# Table of Contents

The work situation ................................................................................................................. 6  
Creating our First drop list ................................................................................................. 8  
Creating a "Dependent- Diminishing" drop list ................................................................. 10  
  The preparation list ........................................................................................................... 11  
  Referencing a spilled array ............................................................................................... 11  
Preparing for our drop lists with multiple levels of dependencies .............................. 13  
  Three functions - Same concept ..................................................................................... 14  
  Unique list of Floors ....................................................................................................... 15  
  Creating the third drop list .............................................................................................. 16  
Scratch the limit of Data Validation in Z4 ................................................................. 18  
List of Owned Suites in any Floor .................................................................................... 19  
  The Six possible options ................................................................................................. 22  
Preparing for the Switch function: .................................................................................. 29  
  Hide contents of Cell AA1 ............................................................................................. 33  
Resetting 2 drop lists (Y3 & Z3) with a code ............................................................... 33  
  Open the VBE .................................................................................................................. 33  
  Writing the Code ............................................................................................................. 35  
Getting Information based upon selection ..................................................................... 35  
  First VLOOKUP ............................................................................................................... 37  
  Second VLOOKUP ........................................................................................................... 38  
  Combining the 2 VLOOKUP functions .......................................................................... 39  
  A little issue to avoid ........................................................................................................ 39
A second IFERROR function................................................................. 39
An Input Message .................................................................................. 39
Finalizing the project................................................................................ 40
Two conditional formatting rules............................................................. 40
Hiding Columns....................................................................................... 41
Zoom In and Out ..................................................................................... 41
Create 2 macros for the Zooming Functionality .................................... 42
Assigning Macros................................................................................... 42
## Table of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SOURCE DATA</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>PROJECT RANGE</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>PREPARATION AREA</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>DATA VALIDATION</td>
<td>9</td>
</tr>
<tr>
<td>5</td>
<td>LIST VALUES FOR THE FIRST DROP LIST</td>
<td>9</td>
</tr>
<tr>
<td>6</td>
<td>INPUT MESSAGE</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>THE FIRST DROP LIST IN Y2</td>
<td>10</td>
</tr>
<tr>
<td>8</td>
<td>COUNTIF AND FILTER FUNCTIONS</td>
<td>11</td>
</tr>
<tr>
<td>9</td>
<td>DEPENDENT DATA VALIDATION</td>
<td>12</td>
</tr>
<tr>
<td>10</td>
<td>TESTING THE SECOND DROP LIST</td>
<td>13</td>
</tr>
<tr>
<td>11</td>
<td>THE SOURCE LIST</td>
<td>14</td>
</tr>
<tr>
<td>12</td>
<td>FLOOR/ SUITE/ RESIDENT</td>
<td>15</td>
</tr>
<tr>
<td>13</td>
<td>UNIQUE LIST OF FLOORS</td>
<td>16</td>
</tr>
<tr>
<td>14</td>
<td>SWITCH FUNCTION FOR 3 OPTIONS</td>
<td>18</td>
</tr>
<tr>
<td>15</td>
<td>THREE POSSIBILITIES IN Y3 DROP LIST</td>
<td>19</td>
</tr>
<tr>
<td>16</td>
<td>OWNED SUITES IN FLOOR 01</td>
<td>20</td>
</tr>
<tr>
<td>17</td>
<td>LIST OF OWNED SUITES IN EACH FLOOR</td>
<td>20</td>
</tr>
<tr>
<td>18</td>
<td>RESIDENT NAMES IN OWNED SUITES IN FLOOR 01</td>
<td>21</td>
</tr>
<tr>
<td>19</td>
<td>OWNERS LIST IN EACH FLOOR</td>
<td>22</td>
</tr>
<tr>
<td>20</td>
<td>ONE SUITE RETURNED</td>
<td>23</td>
</tr>
<tr>
<td>21</td>
<td>LIST OF OWNED SUITES FOR A SELECTED FLOOR</td>
<td>24</td>
</tr>
<tr>
<td>22</td>
<td>FIRST RESIDENT NAME FOR A SELECTED FLOOR</td>
<td>25</td>
</tr>
<tr>
<td>23</td>
<td>ALL RESIDENT NAMES OF A SELECTED FLOOR</td>
<td>25</td>
</tr>
<tr>
<td>24</td>
<td>FLOOR NUMBER OF A SELECTED SUITE</td>
<td>26</td>
</tr>
<tr>
<td>25</td>
<td>RESIDENT NAME FOR A SELECTED SUITE</td>
<td>27</td>
</tr>
<tr>
<td>26</td>
<td>THE SIX PREPARATION FUNCTIONS</td>
<td>28</td>
</tr>
<tr>
<td>27</td>
<td>COMBINE SELECTIONS FROM THE 2 DROP LISTS</td>
<td>29</td>
</tr>
<tr>
<td>28</td>
<td>OPTION1 FLOOR- SUITE</td>
<td>31</td>
</tr>
<tr>
<td>29</td>
<td>OPTION2 FLOOR- RESIDENT</td>
<td>31</td>
</tr>
<tr>
<td>30</td>
<td>OPTION3 SUITE-FLOOR</td>
<td>32</td>
</tr>
<tr>
<td>31</td>
<td>OPTION4 SUITE-RESIDENT</td>
<td>32</td>
</tr>
<tr>
<td>32</td>
<td>OPTIONS RESIDENT- SUITE</td>
<td>33</td>
</tr>
<tr>
<td>33</td>
<td>OPTION6 RESIDENT- FLOOR</td>
<td>33</td>
</tr>
<tr>
<td>34</td>
<td>SWITCHING TO VBE</td>
<td>34</td>
</tr>
<tr>
<td>35</td>
<td>SELECT THE CHANGE EVENT</td>
<td>34</td>
</tr>
<tr>
<td>36</td>
<td>DELETE THE SELECTION CHANGE EVENT</td>
<td>34</td>
</tr>
<tr>
<td>37</td>
<td>THE CHANGE EVENT CODE</td>
<td>35</td>
</tr>
<tr>
<td>38</td>
<td>THE R1C1 REFERENCE</td>
<td>36</td>
</tr>
<tr>
<td>39</td>
<td>COMPARE COLUMN LETTERS AND NUMBERS</td>
<td>37</td>
</tr>
<tr>
<td>40</td>
<td>TESTING THE FIRST VLOOKUP</td>
<td>38</td>
</tr>
<tr>
<td>41</td>
<td>TESTING THE SECOND VLOOKUP</td>
<td>38</td>
</tr>
</tbody>
</table>
The Magic of Data Validation

FIGURE 42- INPUT MESSAGE IN Z3
FIGURE 43- PAST DUE CONDITIONAL FORMATTING
FIGURE 44- OVER PMT CONDITIONAL FORMATTING
FIGURE 45- RESIZE AND REPOSITION ICONS
FIGURE 46- ASSIGN A MACRO TO ZOOM IN
FIGURE 47- THE ASSIGN MACRO DIALOG BOX
FIGURE 48- ASSIGNING A ZOOM OUT MACRO
The Magic of Data Validation

Introduction

Dynamic Data Validation… is a Game changer in Excel, at the basic level it controls what a user can enter or cannot enter in a cell. It also makes data entry easier and ensures the integrity of your worksheet by avoiding typing errors when users can simply select an option from a drop list.

However, in more advanced scenario drop lists can make a regular worksheet look as a dynamic dashboard.

With drop lists, you can select input values for formulas and functions, you can switch charts or source values for Power Query, You can create random quizzes or name sheets and you can also have the contents change based on another drop list.

In this article, we explore the full potential of drop lists by creating basic drop lists, diminishing drop lists and then we create dependent drop lists with 6 levels of dependency. We will learn about the extreme power of combining Drop lists with Dynamic arrays and touch the limit of what Data Validation can do.

So, Let's dive in

The work situation

Skymark is a nice high rise building of 35 floors and 8 units on each floor. While most units are owned, few units are rented. The building is well maintained due to the monthly fees each resident must pay beside some individual repair charges on some units.

The property management board wants to track the amounts paid by Owners and whether there are amounts past due or may be over payment.

To easily retrieve the information, they need a simple yet powerful dynamic way to search by Floor, by Suite number or by Resident's name.
So we created a set of drop lists with up to 6 levels of dependency that can extract the information by any search criteria

You can download the exercise File and follow along.

The spreadsheet we have, shows a list of 23 columns and 274 rows.

**The columns store:**

| Floor, Suite number, Resident name, Type (Owned/Rental), Maintenance, Repairs, 12 Months collection, Total Paid, Total due, Past Due, Over Payment. |

Rental units pay their obligations with postdated cheques, so we are mainly concerned about tracking payments for Owned units.

We will create our solution in the same worksheet in the range Y1:AD3 where we just have some labels in row 2
We also need to have a **Preparation Area** which is the real kitchen where we prepare for our work. My preparation area will be to the right side of column AF.

The user should be able to search either by Floor, Suite or Resident, so we start by creating a basic Drop List in Cell Y2 for the 3 options (available in AG2:AG4)

**Creating our First drop list**
Select cell Y2.

Then click on the Data Tab of the Ribbon ► and click on Data Validation
Under "Allow", select "List". With the mouse pointer in the source ►click and drag to select AG2:AG4 (the range having the source values).

Click on the "Input Message" Tab ►enter a "Title" >"Select an option to start search"

Note: The limit for the title text is 32 characters

In the "Input Message" box, type

"You can initiate your search either by Floor, Suite number or Resident's name"

Hit OK
The first basic drop list is created and looks like this:

![Image of a basic drop list]

Creating a "Dependent- Diminishing" drop list
My next step is to replace the static text in cell Z2 with a Dependent Drop list. The contents of this drop list exclude the option you selected in Y2 (e.g. if in Y2 the user selects "Floor" then the drop list in Z2 will exclude "Floor"
and show only the 2 other options "Suite" and "Resident". This requires creating a preparation list.

**THE PREPARATION LIST**

Select cell AH2 in Cell AH2 create a Dynamic Array function (that spills to the cells below)

\[ \text{=FILTER(AG2:AG4,COUNTIF(Y2,AG2:AG4)<1)} \]

Function Arguments:

\[ \text{=FILTER(array,include,if\_empty)} \]
\[ \text{=COUNTIF(range,criteria)} \]

The COUNTIF function looks at cell Y2 (currently "Floor" is selected) and counts the number of the selected option in the range AG2:AG4 if it is less than 1 ► returning a TRUE for the unselected options and False for the selected option.

Then we wrap the COUNTIF function in a FILTER function that returns the 2 unselected options (The TRUE values of the COUNTIF).

![Figure 8- COUNTIF and FILTER functions](image)

**REFERENCING A SPILLED ARRAY**

Now, let's create the Dependent Diminishing drop list in cell Z2 referencing the result of the previous function. To reference the result of a Dynamic Array function:

We Select Only the Top Cell (where the function lives)

Followed by a number Sign #
Note: The number sign is a Spilled array symbol. We may call it pound sign, Hashtag and refers to the entire range returned by the dynamic array function.

Select Z2  Open the Data Validation dialog box (as before)

Note: You can use the shortcut ALT > D > L followed by TAB > L > Tab sequentially

![Data Validation dialog boxes](image)

**Figure 9- Dependent Data Validation**

In the Source box, Select AH2 followed by # for the source (=$AH$2#)

On the Input Message tab, enter a Title and Input Message ► Hit OK

Test your second drop list in cell Z2 while changing options from the first drop list in cell Y2.
Preparing for our drop lists with multiple levels of dependencies

Our next step is to prepare for our second level of dependent drop lists in Y3 (3 levels of dependency) and Z3 (6 levels of dependency) by creating **15 Dynamic Array functions** in the preparation area. Remember, dynamic array functions:

- Spill to the adjacent cells (in one single direction).
- If there are obstacles in the spill range, they return a #SPILL! Error.
- They do not require locking cell references (unless you copy them in the other direction after spilling)
- The function lives in one single cell (other spilled cells are greyed out).
- These functions are available in Microsoft 365

Remember the source list stores the **Floor** in column A, the **Suite** number in column B, the **Resident** name in Column C, the **Type** in column D
**Figure 11- The source list**

**THREE FUNCTIONS - SAME CONCEPT**

**Function #1**

In cell **AG19**: Extracts the Floor of the **Owned** suites (excluding the Rental suites)

=FILTER(A3:A274,D3:D274="Owned")

**Function #2**

In cell **AH19**: Extracts the Suite number of the **Owned** suites

=FILTER(B3:B274,D3:D274="Owned")

**Function #3**

In cell **AI19**: Extracts the Resident name of the **Owned** suites

=FILTER(C3:C274,D2:D273="Owned")
### UNIQUE LIST OF FLOORS

Assuming that we search first by **Floor** in the first drop list in Y2 then, the contents of the drop list below it in Y3 should show a unique list of **Floor**. So, our next preparation function will extract the unique list of Floor twice in Rows #3 & #13 as follows:

**Function #4**

<table>
<thead>
<tr>
<th>Floor</th>
<th>Suites</th>
<th>Residents</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>0101</td>
<td>L Leroux</td>
</tr>
<tr>
<td>01</td>
<td>0102</td>
<td>K Hassab</td>
</tr>
<tr>
<td>01</td>
<td>0103</td>
<td>U Leedham</td>
</tr>
<tr>
<td>01</td>
<td>0104</td>
<td>J Yazdani</td>
</tr>
<tr>
<td>01</td>
<td>0106</td>
<td>Q Cole</td>
</tr>
<tr>
<td>01</td>
<td>0107</td>
<td>Q Kosareva</td>
</tr>
<tr>
<td>01</td>
<td>0108</td>
<td>X Cuenta</td>
</tr>
<tr>
<td>02</td>
<td>0201</td>
<td>I Jin</td>
</tr>
<tr>
<td>02</td>
<td>0202</td>
<td>O Nelson</td>
</tr>
<tr>
<td>02</td>
<td>0203</td>
<td>F Pathak</td>
</tr>
<tr>
<td>02</td>
<td>0204</td>
<td>S Harding</td>
</tr>
<tr>
<td>02</td>
<td>0205</td>
<td>E Gretton</td>
</tr>
<tr>
<td>02</td>
<td>0208</td>
<td>Z Sparkes</td>
</tr>
<tr>
<td>03</td>
<td>0301</td>
<td>P Wong</td>
</tr>
<tr>
<td>03</td>
<td>0302</td>
<td>K Sung</td>
</tr>
<tr>
<td>03</td>
<td>0303</td>
<td>H Rea</td>
</tr>
<tr>
<td>03</td>
<td>0304</td>
<td>O Menezes</td>
</tr>
<tr>
<td>03</td>
<td>0305</td>
<td>J Thomas</td>
</tr>
<tr>
<td>03</td>
<td>0306</td>
<td>T Nolan</td>
</tr>
</tbody>
</table>
In cell **AM3** ► Extracts a unique list of Floors switched in a single row.

I use the UNIQUE dynamic array function and I wrap it in a Transpose function.

Function arguments:

=UNIQUE(array,by_col,exactly_once)

=TRANSPOSE(array)

The function Spills to the adjacent 34 columns (since there are 35 floors).

=TRANSPOSE(UNIQUE(A3:A274))

**Function #5**

In cell **AM13** ► Extracts another unique list of Floors switched in a single row.

I use the same exact function as the previous one.

The function Spills to the adjacent 34 columns

=TRANSPOSE(UNIQUE(A3:A274))

---

**Figure 13 - Unique List of Floors**

**CREATING THE THIRD DROP LIST**

We can now create our third drop list in cell **Y3** using the available data from the previous 5 functions.

This drop list has three levels of dependency based on which option the user selects from the drop list above: Floor/ Suite/ Resident.
For each option, we want our drop list to refer to one of the three spilled arrays returned in cells: **AM13/ AH19/ Al19** respectively,

This requires combining the 3 options in a **SWITCH** function.

Function arguments:

```
=SWITCH(expression,value1,result1,default_or_value2,result2,...)
```

The SWITCH function will evaluate the value in Y2 (the first drop list) and returns one of the following:

If Y2 = Floor ► it returns the Spilled array of AM13# (list of floors)
If Y2 = Suite ► it returns the Spilled array of AH19# (list of Suites)
If Y2 = Resident ► it returns the Spilled array of Al19# (list of Residents)

As a test, we will be creating our **SWITCH** function temporarily in cell **AH13** preceded by a single apostrophe to disable it, as follows:

```
Function #6

=SWITCH($Y$2,"Floor",$AM$3#,"Suite",$AH$19#,"Resident",$AI$19#)
```

Now we can copy this function (without the single apostrophe) and paste it in the Data Validation dialog box. Let's do it

Select cell Y3 (for the third drop list) ► Open the Data validation dialog box (as we did before) ► in the Source box, paste the SWITCH function above ► Hit OK
Now I want to test the contents of the drop list in Y3 three times by changing options in the first drop list in Y2.

**SCRATCH THE LIMIT OF DATA VALIDATION IN Z4**

Our most robust function, is the one we use for our fourth drop list in cell Z4 as it switches between 6 levels of dependency and it requires a lot more doing.
The drop list we created in Y3 may display either a list of Floors, or a list Suite numbers or a list of Resident names depending on our selection from the drop list above in Y2. Wonderful…

Now we understand the concept of switching dependency in a drop list using the SWITCH function.

**LIST OF OWNED SUITES IN ANY FLOOR**

In the preparation section, in Cell AM4, we’ll create a function that extracts the Suite number of the Owned units in each floor.

**Function #7**

This function (in AM4) will filter the Suites in column B based upon a condition in the Floor (Column A) and another condition in the Type (Column D). Because I want both conditions to be met, I multiply them (that is the equivalent of the AND operator).

My function in AM4 should read:

```
=FILTER($B$3:$B$274,($A$3:$A$274=AM3)*($D$3:$D$274="Owned"))
```

AM3 is the specific Floor (cell above)

"Owned" is the specific Type we are analyzing.

The Function Spills to the cells below (only 1 direction), returning only the Suite numbers where both conditions are met.

Note that we are locking the ranges to copy in a second direction.
But, because the function cannot spill in two directions, we drag it horizontally all the way to Floor 35 (last Floor) to extract the corresponding values for each Floor.

Note: The number of returned Suites differs in each Floor.

This function (in AM14) will filter the Residents in column C based upon a condition in the Floor (Column A) and another condition in the Type (Column...
D). Because I want both conditions to be met, I multiply them (that's the equivalent of the AND operator).

My function in **AM14** should read:

```excel
=FILTER($C$3:$C$274,($A$3:$A$274=AM13)*($D$3:$D$274="Owned"))
```

AM13 is the specific Floor (cell above)

"Owned" is the specific Type we are analyzing.

The Function Spills to the cells below, returning only the Suite numbers where both conditions are met.

<table>
<thead>
<tr>
<th></th>
<th>AL</th>
<th>AM</th>
<th>AN</th>
<th>AO</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Floor</td>
<td>01</td>
<td>02</td>
<td>03</td>
</tr>
<tr>
<td>14</td>
<td>Resident</td>
<td>K Benedick</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td></td>
<td>L Leroux</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td></td>
<td>K Hassab</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td></td>
<td>U Leedham</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td></td>
<td>K Beckford</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td></td>
<td>Q Cole</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>Q Kosareva</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Figure 18* - Resident names in Owned Suites in Floor 01

Because the function cannot spill in two directions, we drag it horizontally all the way to Floor 35 (last Floor) to extract the corresponding values for each Floor. The number of returned Residents differs in each Floor, it matches the Suites returned by the previous function.
THE SIX POSSIBLE OPTIONS

The next step requires envisioning the different combinations (Six possible options) that a user can select from our first 2 drop lists in cells Y2 & Z2. These combinations are:

1. Floor-Suite
2. Floor-Resident
3. Suite-Floor
4. Suite-Resident
5. Resident-Suite
6. Resident-Floor

For each one of these combinations we will create a dynamic array function that returns the values corresponding to that combination.

Then, we will put them all in a second SWITCH function.

Finally, we use the SWICH function in our last (Fourth) drop list in cell Z3.

We will create the functions in the preparation area in cells AI6 to AI11, and park them with a single apostrophe, to have them ready to use in the powerful SWITCH function later.
So, Let's create these preparation functions. All of them look at our selection from the drop list in cell Y3.

---

**Function #9**

This function will deal with combination # 1: **Floor - Suite**

It will extract the list of Suites for any selected Floor using our preparation functions starting in AM3# Down and to the Right.

This Function will look at the specific Floor in Y3 then pulls the entire list of Suites from the preparation area.

This will be an **XLOOKUP** function that requires 3 arguments:

- **Lookup Value** ► Y3 (the third drop list)
- **Lookup Array** ► The Spilled Array of Floors AM3#
- **Return Array** ► The unequal number of Suites Spilled below the corresponding Floor in the previous Lookup Array. Because they are **unequal** then we will use for this argument an **OFFSET** function.

Function arguments:

=**XLOOKUP**(lookup_value,lookup_array,return_array,if_not_found,match_mode,search_mode)

=**OFFSET**(reference,rows,cols,height,width)

Our Function that we will create for now in A16 will read:

=**XLOOKUP**(Y3,AM3#,OFFSET(AM3#,1,0))

---

*Figure 20 - One Suite returned*
This Function when tested returns **ONLY** the first Suite number for the selected Floor in Y3. To extracts all the Spilled list of Suites, I add another # sign at the end. My function looks like that:

```excel
=XLOOKUP(Y3,AM3#,OFFSET(AM3#,1,0))#
```

Let’s park this function by typing a single apostrophe before the = sign.

**Function #10**

This function will deal with combination # 2: **Floor - Resident**

We will temporarily create it in **A17**.

It will extract the list of Residents for any selected Floor using our preparation functions starting in AM13# **Down and to the Right**.

This Function will look at the specific Floor in Y3 then pulls the entire list of Residents from the preparation area.

We could use a similar XLOOKUP function like the one above, but let's learn another technique.

This will be an **OFFSET** function that requires 5 arguments, the first 3 describe a starting point and the next 2 describe the Height and Width of the range to return

Reference ► AM13

Rows ► 1 (We want to move the start point 1 row down)
Columns ► To move the starting point to the column corresponding to the Floor in Y3. So we use a Match function for this argument. When added to the starting cell in AM13 >> It takes me to the next Floor, so I subtract 1.

My Function that I will create for now in A17 will read:

\[
=\text{OFFSET}(\text{AM13},1,\text{MATCH(Y3,AM13\#,0)}-1)
\]

It returns ONLY the name of the first resident in the list of residents for the selected Floor.

To return the entire list of residents, I add another spilled array symbol after the closing bracket of the OFFSET function.

\[
=\text{OFFSET}(\text{AM13},1,\text{MATCH(Y3,AM13\#,0)}-1)\#
\]

Let’s park this function in A17 by typing a single apostrophe before the = sign.

Function #11
This function will deal with combination # 3: **Suite - Floor**

We will temporarily create this function in **AI8**

It's a simple XLOOKUP function, that looks at cell Y3 (The drop list), finds a match in the spilled array range AH19#, and returns a corresponding value from the Spilled array in AG19# (in the preparation area). The function reads:

\[=XLOOKUP(Y3,AH19#,AG19#)\]

---

**Figure 24- Floor number of a selected Suite**

Let's park this function in **AI8** by typing a single apostrophe before the = sign.

**Function #12**

This function will deal with combination # 4: **Suite - Resident**

We will temporarily create this function in **AI9**

It's a simple XLOOKUP function, that looks at cell Y3 (The drop list), finds a match in the spilled array range AH19#, and returns a corresponding value from the Spilled array in AI19# (in the preparation area). The function reads:

\[=XLOOKUP(Y3,AH19#,AI19#)\]
Let's park this function in **AI8** by typing a single apostrophe before the = sign.

**Function #13**

This function will deal with combination # 5: **Resident - Suite**

We will temporarily create this function in **AI10** (similar to previous)

It's a simple XLOOKUP function, that looks at cell Y3 (The drop list), finds a match in the spilled array range AI19#, and returns a corresponding value from the Spilled array in AH19# (in the preparation area). The function reads:

```
=XLOOKUP(Y3, AI19#, AH19#)
```

**Function #14**

This function will deal with combination # 6: **Resident - Floor**

We will temporarily create this function in **AI11** (similar to previous)
It's a simple XLOOKUP function, that looks at cell Y3 (The drop list), finds a match in the spilled array range AI19#, and returns a corresponding value from the Spilled array in AG19# (in the preparation area). The function reads:

\[ = \text{XLOOKUP}(Y3, \text{AI19#}, \text{AG19#}) \]

Here are the 6 preparation functions needed for the fourth drop list. To show all of them I split the window and they are all temporarily parked in column AI from row 6 to 11. We will be later combining them together in a SWITCH function inside the data Validation dialog box.

Problem with the SWITCH function in the Data Validation box:

The SWITCH function is intended to return one of SIX different ranges for the drop list in Z3 based upon the 6 possible combinations in Y2 & Z2. So it should originally read like this:

\[ = \text{SWITCH}(Y2\&"-"\&Z2,"Floor-Suite","Floor-Resident", \text{XLOOKUP}(Y3,\text{AM3#},\text{OFFSET(AM3#,1,0)#}),"Suite-Floor", \text{XLOOKUP}(Y3,\text{AM13#,MATCH(Y3,\text{AM13#,0)-1)#}),"Suite-Resident", \text{XLOOKUP}(Y3,\text{AH19#,AG19#}),"Resident-Suite", \text{XLOOKUP}(Y3,\text{AI19#,AH19#}),"Resident-Floor", \text{XLOOKUP}(Y3,\text{AI19#,AG19#})) \]

The LEN (length) of this function is **281 characters**, while the maximum limit of any function in the Data Validation dialog box is **255 Characters**. So we need to create a shorter form of the SWITCH function to be able to use it in a drop list.

Nabil Mourad
PREPARING FOR THE SWITCH FUNCTION:
We abbreviated in Column AH from AH6 to AH11 all the six possible combinations from the drop lists Y2 & Z2, as follows:

<table>
<thead>
<tr>
<th>Floor-Suite</th>
<th>FS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Floor-Resident</td>
<td>FR</td>
</tr>
<tr>
<td>Suite-Floor</td>
<td>SF</td>
</tr>
<tr>
<td>Suite-Resident</td>
<td>SR</td>
</tr>
<tr>
<td>Resident-Suite</td>
<td>RS</td>
</tr>
<tr>
<td>Resident-Floor</td>
<td>RF</td>
</tr>
</tbody>
</table>

In cell AA1, I create a VLOOKUP function that returns the abbreviation based upon the selection from Both Y2 & Z2.

Function arguments:

=VLOOKUP(lookup_value,table_array,col_index_num,range_lookup)

The VLOOKUP function reads:

=VLOOKUP(Y2"-"&Z2,AG6:AH11,2,0)

Now we can rebuild our SWITCH function using the abbreviation returned in AA1 Rather than the long names in Y2 & Z2.

We will temporarily build our SWITCH function in cell AH14 and test it (►#Spill! Error because we have cell contents below), measure it's length in AH15 ►now it becomes 209 character instead of the long format of 281
characters. Park it with a single apostrophe at the beginning. The function in AH14 reads:

```
=SWITCH(AA1,"FS",XLOOKUP(Y3,AM3#,OFFSET(AM3#,1,0))#, "FR",OFFSET(AM13,1,MATCH(Y3,AM13#,0)-1)#, "SF",XLOOKUP(Y3,AH19#,AG19#), "SR",XLOOKUP(Y3,AH19#,AI19#), "RS",XLOOKUP(Y3,AI19#,AH19#), "RF",XLOOKUP(Y3,Al19#,AG19#))
```

We are ready to create our last drop list in cell Z3 using the shortened SWITCH function (now 209 characters)

Select cell Z3

Open the Data Validation dialog box ► Select "List"

In the source box, paste the new SWITCH function.

Make a selection from Y2, Z2 and Y3 ► then Test the 6 options for Z3

We just touched the limit of what data validation can do with 6 levels of dependencies. Let’s explore them:

**Option #1**

Floor-Suite ► FS

```
=SWITCH(AA1,"FS",XLOOKUP(Y3,AM3#,OFFSET(AM3#,1,0))#, "FR",OFFSET(AM13,1,MATCH(Y3,AM13#,0)-1)#, "SF",XLOOKUP(Y3,AH19#,AG19#), "SR",XLOOKUP(Y3,AH19#,AI19#), "RS",XLOOKUP(Y3,AI19#,AH19#), "RF",XLOOKUP(Y3,Al19#,AG19#))
```
Option #2

Floor-Resident► FR

=SWITCH(AA1,"FS",XLOOKUP(Y3,AM3#,OFFSET(AM3#,1,0))#,"FR",OFFSET(AM13,1,MATCH(Y3,AM13#,0)-1)#,"SF",XLOOKUP(Y3,AH19#,AG19#),"SR",XLOOKUP(Y3,AH19#,AI19#),"RS","RF",XLOOKUP(Y3,AI19#,AH19#))
"SF",XLOOKUP(Y3,AH19#,AG19#),"SR",XLOOKUP(Y3,AH19#,AI19#),"RS",XLOOKUP(Y3,Al19#,AH19#),"RF",XLOOKUP(Y3,Al19#,AG19#))

**Figure 30 - Option 3 Suite-Floor**

**Option #4**

Suite-Resident ► SR

=SWITCH(AA1, "FS", XLOOKUP(Y3,AM3#,OFFSET(AM3#,1,0))#, "FR",OFFSET(AM13,1,MATCH(Y3,AM13#,0)-1)#, "SF",XLOOKUP(Y3,AH19#,AG19#),"SR",XLOOKUP(Y3,AH19#,AI19#),"RS",XLOOKUP(Y3,Al19#,AH19#),"RF",XLOOKUP(Y3,Al19#,AG19#))

**Figure 31 - Option 4 Suite-Resident**

**Option #5**

Resident-Suite ► RS

=SWITCH(AA1, "FS", XLOOKUP(Y3,AM3#,OFFSET(AM3#,1,0))#, "FR",OFFSET(AM13,1,MATCH(Y3,AM13#,0)-1)#, "SF",XLOOKUP(Y3,AH19#,AG19#),"SR",XLOOKUP(Y3,AH19#,AI19#),"RS",XLOOKUP(Y3,Al19#,AH19#),"RF",XLOOKUP(Y3,Al19#,AG19#))
Option #6

Resident-Floor ► RF

=SWITCH(AA1,"FS",XLOOKUP(Y3,AM3#,OFFSET(AM3#,1,0))#,"FR",OFFSET(AM13,1,MATCH(Y3,AM13#,0)-1)#, "SF",XLOOKUP(Y3,AH19#,AG19#),"SR",XLOOKUP(Y3,AH19#,AI19#),"RS ",XLOOKUP(Y3,AL19#,AH19#),"RF",XLOOKUP(Y3,AL19#,AG19#))

Hide contents of Cell AA1
Since Cell AA1 is just providing a Value for our SWITCH function we want to hide it from the view by changing the Font color to White.

Resetting 2 drop lists (Y3 & Z3) with a code
Every time we make a change in any drop list in Y2 or Z2, we want the previously selected value in Y3 & Z3 to be deleted in order to make a new fresh selection in these cells. This requires creating a "Change Event" code for the worksheet as follows:

Open the VBE
Right click the sheet Tab and select ► View Code
The Visual Basic Editor opens. There are 2 drop lists at the top. Click on the Left drop List and select ► Worksheet

Then Click on the Right drop list and select ►Change

Keep only the Worksheet Change event and delete the Worksheet Selection Change event

Figure 34- Switching to VBE

Figure 35- Select the Change Event

Figure 36-Delete the Selection Change Event
**Writing the Code**

Between the Private Sub and the End Sub lines, copy and paste this code:

```
If Not Intersect(Target, Range("Y2:Z2")) Is Nothing Then
    Range("Y3:Z3").ClearContents
End If
```

**This code means:**

If you change your selection from either drop lists in Y2 or Z2 ► Clear the contents of Y3 and Z3

![Figure 37 - The Change event code](image)

Close the Visual Basic editor to go back to Excel and test. Although the values in Y3 and Z3 are deleted with any change in Y2 or Z2, however the drop lists in Y3 and Z3 are fully functional.

**Note:** Your Excel file must be stored in a .XLSM format.

**Getting Information based upon selection**

Our next step will be extracting the corresponding: Total Paid, Total Due, Past Due and Over Payment for whichever option we select (6 combinations) from Y3 and Z3.

**Note:** we will use nested VLOOKUP functions to do that

Function arguments:

=VLOOKUP(lookup_value,table_array,\*col_index_num*,range_lookup)

We will be creating 2 VLOOKUP functions.

Nabil Mourad
To avoid confusion, you can create each VLOOKUP function separately and copy it to the Clipboard.

Also Note the Column Numbers from which we will extract the return values (Column Index Number argument of the VLOOKUP)

To show the column numbers instead of the default column letters:

File ► Options ► Formulas ► check R1C1 reference

To make it easier for you, we will hide columns 5 to 19 since we are not extracting values from any of these columns and we will bring back the column letters (File ► Options ► Formulas ► uncheck R1C1 reference)
FIRST VLOOKUP

Our first VLOOKUP function will extract values by Suite number:

- The Lookup Column is column B
- The Lookup Array is: B3:W274
- The column index number will be 19, 20, 21, 22

Since I create a single function (in cell AA3 under Total Paid) and drag it to the right, I want the column index number to increment as I drag the function, I do that by using a COLUMNS function that refers to any cell twice, lock the first reference only, then add 18.

Note in this array starting from Column B ► The Total Paid is column 19.

So, the Column Index argument of this VLOOKUP will look like this:

\[
\text{COLUMNS($X$1:X1)+18}
\]

And the VLOOKUP function will look like this:

\[
=\text{VLOOKUP($Z$3,$B$3:$W$274,COLUMNS($X$1:X1)+18,0)}
\]
What if the user searches by Resident name instead of by Suite? Then, the VLOOKUP function is different:

**SECOND VLOOKUP**

Our second VLOOKUP function will extract values by Resident:

- The Lookup Column is column C (Resident)
- The Lookup Array is: C3:W274
- The column index number will be 18, 19, 20, 21

Since I create a single function (in cell AA3 under Total Paid) and drag it to the right, I want the column index number to increment as I drag the function. I do that by using a COLUMNS function that refers to any cell twice, lock the first reference only, then add 17.

Note in this array starting from Column C ► The Total Paid is column 18

So, the Column Index argument of this VLOOKUP will look like this:

```
COLUMNS($X$1:X1)+17
```

And the VLOOKUP function will look like this:

```
=VLOOKUP($Z$3,$C$3:$W$274,COLUMNS($X$1:X1)+17,0)
```

To avoid confusion, you can create each VLOOKUP function separately and copy it to the Clipboard.
COMBINING THE 2 VLOOKUP FUNCTIONS

Now I want to combine both functions together to include both options (whether the user searches by Suite or by Resident) so I will wrap them together in an **IFERROR** function, pasting them from the clipboard.

My combined function will look like this:

```
=IFERROR(VLOOKUP($Z$3,$B$3:$W$274,\[COLUMNS($X$1:X1)+18,0),VLOOKUP($Z$3,$C$3:$W$274,\[COLUMNS($X$1:X1)+17,0))
```

A LITTLE ISSUE TO AVOID

Finally, If the user selects either a Suite or Resident in Y3 and a **Floor** in Z3 ► Our VLOOKUP function will return a **#N/A error**. Because Z3 is always our lookup value, we do not want to search by Floor in this cell. But if it happens, we also do not want to see the #N/A error.

A SECOND IFERROR FUNCTION

To do that we will wrap the previous function in a second **IFERROR** function that returns a blank.

Our final function should look like this:

```
=IFERROR(IFERROR(VLOOKUP($Z$3,$B$3:$W$274,\[COLUMNS($X$1:X1)+18,0),VLOOKUP($Z$3,$C$3:$W$274,\[COLUMNS($X$1:X1)+17,0)),""
```

AN INPUT MESSAGE

To make it practical, add an input message to the drop list in cell Z3 saying:

*Select either a Suite number or a Resident name*

Copy the Final function in AA3 to the right to AD3
Finalizing the project

**TWO CONDITIONAL FORMATTING RULES**

To clearly spot any amount past due or any overpayment, we will be applying Conditional Formatting for the Amount *Past Due* and *Over PMT*

Select Cell AC3

Home Tab ► Conditional Formatting ► Greater Than.. ► Type 0 and keep the red color

Select Cell AD3

Home Tab ► Conditional Formatting ► Greater Than.. ► Type 0 and change the color to Green
Test again by selecting different options in Z3

Our project is finished!

But we have a lot of numbers and calculations (in the preparation area) so we want to make our sheet much more user friendly.

**HIDING COLUMNS**
Select then► Hide Columns from D to S
Select then► Hide Columns from AG to BT

**ZOOM IN AND OUT**
Insert 2 icons (Insert Tab ► Icons ► Search for "Zoom" ► select the 2 magnifying lenses

Resize and position each icon to fit on top of cells AA1 and AB1. Change the color however you like (Graphics Format ► Graphics Fill). Flip one of them Horizontally

They will look like this:
These 2 icons will be used to **Zoom In** and **Zoom Out** respectively on the range from Y1 to AD3, for now they are not functioning.

**CREATE 2 MACROS FOR THE ZOOMING FUNCTIONALITY**

Switch to the Visual Basic editor by hitting **ALT + F11**

Insert a Module (Blank Space): Insert Menu ► Module

```
Note: You can use the shortcut **ALT > I > M**
```

Click in the white space

Copy and paste these two codes

```
Sub RangeZoom()
    Sheets("Start").Range("Y1:AE3").Select
    ActiveWindow.Zoom = True
    Range("Y1").Select
End Sub

Sub UnZoom()
    Sheets("Start").Range("Y1:AE3").Select
    ActiveWindow.Zoom = False
    Range("Y1").Select
End Sub
```

Close the visual Basic Editor ► Back to Excel

**ASSIGNING MACROS**

Select the magnifying icon for zooming in (plus sign) ► Right click it ► Assign macro.
The Assign Macro dialog box opens ► select **RangeZoom** macro ► Hit OK.

Every time you click on this icon ► the range Y1: AE3 will fill the entire screen.
Select the magnifying icon for zooming Out (minus sign)►Right click it ► Assign macro.

The Assign Macro dialog box opens ► select **UnZoom** macro ► Hit OK.
Push it to the limit... and beyond

Every time you click on this icon► the range Y1: AE3 will zoom out to 100%
Test Zooming in and Out.
Test all your drop lists with different combinations
Test the extraction of records and conditional formatting.
Congratulations the project is now complete! 😊