MET MA 603: SAS Programming and Applications

Logical Statements

IF-THEN Statements

IF-THEN statements allow for statements to be executed conditionally. In other words, a statement may be executed for some observations, but not for others.

IF-THEN statements have two parts: a condition and an action. The condition is a logical expression that is evaluated for each observation, and has a result that is either *true* or *false*. The condition uses characters such as =, <, >, <=, >=, ~= (not equal). The action is a statement that will only be executed when the condition is *true*.

```
data   occupancy;
set          mydata1.occupancy;
if     DogBreed = “” then DogBreed = “NO DOGS” ;
run ;
```

ELSE Statement

In the previous example, the action statement is either executed, or it is not. There are situations where one statement should be executed if the condition is true, and a different statement should be executed if the condition is false. The **ELSE statement** allows for a statement to be executed when the IF condition is false.

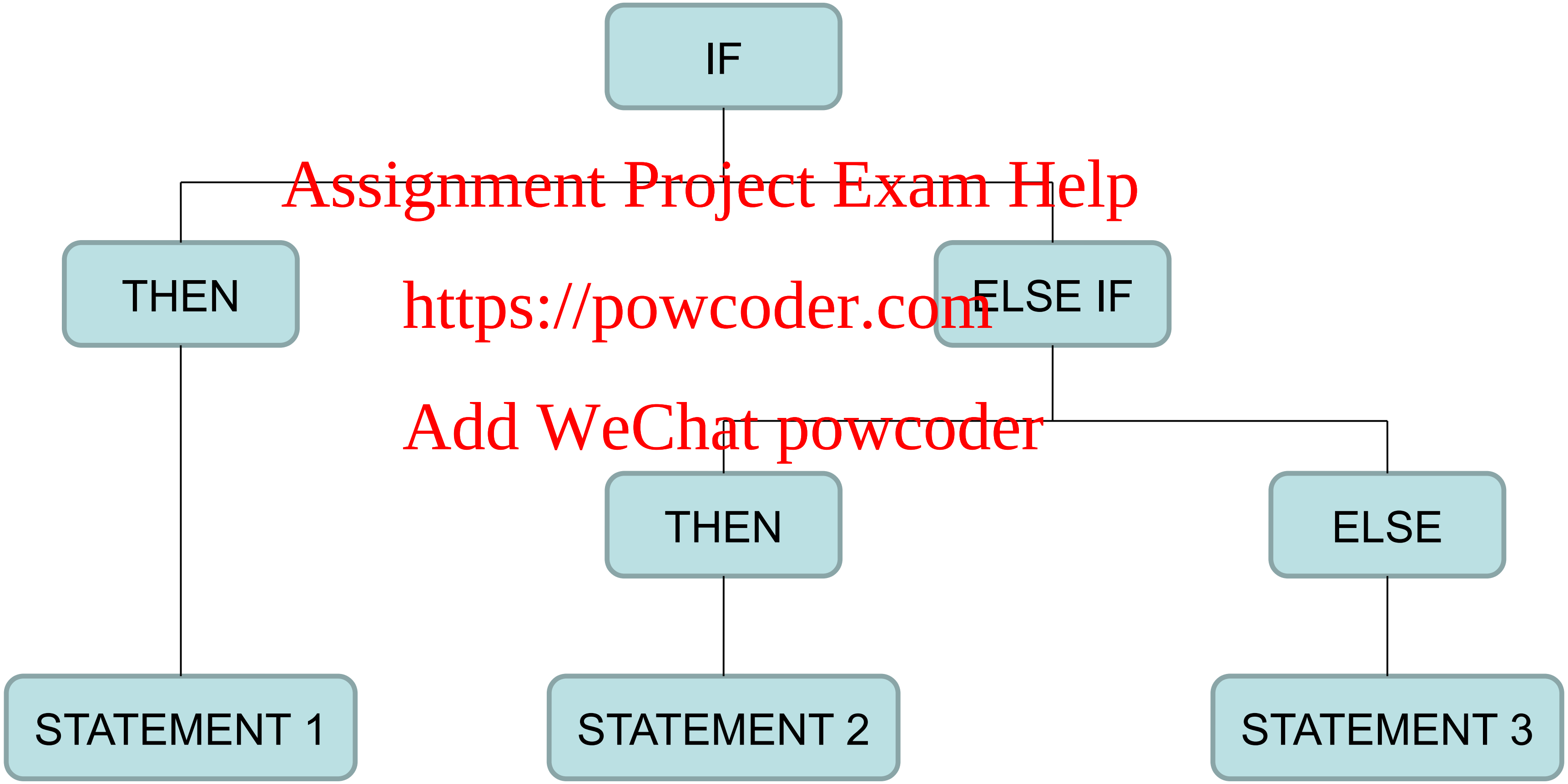
```
      data   occupancy;
      set          mydata1.occupancy;
      if     dogs > residents    then House_Type = “More
Dogs” ;
                                               else
House_Type = “More People” ;
      run ;
```

ELSE IF

The previous example used one condition. There are situations where more than one condition needs to be evaluated. ELSE IF allows for additional conditions to be tested. Note that an ELSE IF condition is only evaluated when all preceding IF and ELSE IF conditions are false. Once a true condition is found, the corresponding action is executed and the statement is finished.

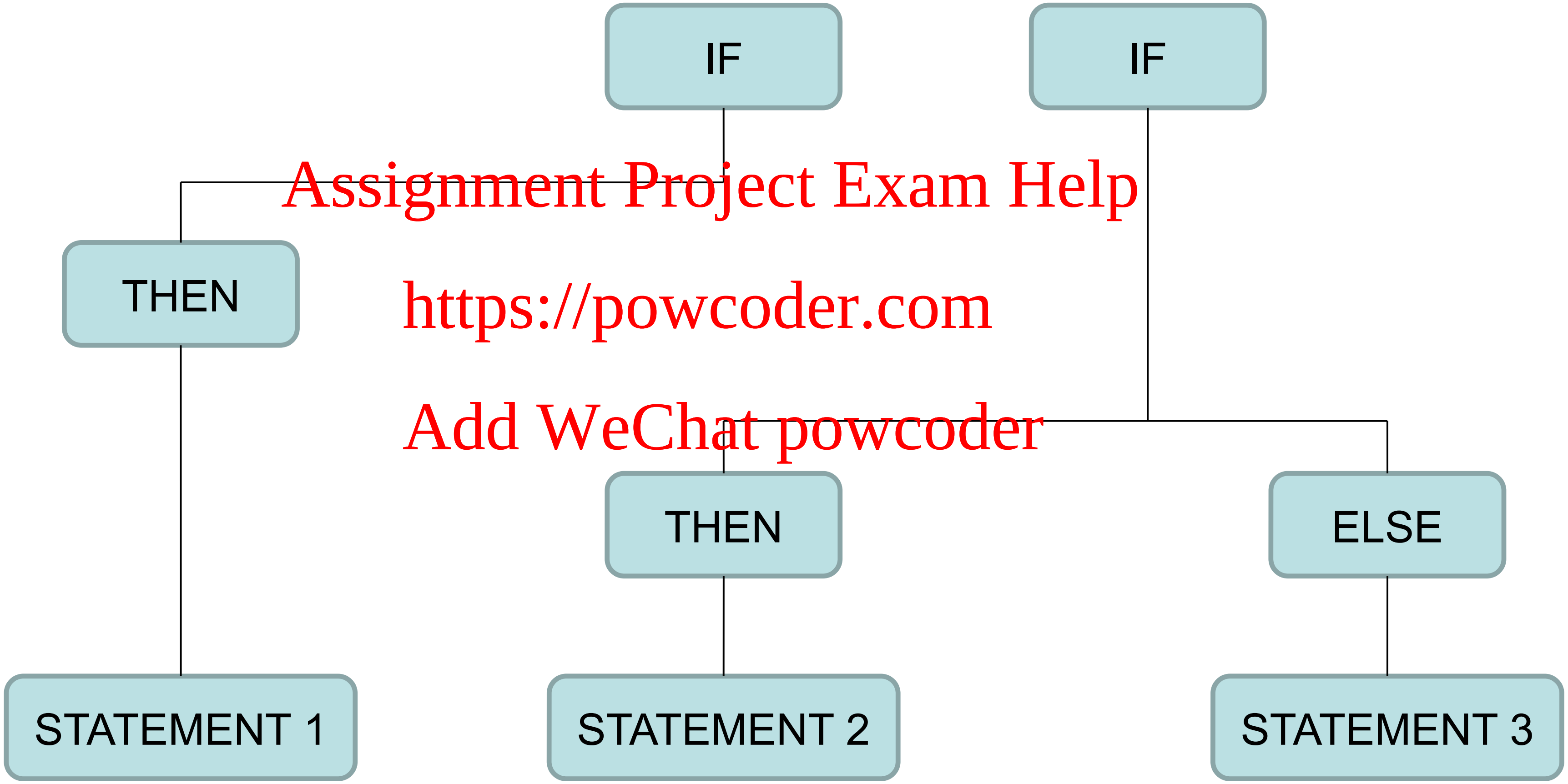
```
      data    occupancy;
      set            mydata1.occupancy;
      if             dogs > residents then House_Type =
“More Dogs” ;
      else if dogs<residents then House_Type = “More
People”
                                                         else
House_Type = “Same” ;
      run ;
```

Graphical Representation of IF-ELSE IF-ELSE Structure



Only one of the Statements will be executed.

Graphical Representation of Parallel IF Statements

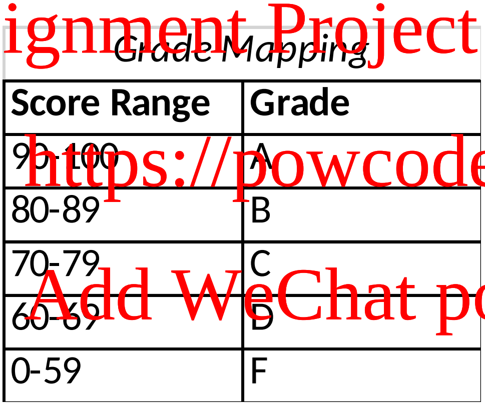


Statements 1 might be executed. One of Statements 2 and 3 will be executed.

Practice

Use the scores1.sas7bdat dataset.

Create a variable called Grade based on the following mapping:



Grade Mapping

Score Range	Grade
90-100	A
80-89	B
70-79	C
60-69	D
0-59	F

Logical Operators

AND, **OR**, **IN** and **NOT IN** are logical operators that allow for more sophisticated conditions to be tested.

Example of the AND logical operator:

```
      data occupancy;
      set      mydata1.occupancy;
      if      residents=1 AND dogs=0 then Lonely Person
= "Yes";

      else Lonely Person = "No";
      run ;
```

Example of the OR logical operator:

```
      data   occupancy;
      set           mydata1.occupancy;
      if            residents>1 OR dogs>0 then Lonely
Person = "No";
```

Logical Operators (continued)

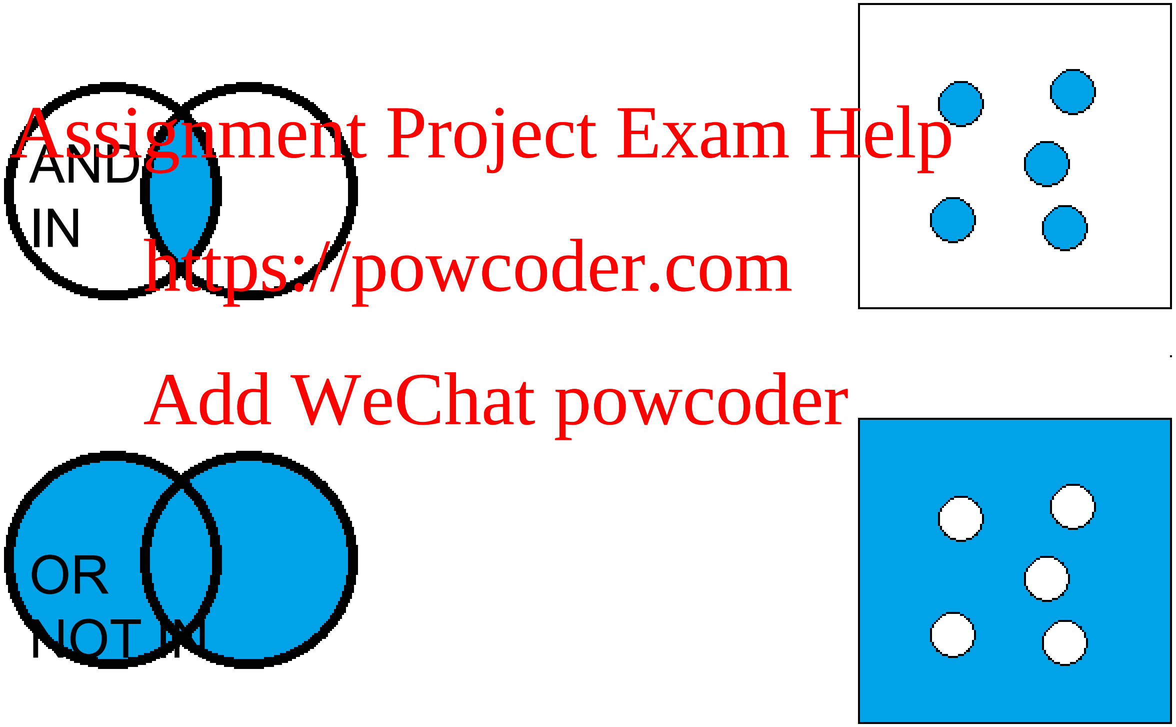
Example of the IN logical operator:

```
        data   occupancy;
        set          mydata1.occupancy;
        if           dogbreed in ("NONE OF THE ABOVE",
                                                   "None of
the above")
                                                   then
BreedType = "Safe Breed";
        run ;
```

Example of the NOT IN logical operator:

```
        data   occupancy;
        set          mydata1.occupancy;
        if           dogbreed not in ("NONE OF THE ABOVE",
                                                   "None of
the above", "")
                                                   then
BreedType = "Unsafe Breed";
```

Graphical Representation of Logical Operators



AND

IN

OR

NOT IN

DO-END Statements

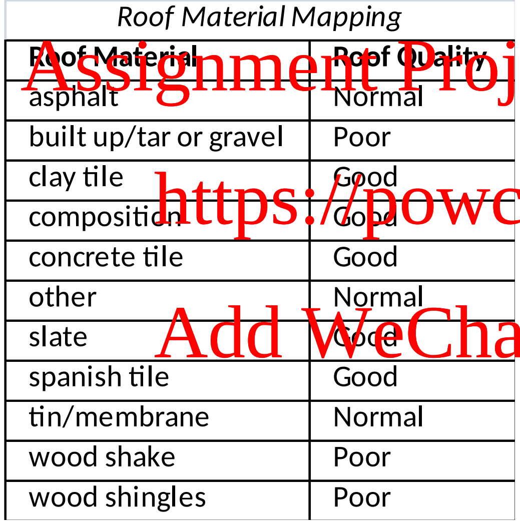
Sometimes more than once action statement needs to be executed when a condition is met. The **DO and END statements** indicate to SAS that all statements inside this structure are to be executed when the condition is met.

```
        data      occupancy;
        set              mydata1.occupancy;
        length DogInd $2;
        if        dogs > 0 then DogInd = “1” ;
                                        else      do;

DogInd = “0” ;

DogBreed = “No Dogs”;
                                                       end;
        run ;
```

Practice

Use the policy_info.sas7bdat dataset.

Create a variable called RoofQuality based on the following mapping:



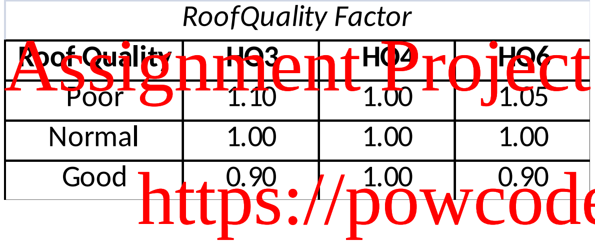
Roof Material Mapping

Roof Material	Roof Quality
asphalt	Normal
built up/tar or gravel	Poor
clay tile	Good
composition	Good
concrete tile	Good
other	Normal
slate	Good
spanish tile	Good
tin/membrane	Normal
wood shake	Poor
wood shingles	Poor

(see the next slide for the rest of the problem)

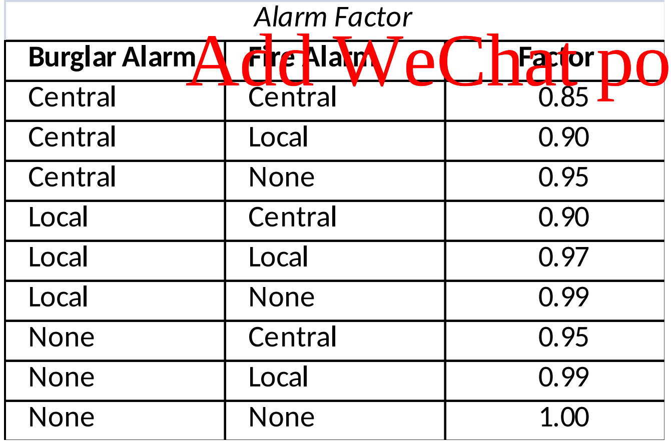
Practice (continued)

Based on each policy's characteristics, assign the Roof and Alarm factors based on the following criteria:



RoofQuality Factor

Roof Quality	HO3	HO4	HO6
Poor	1.10	1.00	1.05
Normal	1.00	1.00	1.00
Good	0.90	1.00	0.90



Alarm Factor

Burglar Alarm	Fire Alarm	Factor
Central	Central	0.85
Central	Local	0.90
Central	None	0.95
Local	Central	0.90
Local	Local	0.97
Local	None	0.99
None	Central	0.95
None	Local	0.99
None	None	1.00

Your result should match the discount.sas7bdat dataset.

Readings

- Textbook sections 3.5, 3.6