Solutions to Practices

3. Performing a One-to-Many Merge

```
proc sort data=pg2.np_codelookup out=work.codesort;
    by ParkCode;
run;

proc sort data=pg2.np_2016traffic(rename=(Code=ParkCode))
          out=work.traf2016Sort;
    by ParkCode month;
run;

data work.trafficStats;
    merge work.traf2016Sort
          work.codesort;
    by ParkCode;
    drop Name_Code;
run;
```

4. Writing Matches and Nonmatches to Separate Tables

```
proc sort data=pg2.np_CodeLookup
     out=work.sortedCodes;
    by ParkCode;
run;

proc sort data=pg2.np_2016
     out=work.sorted_code_2016;
    by ParkCode;
run;

data work.parkStats(keep=ParkCode ParkName Year Month DayVisits)
     work.parkOther(keep=ParkCode ParkName);
    merge work.sorted_code_2016(in=inStats) work.sortedCodes;
    by ParkCode;
    if inStats=1 then output work.parkStats;
    else output work.parkOther;
run;
```

5. Combining Multiple Tables with Different Matching Columns

Why must you use IF instead of a WHERE statement?

You must use a subsetting IF statement because the DayVisits column is in only one of the tables in the MERGE statement.

```
proc sort data=pg2.np_codelookup
          out=sortnames(keep=ParkName ParkCode);
    by ParkName;
run;

proc sort data=pg2.np_final out=sortfinal;
    by ParkName;
run;

data highuse(keep=ParkCode ParkName);
    merge sortfinal sortnames;
    by ParkName;
    if DayVisits ge 5000000;
run;

proc sort data=pg2.np_species
          out=birds(keep=ParkCode Species_ID Scientific_Name
                         Common_Names);
    by ParkCode Species_ID;
    where Category='Bird' and Abundance='Common';
run;

proc sort data=highuse;
    by ParkCode;
run;
data work.birds_largepark;
    merge birds highuse(in=inPark);
    by ParkCode;
    if inPark=1;
run;
```

End of Solutions