36=vehicle_type_4 element37=vehicle_type_5;
run;

/* form the date and time variables */
data gp.try_crash; set gp.try_crash;
    date_year = substr(date,1,4);
    date_month = substr(date,6,2);
    date_day = substr(date,9,2);
    drop date;
    format date2 MMDDYY10.;
    date2 = input(cats(date_month,date_day,date_year), MMDDYY10.);
    drop date_month date_day date_year;
    rename date2 = date;
run;


/* count the number of vehicles involved based on the contributing_vehicle */


data gp.try_crash_count; set gp.try_crash;
    do;

```

```
81      if contributing_vehicle_1='' then contributing_vehicle_1_2=0;
82      else contributing_vehicle_1_2 = 1;
83      end;
84      drop contributing_vehicle_1;
85      rename contributing_vehicle_1_2 = contributing_vehicle_1;
86  run ;
87  data gp.try_crash_count; set gp.try_crash_count;
88      do;
89      if contributing_vehicle_2='' then contributing_vehicle_2_2=0;
90      else contributing_vehicle_2_2 = 1;
91      end;
92      drop contributing_vehicle_2;
93      rename contributing_vehicle_2_2 = contributing_vehicle_2;
94  run ;
95  data gp.try_crash_count; set gp.try_crash_count;
96      do;
97      if contributing_vehicle_3='' then contributing_vehicle_3_2=0;
98      else contributing_vehicle_3_2 = 1;
99      end;
100     drop contributing_vehicle_3;
101     rename contributing_vehicle_3_2 = contributing_vehicle_3;
102 run ;
103 data gp.try_crash_count; set gp.try_crash_count;
104     do;
105     if contributing_vehicle_4='' then contributing_vehicle_4_2=0;
106     else contributing_vehicle_4_2 = 1;
107     end;
108     drop contributing_vehicle_4;
109     rename contributing_vehicle_4_2 = contributing_vehicle_4;
110 run ;
111 data gp.try_crash_count; set gp.try_crash_count;
112     do;
113     if contributing_vehicle_5='' then contributing_vehicle_5_2=0;
114     else contributing_vehicle_5_2 = 1;
115     end;
116     drop contributing_vehicle_5;
117     rename contributing_vehicle_5_2 = contributing_vehicle_5;
118 run ;
119
120 data data gp.try_crash_count; set gp.try_crash_count;
121     DO i = 1 to contributing_vehicle_1;
122         vehicle_num_1 = contributing_vehicle_1 + contributing_vehicle_2 +contributing_vehicle_3+contributing_
123     end;
124     drop i contributing_vehicle_1 contributing_vehicle_2 contributing_vehicle_3 contributing_vehicle_4 contril
125 run;
126
127 /* count the number of vehicles involved based on the contributing_vehicle */
128
129 data gp.try_crash_count; set gp.try_crash_count;
130     do;
131     if vehicle_type_1='' then vehicle_type_1_2=0;
132     else vehicle_type_1_2 = 1;
133     end;
134     drop vehicle_type_1;
135     rename vehicle_type_1_2 = vehicle_type_1;
136 run ;
137 data gp.try_crash_count; set gp.try_crash_count;
138     do;
139     if vehicle_type_2='' then vehicle_type_2_2=0;
140     else vehicle_type_2_2 = 1;
141     end;
142     drop vehicle_type_2;
143     rename vehicle_type_2_2 = vehicle_type_2;
144 run ;
145 data gp.try_crash_count; set gp.try_crash_count;
146     do;
147     if vehicle_type_3='' then vehicle_type_3_2=0;
148     else vehicle_type_3_2 = 1;
149     end;
150     drop vehicle_type_3;
151     rename vehicle_type_3_2 = vehicle_type_3;
152 run ;
153 data gp.try_crash_count; set gp.try_crash_count;
154     do;
155     if vehicle_type_4='' then vehicle_type_4_2=0;
156     else vehicle_type_4_2 = 1;
157     end;
158     drop vehicle_type_4;
159     rename vehicle_type_4_2 = vehicle_type_4;
160 run ;
    data gp.try_crash_count; set gp.try_crash_count;
161
```

```
162     do;
163     if vehicle_type_5='' then vehicle_type_5_2=0;
164     else vehicle_type_5_2 = 1;
165     end;
166     drop vehicle_type_5;
167     rename vehicle_type_5_2 = vehicle_type_5;
168 run ;
169
170 data gp.try_crash_count; set gp.try_crash_count;
171     DO i = 1 to vehicle_type_1;
172         vehicle_num_2 = vehicle_type_1 + vehicle_type_2 +vehicle_type_3+vehicle_type_4+vehicle_type_5;
173     end;
174     drop i vehicle_type_1 vehicle_type_2 vehicle_type_3 vehicle_type_4 vehicle_type_5;
175 run;
176
177
178 /* calculatet the maximum number of vehicles involved */
179 data gp.try_crash_count; set gp.try_crash_count;
180     vehicle_num = max(vehicle_num_1,vehicle_num_2);
181     drop vehicle_num_1 vehicle_num_2;
182 run;
183
184 proc export
185   data=GP.TRY_CRASH_COUNT
186   dbms=csv
187   outfile="/folders/myfolders/group project/20200310/crash_count.csv"
188   replace;
189 run;
190
191 /* Pre-processing Done*/
192
193
194
195
196
197
198 /* check the multiple collison in 2019*/
199 data gp.multi_collision; set gp.try_crash_count;
200     where vehicle_num > 2;
201 run;
202
203 data gp.multi_collision2019; set gp.multi_collision;
204     where year(date) = 2019;
205 run;
206
207
208
209 /* load vehicle dataset in 2019 */
210 libname gpvehi '/folders/myfolders/group project/20200310';
211
212 FILENAME REFFILE '/folders/myfolders/group project/20200310/19 data/vehicles19.xlsx';
213
214 PROC IMPORT DATAFILE=REFFILE
215     DBMS=XLSX
216     OUT=GPVEHI.VEHICLE_2019;
217     GETNAMES=YES;
218 RUN;
219
220 PROC CONTENTS DATA=GPVEHI.VEHICLE_2019; RUN;
221
222
223
224 /* do inner join and keep variables we want: cars involved in multi-vehicle-collisions*/
225 PROC SQL;
226 CREATE TABLE GPVEHI.MULTI_VEHICLE_COLLISION_2019
227 AS
228 SELECT MULTI_COLLISION2019.collision_ID, MULTI_COLLISION2019.vehicle_num, MULTI_COLLISION2019.time, MULTI_COLL
229 FROM GPVEHI.VEHICLE_2019 VEHICLE_2019
230 INNER JOIN GP.MULTI_COLLISION2019 MULTI_COLLISION2019
231 ON
232    ( VEHICLE_2019.COLLISION_ID = MULTI_COLLISION2019.collision_ID ) ;
233 QUIT;
234
235
236
237 PROC SORT data=GPVEHI.MULTI_VEHICLE_COLLISION_2019;
238     by collision_ID;
239 run;
240
241
    /* delete the missing data based on vehicle_occupants */
242
```

```
243 data GPVEHI.MULTI_VEHICLE_COLLISION_2019 ; set GPVEHI.MULTI_VEHICLE_COLLISION_2019;
244     if VEHICLE_OCCUPANTS = '' THEN DELETE;
245 RUN;
246
247 proc export
248   data=GPVEHI.MULTI_VEHICLE_COLLISION_2019
249   dbms=csv
250   outfile="/folders/myfolders/group project/20200310/multi_vehicle_collision_2019.csv"
251   replace;
252 run;
253
254 /* count the number of crashes in each day in 2019 */
255 PROC SQL;
256 CREATE TABLE gpvehi.VehicleTimeLineByDay
257 AS
258 SELECT VEHICLE_2019.crash_date, COUNT(DISTINCT VEHICLE_2019.COLLISION_ID)
259 AS COLLISION_NUM
260 FROM GPVEHI.VEHICLE_2019 VEHICLE_2019
261 GROUP BY VEHICLE_2019.crash_date;
262 QUIT;
263
264
265
266
267 /* all plots used in problem statement 3 in the final report are produced by excel*/
268 /* plot the vehicle in 2019 by day*/
269 ods graphics / reset width=6.4in height=4.8in imagemap;
270
271 proc sort data=GPVEHI.VEHICLETIMELINEBYDAY out=_SeriesPlotTaskData;
272     by CRASH_DATE;
273 run;
274
275 proc sgplot data=_SeriesPlotTaskData;
276     title height=14pt "Vehicles involved in crashes in 2019 by day";
277     series x=CRASH_DATE y=COLLISION_NUM /;
278     xaxis grid;
279     yaxis grid;
280 run;
281
282 ods graphics / reset;
283 title;
284
285 proc datasets library=WORK noprint;
286     delete _SeriesPlotTaskData;
287     run;
288
289
290
291 /* export it to count the vehecles in each month */
292
293 proc export
294   data=gpvehi.VehicleTimeLine
295   dbms=csv
296   outfile="/folders/myfolders/group project/20200310/multi-timeline-2019.csv"
297   replace;
298 run;
299
300 /* Used excel to adjust the date and calculate the number of crashes by month */
301
302
303
304
305 /* import the setup data and plot */
306
307 FILENAME REFFILE '/folders/myfolders/group project/20200310/plot_vehecle2019.csv';
308
309 PROC IMPORT DATAFILE=REFFILE
310     DBMS=CSV
311     OUT=GPVEHI.PlotVehicle2019;
312     GETNAMES=YES;
313 RUN;
314
315 PROC CONTENTS DATA=GPVEHI.PlotVehicle2019; RUN;
316
317
318 /* plot the vehicle in 2019 by month*/
319 ods graphics / reset width=6.4in height=4.8in imagemap;
320
321 proc sort data=GPVEHI.PLOTVEHICLE2019 out=_SeriesPlotTaskData;
322     by MMYY;
    run;
323
```

```
324
325 proc sgplot data=_SeriesPlotTaskData;
326     title height=14pt "Vehicles involved in crashes in 2019 by month";
327     series x=MMYY y=COLLISION_ID /;
328     xaxis grid;
329     yaxis grid label="Vehicles_involed";
330 run;
331
332 ods graphics / reset;
333 title;
334
335 proc datasets library=WORK noprint;
336     delete _SeriesPlotTaskData;
337     run;
338
339
340
341 /* plot bar chart to show different number of vehicles involved
342 in one crash in different years*/
343 data GP.TRY_CRASH_COUNT_plot; set GP.TRY_CRASH_COUNT;
344     year = year(date);
345 run;
346
347 ods graphics / reset width=6.4in height=4.8in imagemap;
348
349 proc sgplot data=GP.TRY_CRASH_COUNT_plot;
350     vbar year / group=vehicle_num groupdisplay=cluster datalabel;
351     yaxis grid;
352 run;
353
354 ods graphics / reset;
355
356
357 /* plot pie charts to show collision type in different year */
358 data GP.TRY_CRASH_COUNT_plot; set GP.TRY_CRASH_COUNT;
359     year = year(date);
360     format collision_type $8.;
361     if vehicle_num <3 then collision_type = 'Simple';
362     else collision_type = 'Multi';
363 run;
364
365     /* by changing the year to get percentage in different year*/
366 proc template;
367     define statgraph SASStudio.Pie;
368         begingraph;
369         layout region;
370         piechart category=collision_type / stat=pct;
371         endlayout;
372         endgraph;
373     end;
374 run;
375
376 ods graphics / reset width=6.4in height=4.8in imagemap;
377
378 proc sgrender template=SASStudio.Pie
379         data=GP.TRY_CRASH_COUNT_PLOT (where=(year=2019));
380 run;
381
382 ods graphics / reset;
383
384
385
386 /* plot grams in 2019 by different variables */
387 /* plot the line gram to show the change of time */
388 data GPVEHI.MULTI_VEHICLE_COLLISION_2019_p; set GPVEHI.MULTI_VEHICLE_COLLISION_2019;
389     hour = hour(time);
390 run;
391
392 ods graphics / reset width=6.4in height=4.8in imagemap;
393
394 proc sgplot data=GPVEHI.MULTI_VEHICLE_COLLISION_2019_p;
395     vline hour /;
396     yaxis grid;
397 run;
398
399 ods graphics / reset;
400
401
402 /* plot the pie chart to show the percentage of driver's gender */
403 proc template;
    define statgraph SASStudio.Pie;
404
```

```
405         begingraph;
406         layout region;
407         piechart category=DRIVER_SEX / stat=pct;
408         endlayout;
409         endgraph;
410     end;
411 run;
412 
413 ods graphics / reset width=6.4in height=4.8in imagemap;
414 
415 proc sgrender template=SASStudio.Pie data=GPVEHI.MULTI_VEHICLE_COLLISION_2019;
416 run;
417 
418 ods graphics / reset;
419 
420 
421 /* plot the pie chart to show the percentage of travel direction */
422 proc template;
423     define statgraph SASStudio.Pie;
424         begingraph;
425         layout region;
426         piechart category=TRAVEL_DIRECTION / stat=pct;
427         endlayout;
428         endgraph;
429     end;
430 run;
431 
432 ods graphics / reset width=6.4in height=4.8in imagemap;
433 
434 proc sgrender template=SASStudio.Pie data=GPVEHI.MULTI_VEHICLE_COLLISION_2019;
435 run;
436 
437 ods graphics / reset;
438 
439 /* plot the pie chart to show the percentage of borough */
440 proc template;
441     define statgraph SASStudio.Pie;
442         begingraph;
443         layout region;
444         piechart category=BOROUGH / stat=pct;
445         endlayout;
446         endgraph;
447     end;
448 run;
449 
450 ods graphics / reset width=6.4in height=4.8in imagemap;
451 
452 proc sgrender template=SASStudio.Pie data=GPVEHI.MULTI_VEHICLE_COLLISION_2019;
```

```sas
/* Problem Statement 3 SAS Code */
Libname PJ '/folders/myfolders/SAS Project';


FILENAME REFFILE '/folders/myfolders/SAS Project/Crash.xlsx';

PROC IMPORT DATAFILE=REFFILE
    DBMS=xlsx
    OUT=PJ.Crash;
    GETNAMES=YES;
RUN;

FILENAME REFFILE1 '/folders/myfolders/SAS Project/Direction.xlsx';

PROC IMPORT DATAFILE=REFFILE1
    DBMS=xlsx
    OUT=PJ.Direction;
    GETNAMES=YES;
RUN;

data PJ.crash; set pj.crash;

Drop crash_time;

run;

/*sort two dataset by collision_id */
proc sort data=PJ.crash;
    by collision_id;
run;
proc sort data=PJ.direction;
    by collision_id;
run;


/* Merge dataset by collision_id */
data PJ.crashmerge;
 merge PJ.crash PJ.direction;
 by collision_id;
run;

/*Turn direction into numeric values */
proc format;
    invalue direction
        "East" = 0
        "Northeast" = 1
        "North" = 2
        "Northwest" = 3
        "West" = 4
        "Southwest" = 5
        "South" = 6
        "Southeast" = 7
        "Unknown" = 8;

```

```
54 run;
55
56 data PJ.crashmerge; set PJ.crashmerge;
57
58 Direction = input(travel_direction, direction.);
59 Drop zip_code location;
60 run;
61
62 data PJ.crashmerge;
63 run;
64
65
66
67 Proc corr data=PJ.crashmerge;
68 run;
69
70
71 /* Remove missing values */
72 data PJ.crashmerge;
73 set PJ.crashmerge;
74 if compress(cats(of _all_),'')='' then delete;
75 run;
76
77
78 Libname PJ '/folders/myfolders/Project';
79
80 FILENAME REFFILE '/folders/myfolders/Project/Crashmerge2.csv';
81
82 PROC IMPORT DATAFILE=REFFILE
83     DBMS=csv
84     OUT=PJ.Crashmerge2;
85     GETNAMES=YES;
86 RUN;
87
88 data PJ.crashmerge2; set pj.crashmerge2;
89 drop var1;
90 run;
91
92 Proc corr data=PJ.crashmerge2;
93 run;
94
95
96
97 proc reg data=pj.crashmerge2;
98
99 model crash_time_1 = direction longitude latitude;
100 run;
101
102
103
```

```
/* Problem Statement 4 SAS Code */
libname gp '/folders/myshortcuts/Project_Only_new/gp';

/*Creating the table to work for the analysis. Removed several variables which wouldn't make much sense to analysis and where
PROC SQL;
CREATE TABLE gp.persons
AS
SELECT PERSONS.COLLISION_ID, PERSONS.CRASH_DATE, PERSONS.CRASH_TIME, PERSONS.PERSON_TYPE, PERSONS.PERSON_INJURY, PERSONS.PERS
FROM WORK.PERSONS PERSONS
WHERE
   (
      (
         (
            (
               (
                  (
                     ( PERSONS.COLLISION_ID IS NOT MISSING AND PERSONS.CRASH_DATE IS NOT MISSING ) AND PERSONS.CRASH_TIME IS
                  ) AND PERSONS.PERSON_TYPE IS NOT MISSING
               ) AND PERSONS.PERSON_INJURY IS NOT MISSING
            ) AND PERSONS.PERSON_AGE IS NOT MISSING
         ) AND PERSONS.PED_ROLE IS NOT MISSING
      ) AND PERSONS.PERSON_SEX IS NOT MISSING
   ) ;
QUIT;


data gp.persons; set gp.persons;
run;

data gp.workset; set gp.persons;
run;

/*Injured person observations*/
PROC SQL;
CREATE TABLE GP.Injured
AS
SELECT WORKSET.PERSON_TYPE, WORKSET.PERSON_INJURY, WORKSET.PERSON_SEX
FROM GP.WORKSET WORKSET
WHERE
   ( WORKSET.PERSON_INJURY = 'Injured' ) ;
QUIT;

/* Bar graph to show the split in injured by person type*/

ods graphics / reset width=6.4in height=4.8in imagemap;

proc sgplot data=GP.INJURED;
    vbar PERSON_INJURY / group=PERSON_TYPE groupdisplay=cluster;
    yaxis grid;
run;

ods graphics / reset;

/* Boxplot to show what time of the day is dangerous for people types*/

ods graphics / reset width=6.4in height=4.8in imagemap;

proc sgplot data=GP.TIME_OF_DAY;
    vbox CRASH_TIME / category=PERSON_TYPE;
    yaxis grid;
run;

ods graphics / reset;

/* Merging datasets Persons and Crashes */
PROC SQL;
CREATE TABLE GP.MERGE
AS
SELECT WORKSET.COLLISION_ID, WORKSET.PERSON_TYPE, WORKSET.PERSON_INJURY, WORKSET.PERSON_AGE, WORKSET.PERSON_SEX, CRASHES.BORO
FROM GP.WORKSET WORKSET
INNER JOIN GP.CRASHES CRASHES
ON
   ( WORKSET.COLLISION_ID = CRASHES.COLLISION_ID ) ;
QUIT;

/* Finding out which borough is dangerous by person type */
```

```
ods graphics / reset width=6.4in height=4.8in imagemap;

proc sgplot data=GP.MERGE;
    vbar PERSON_TYPE / group=BOROUGH groupdisplay=cluster;
    yaxis grid;
run;
```