**{LINQ – UIPath**

**LINQ - Language Integrated Query**

*Note: This Event will be useful in learning LINQ queries and what are the places we can apply LINQ in UiPath.*

*--------------------------------------------------------------------------------------------------------------------------*

LINQ was introduced in .NET Frame works, VB.NET and C#. Linq is a powerful concept where we can execute the process in single activity instead of using multiple activities.

LINQ is commonly used in Array, List , Data table , Dictionary and any Data collection.

***Advantages of LINQ***

* LINQ allows debugging which can be useful while troubleshooting.
* Most of the LINQ queries are reusable.
* LINQ provides powerful filtering, ordering, and grouping capabilities with minimum application code.
* Code Readability will be easy while transferring the knowledge
* Decrease the number of activities. Decrease the testing time, where execution time will be much faster compared to other loops.
* This helps you to transform ANY source of data in ANY format of collection
* There is no specific activities required for using LINQ

***Disadvantages of LINQ***

* Exact issue cannot be identified
* Any complex query if revisited after few month or days, user may find difficulties in understanding the logics

**Types of Syntax**

We have two type of syntax used in LINQ;

1. Method Syntax
2. Query Syntax

**Method Syntax:**

Method syntax is predefined methods uses extension to query data from data source

**Ex:** *arr\_name.Select(Function(Item) item\*item).ToArray*

**Query Syntax:**

Query syntax uses query expression to retrieve the data from data source

**Ex:** *(From item in arr\_name Select item\*item).ToArray*

**Used Linq Methods:**

Some of the most used LINQ methods in UIPath are as below :

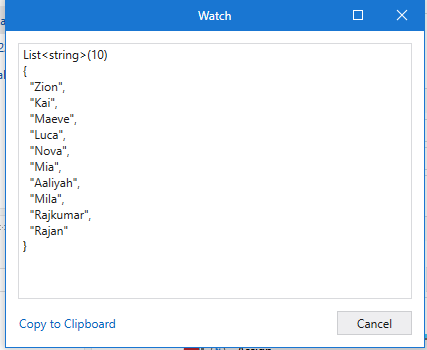
1. Select
2. Where
3. Any
4. All
5. Groupby
6. Orderby
7. Join
8. Take,Takewhile
9. Skip,SkipWhile
10. First
11. Except,Intersect, Distinct
12. Aggregate Function(Sum,Maximum,Minimum

**Input Datatable**

|  |  |  |
| --- | --- | --- |
| Roll No | Name | Class |
| 1 | Zion | 2 |
| 2 | Kai | 2 |
| 3 | Maeve | 4 |
| 4 | Luca | 4 |
| 5 | Nova | 5 |
| 6 | Mia | 5 |
| 7 | Aaliyah | 8 |
| 8 | Mila | 8 |
| 9 | Rajkumar | 9 |
| 10 | Rajan | 10 |

**Select:**

DT\_input.AsEnumerable().Select(Function(DR) DR("Name").ToString).ToList

****

**Where:**

DT\_input.AsEnumerable().Where(Function(DR) CInt(DR("Class").ToString) > 5).CopyToDataTable

**Graphical user interface, application, Word

Description automatically generated**

**All**

**Input**

**Graphical user interface, application, Word

Description automatically generated**

DT\_input.AsEnumerable().Where(Function(DR)Not DR.ItemArray.All(Function(Field) Field Is DBNull.Value or Field.Equals(""))).CopyToDataTable

**Graphical user interface, text, application, Word

Description automatically generated**

**Groupby:**

(From row In DT\_input.AsEnumerable()

Group row By col3 = row(2).tostring Into grp= Group

Let RollNo = String.Join(",", grp.Select(Function(DR) DR("Roll No").tostring).toArray)

Let Name = String.Join(",", grp.Select(Function(DR) DR("Name").tostring).toArray)

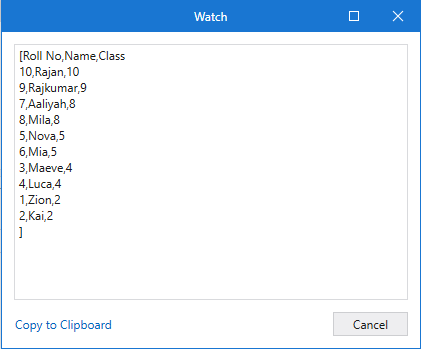
Let Std = col3

Select DT\_Skip.Rows.Add({RollNo,Name,Std})).copytodatatable

Graphical user interface, text, application

Description automatically generated

**Orderby:**

DT\_input.AsEnumerable().OrderByDescending(Function(DR) Cint(DR("Class").ToString)).CopyToDataTable()****

**Take:**

DT\_input.AsEnumerable().Take(5).CopyToDataTable

**Graphical user interface, application, Word

Description automatically generated**

**Skip:**

DT\_input.AsEnumerable().Skip(5).CopyToDataTable

**Graphical user interface, text, application, Word

Description automatically generated**

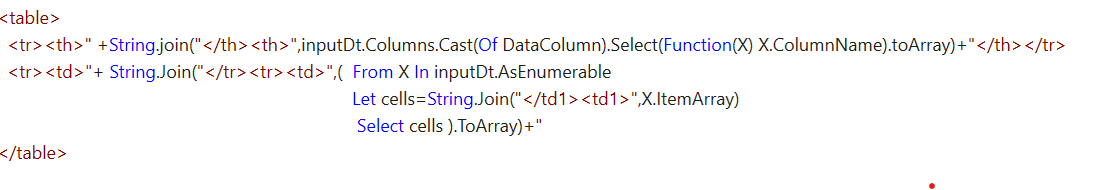
**Tutorial:**

<https://www.tutorialsteacher.com/linq>

**Create Html Code From DataTable By Using Linq:**

<table><tr><th>" +String.join("</th><th>",inputDt.Columns.Cast(OfDataColumn).Select(Function(X) X.ColumnName).toArray)+"</th></tr><tr><td>"+String.Join("</tr><tr><td>",( From X In inputDt.AsEnumerable

Let cells=String.Join("</td1><td1>",X.ItemArray) Select cells ).ToArray)+"</table>



**TO Remove the empty cell from table**

**dt = dt.AsEnumerable.Select(Function(r,i) dt.Clone.LoadDataRow({dt.AsEnumerable.Where(Function(r2,i2) i2<=i andAlso not String.IsNullOrEmpty(r2(0).ToString)).Last.Item(0).Tostring,r(1)},False)).Where(Function(r) not String.IsNullOrEmpty(r(1).ToString)).CopyToDataTable()**

**Input: output:**

|  |  |
| --- | --- |
| column1 | column2 |
| a |  |
|  | b |
| c |  |
|  | d |
| e |  |
|  | f |

|  |  |
| --- | --- |
| column1 | column2 |
| a | b |
| c | d |
| e | f |

**Pivot DataTable:**

(

From row In dt\_PivotTable.AsEnumerable

Group row By key = New With{

key.Region=row.item("Region"),

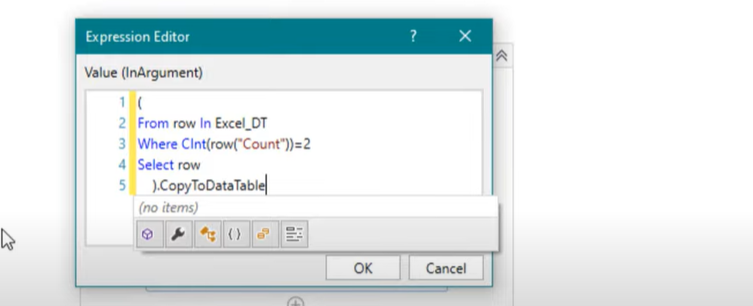
key.Rep=row.item("Rep"),

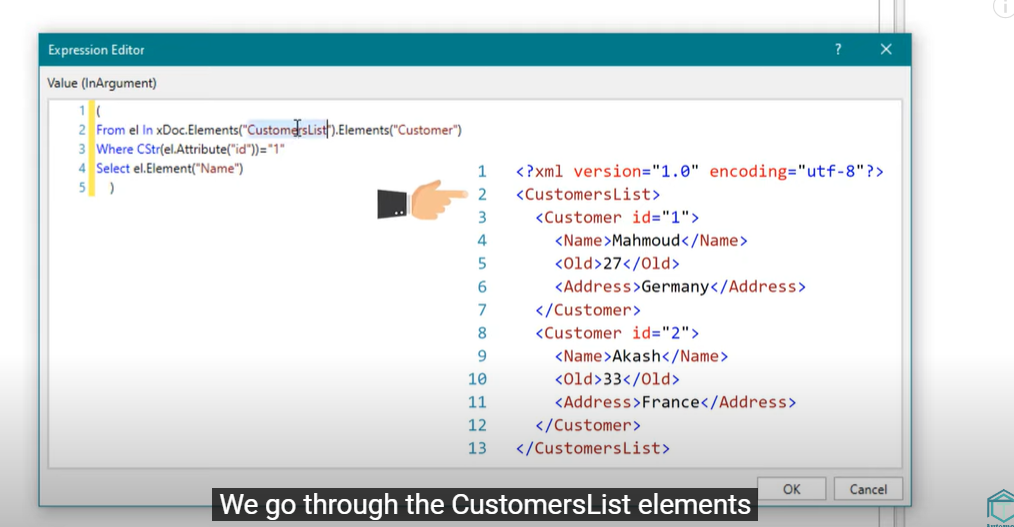
key.Item=row.item("Item")

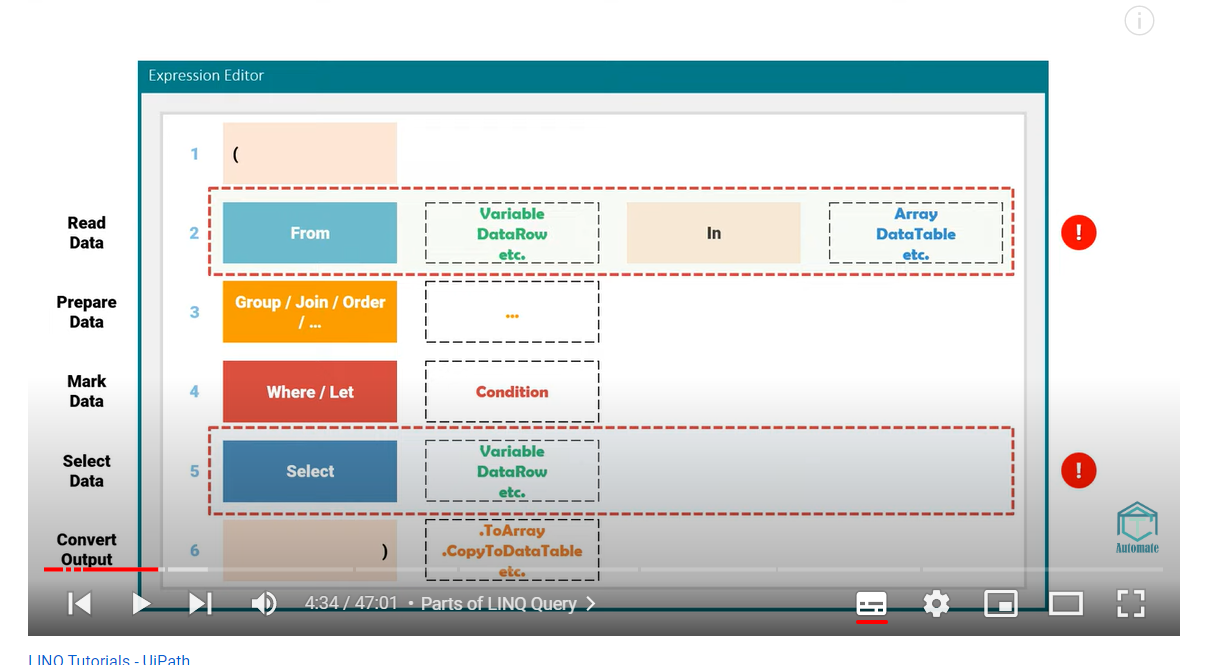
}Into grp=Group

Select dt\_BuildDT.LoadDataRow(New Object(){

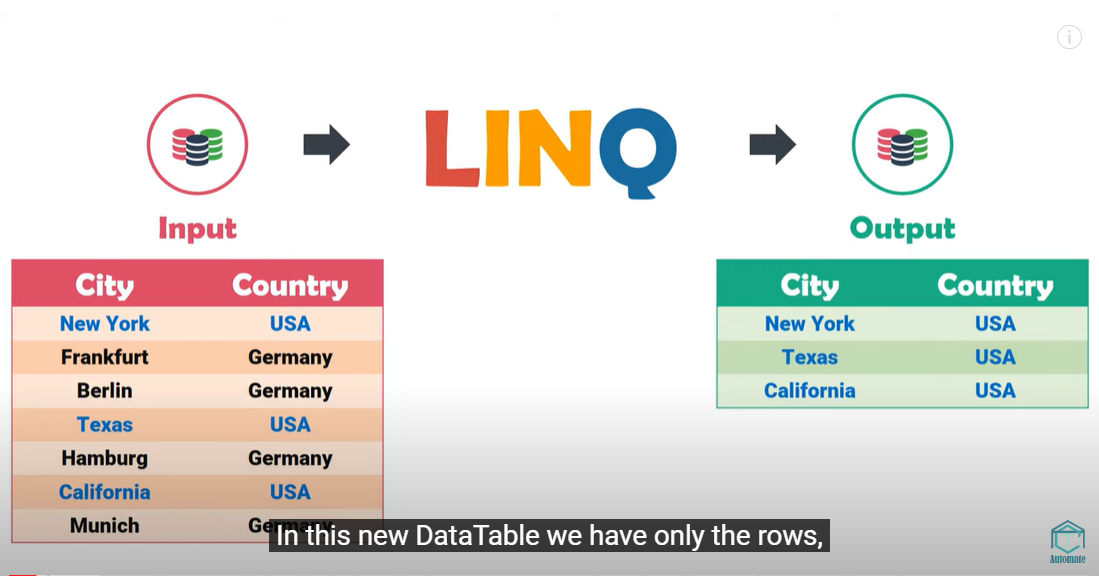
key.Region,key.Rep,key.Item,grp.count()},True)).copyTodataTable()

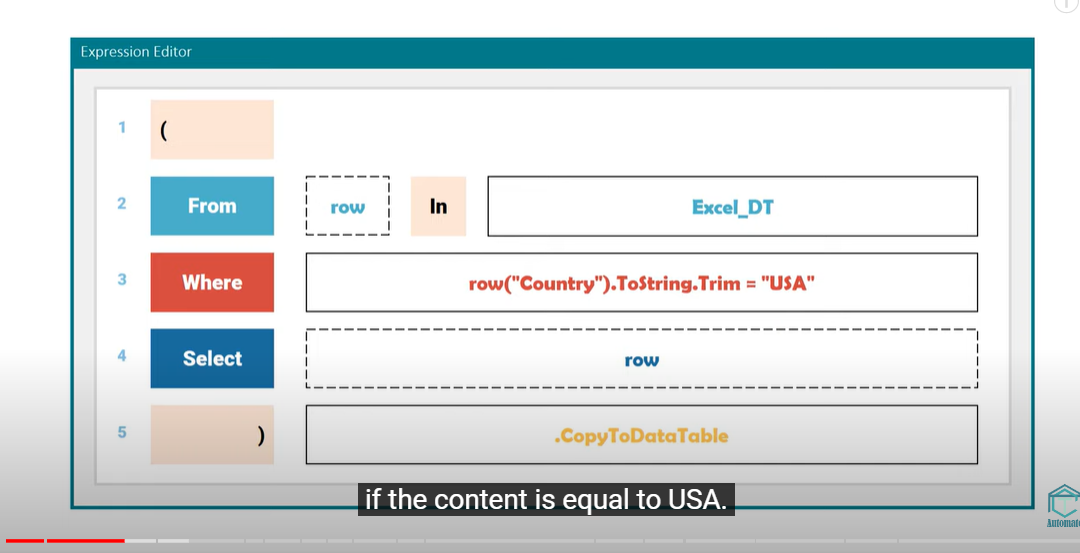


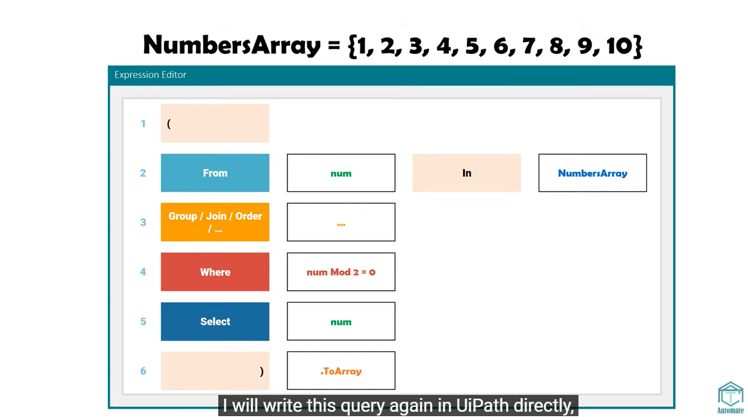


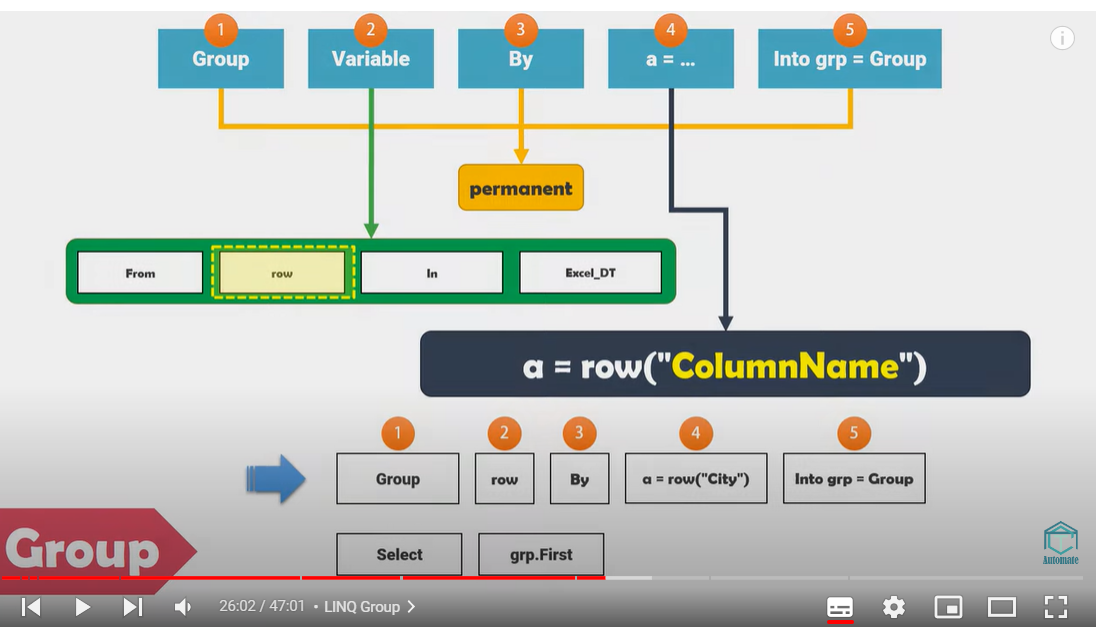


**FILTER DATA TABLE:**

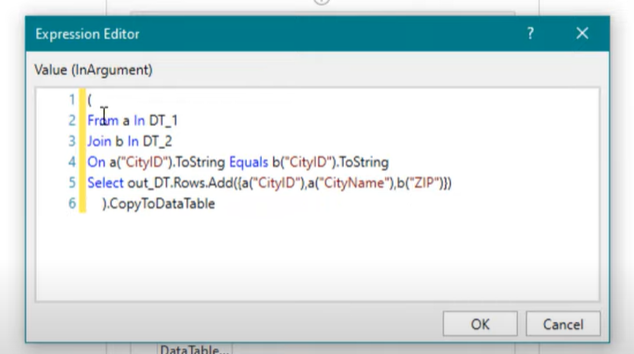
****

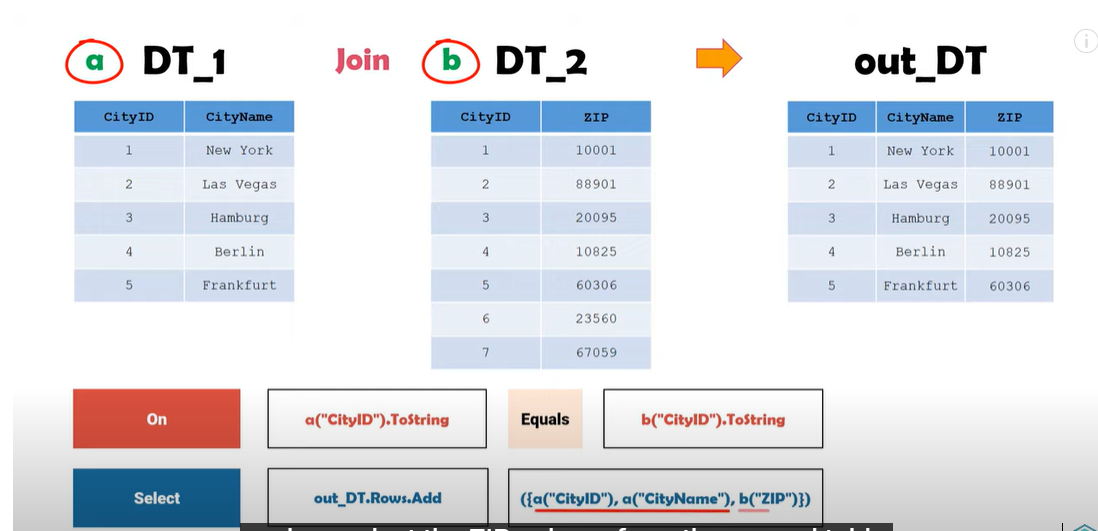


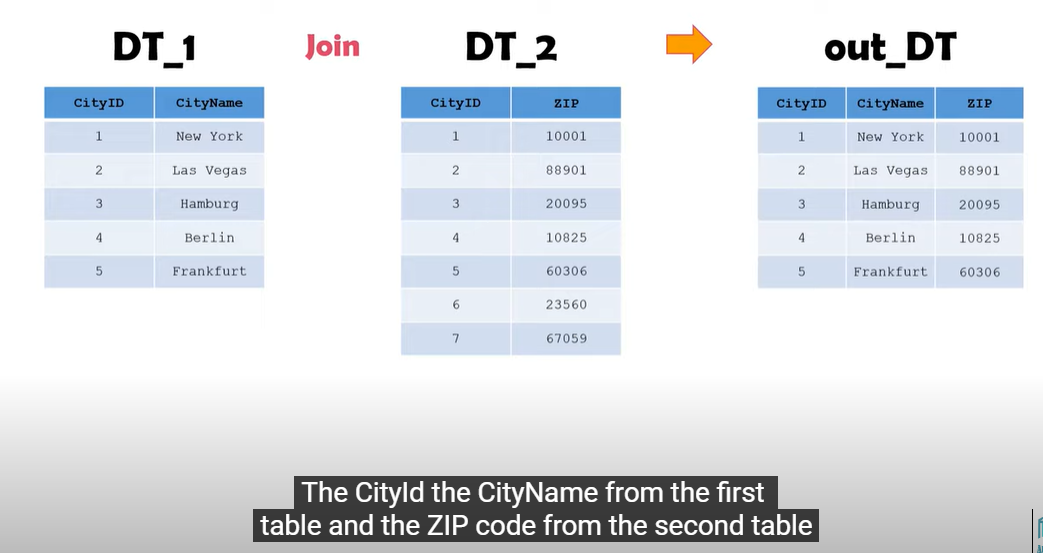






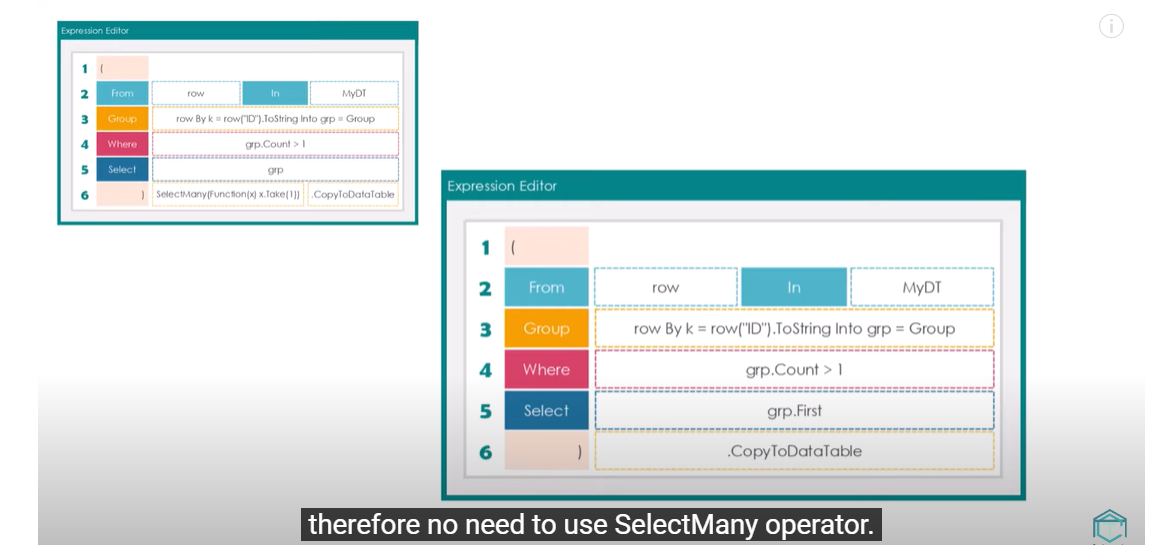


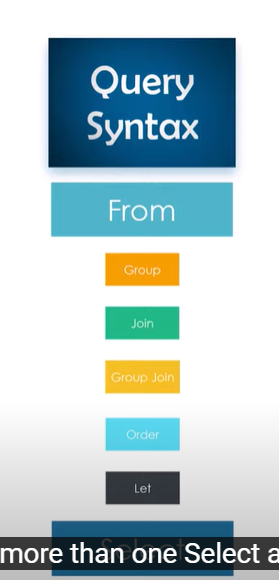




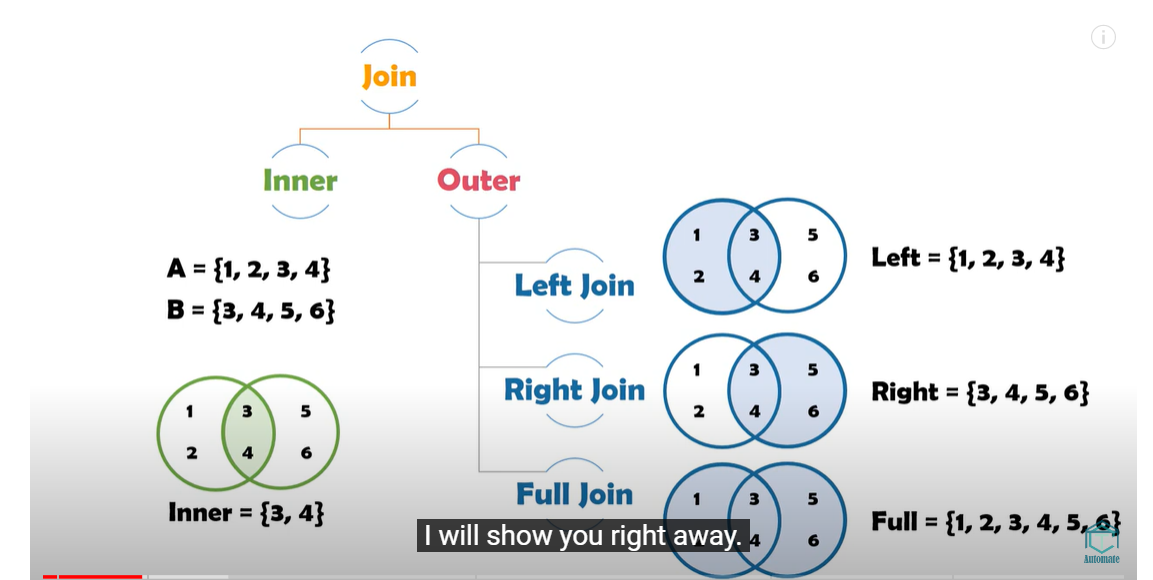


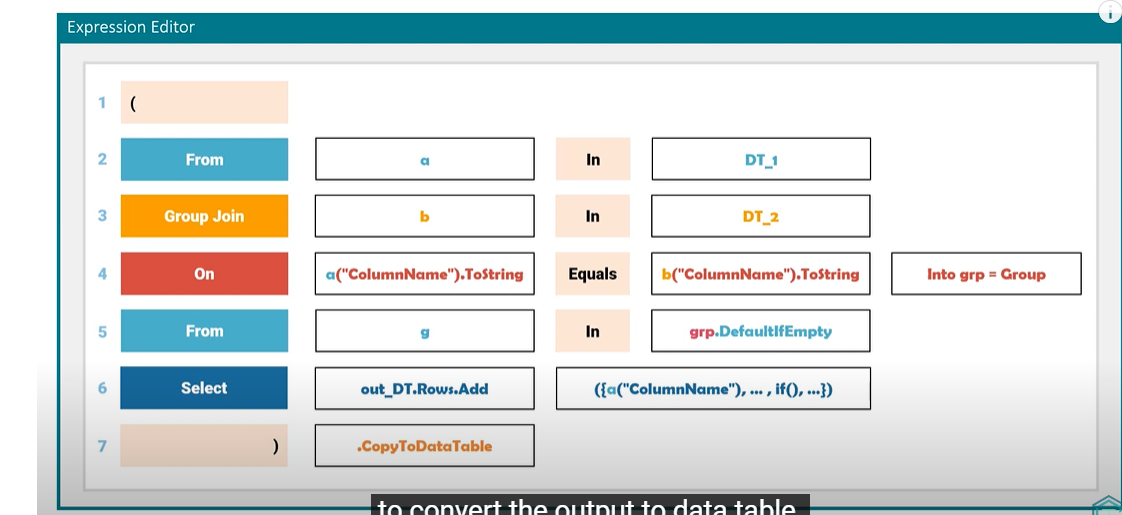
  
both query give same result :

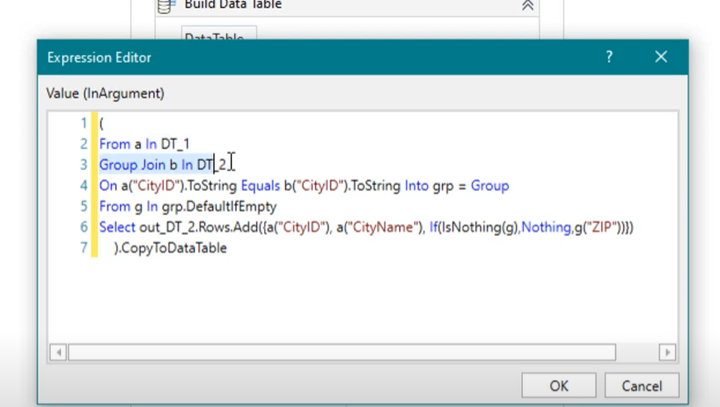


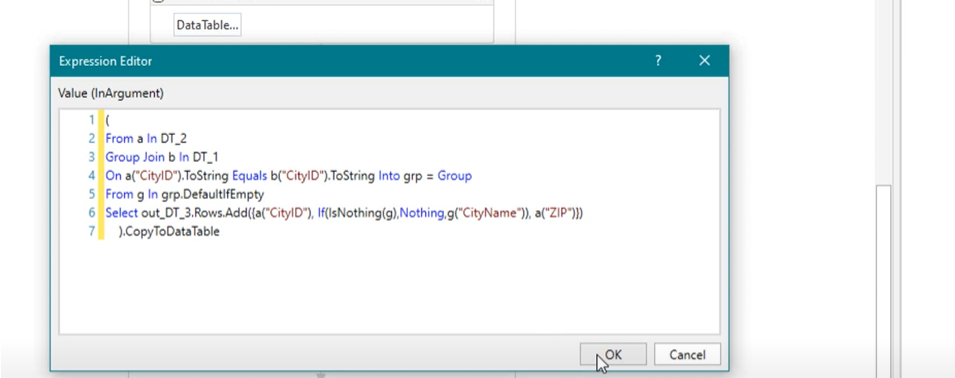


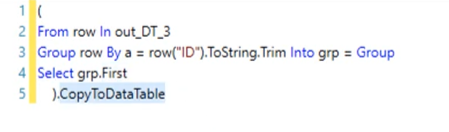
**JOIN:**

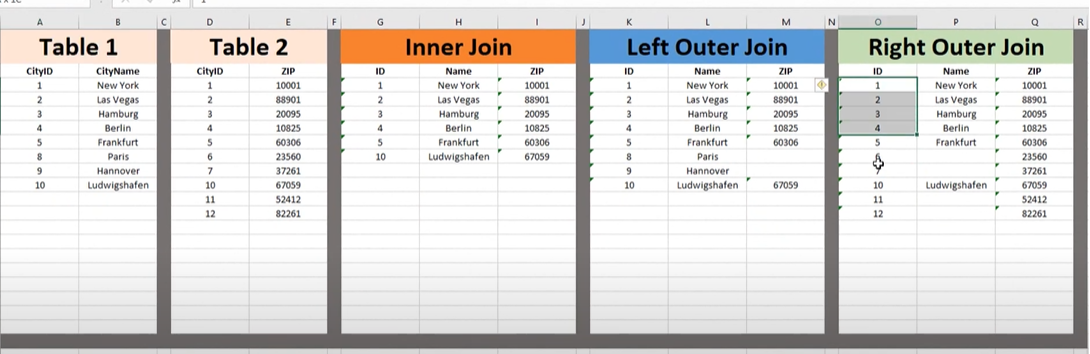
****

****

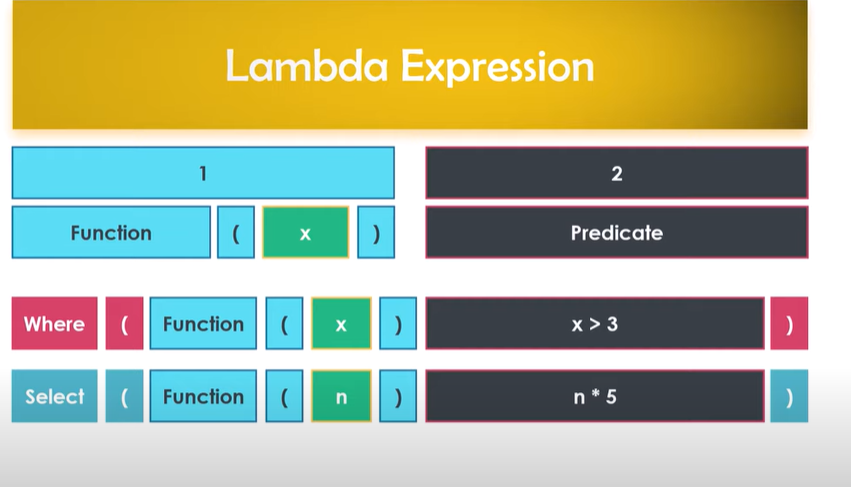
****

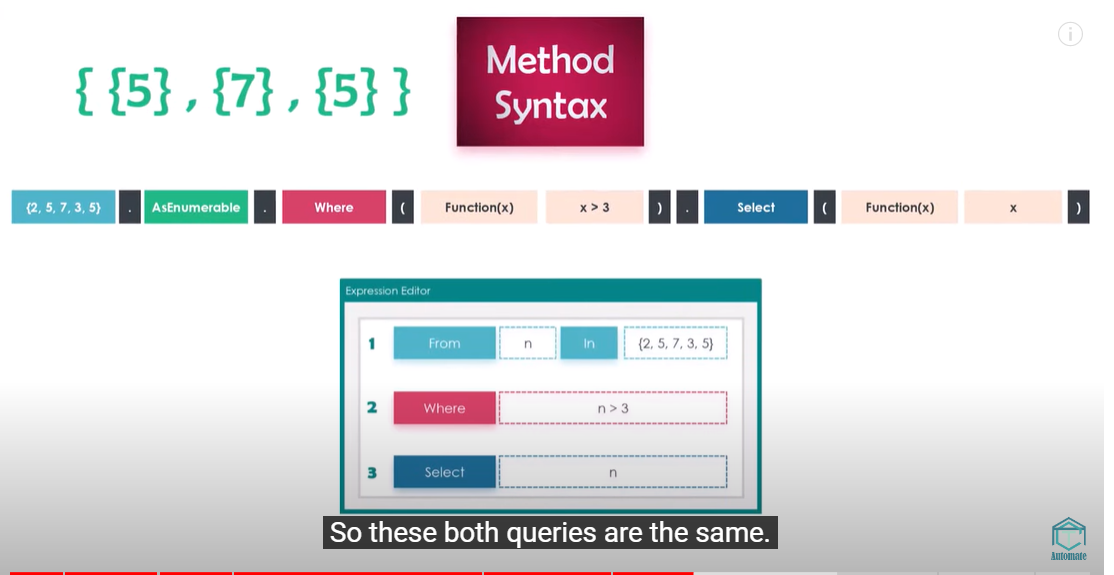
****

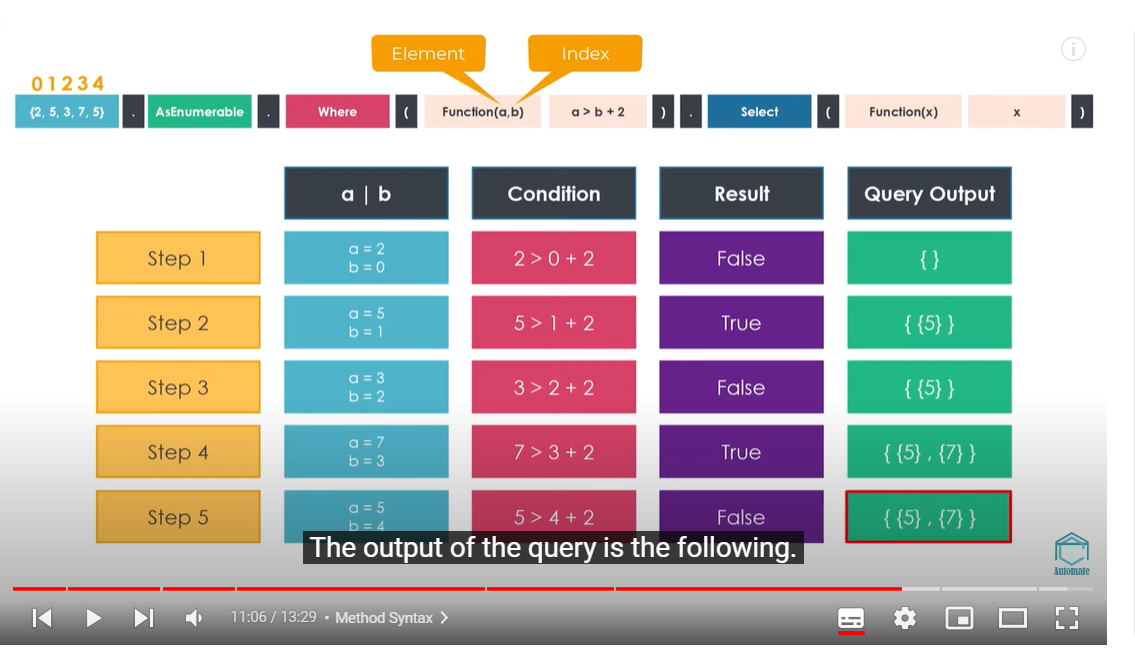
****

**Full Join :Merge data table**

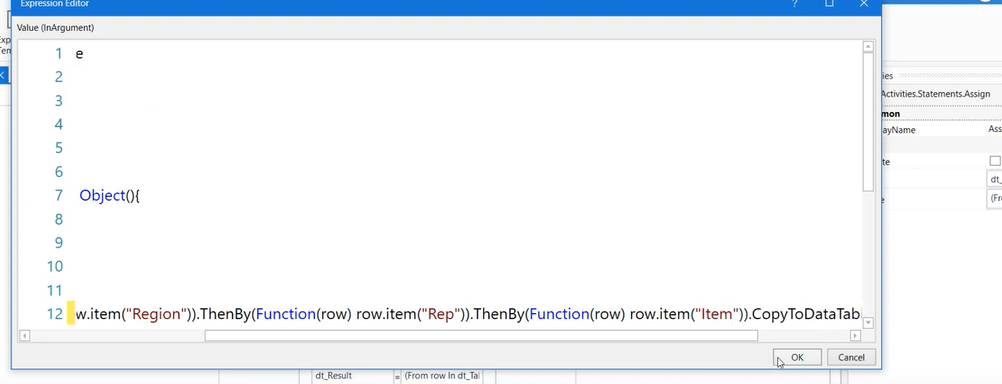
**To REMOVE DUPLICSATES ROWS:**

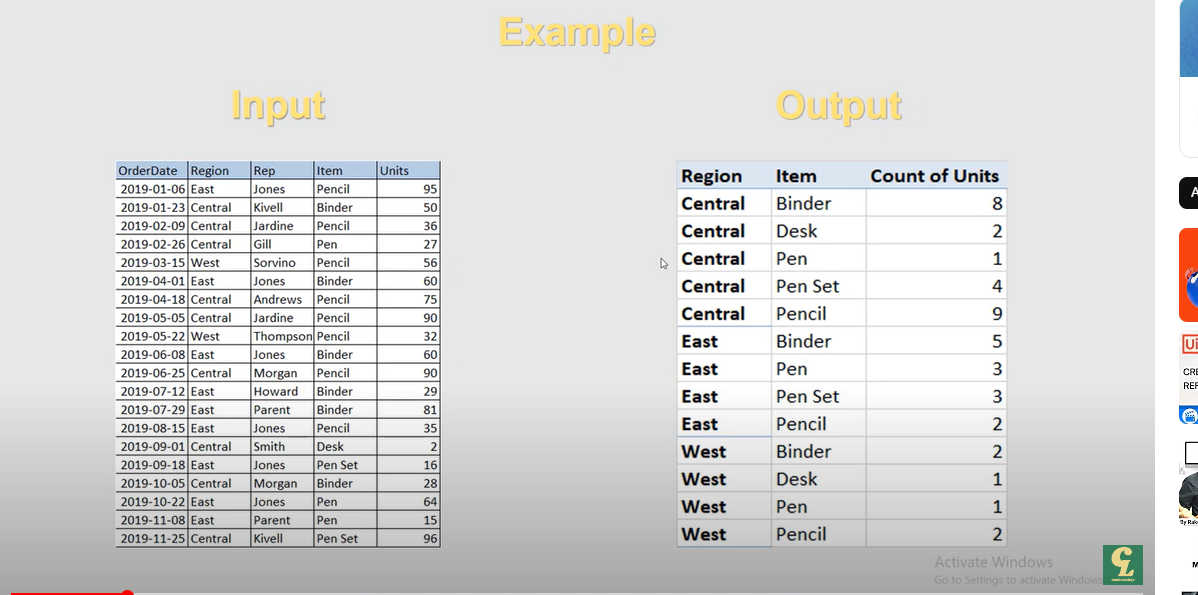






**Pivot data table:** ****



****

**VB CODE FOR EACH:**

**Sub froeachmodel()**

**For Each Number In Range("A2:A9")**

**If Number > 50 Then**

**Number.Offset(0, 2).Value = "pass"**

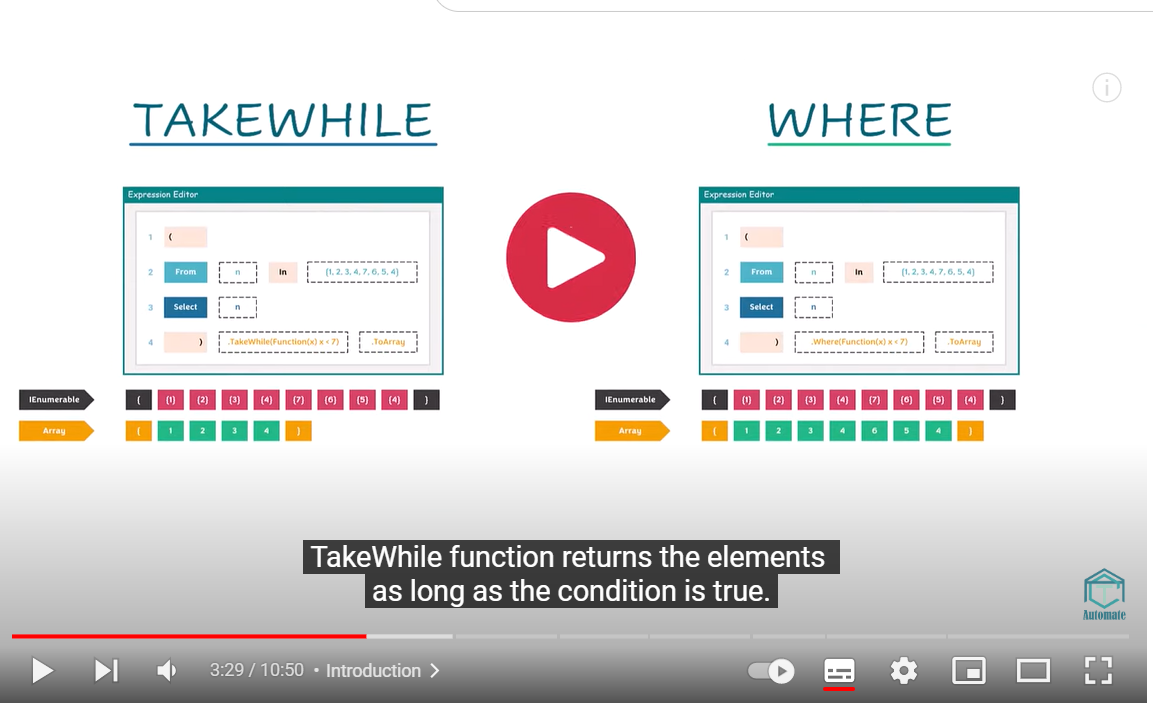
**Else**

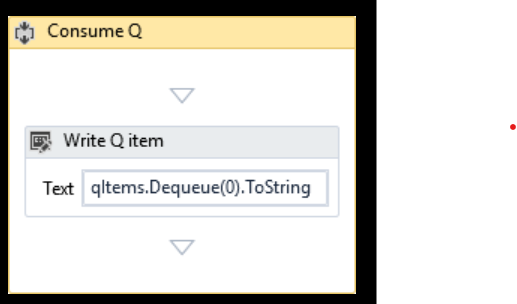
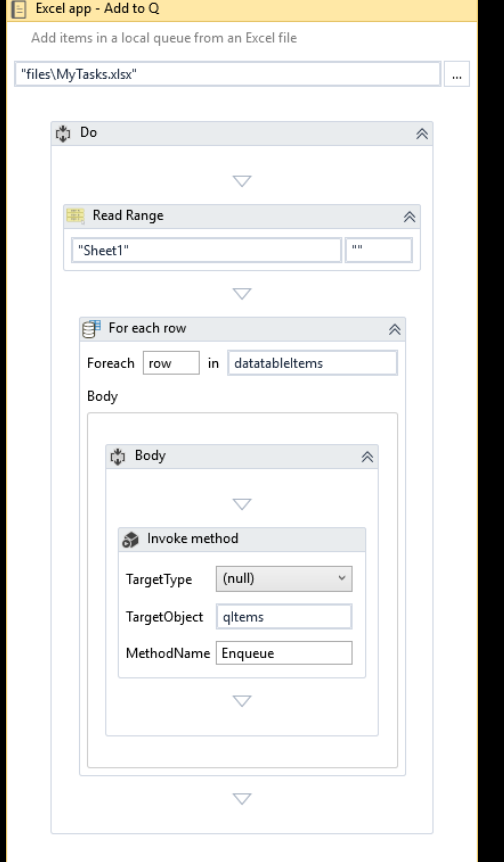
**Number.Offset(0, 1).Value = "Fail"**

**End If**

**Next Number**

**End Sub**

****

1. **Use Variable As a queue :**
2. **qitems = new queue() , Add method : Enqueue Remove Method = DeQueue**
3. **To Retrieve The Values From Queue :**
4. ****
5. ****