HEADLINE: EXPORTING LARGE SHAREPOINT LISTS – IT'S EASIER THAN YOU THINK

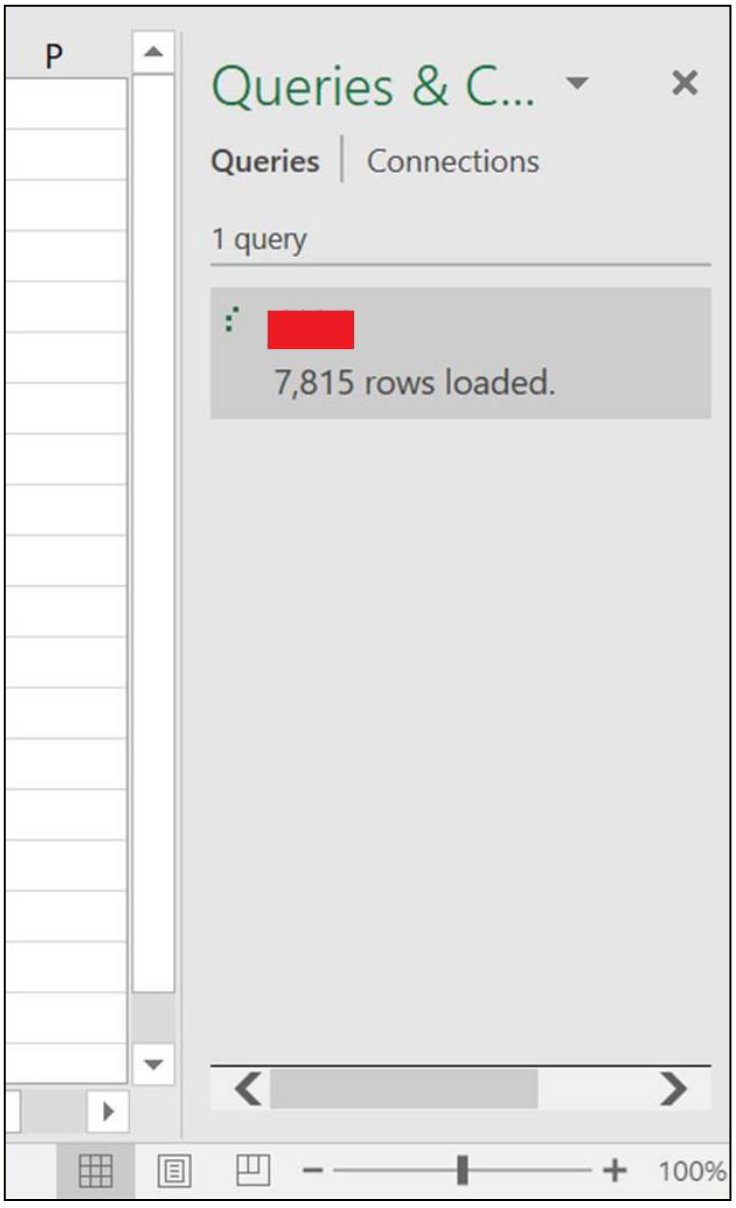
Exporting SharePoint list to Excel or CVS is one of the requirements we get from our customers time and time again.

Typically, we address this by opening Excel and importing data from SharePoint Online. We will go through details later but first; let's see why this approach has one HUGE problem and by HUGE we mean **HUGE**.

The problem

Let's have a look at the following scenario…

We have used the "Excel Import" method and only started to get data from the SharePoint Online list and after about 105 minutes:



7815 rows loaded out of 32932. Do the math and you will have an idea how much time it will take. MS documentations states that we can have 30 million items in a list, so in our scenario with 32932 rows we are working with only 0.1 percent of the limit.

Now do you see what we mean by HUGE???

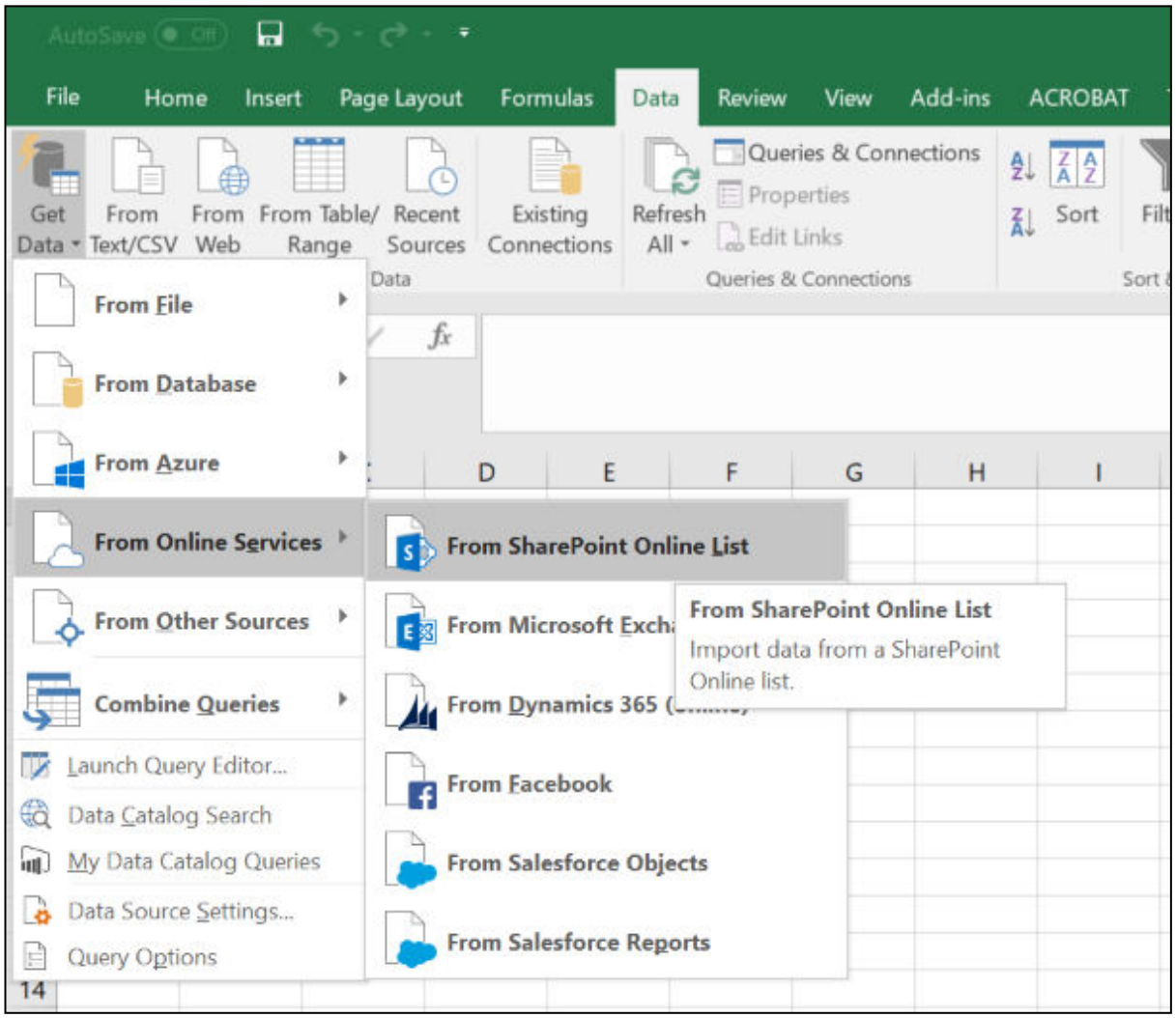
The solution

To export a list with large volume of data, we return to the trusted PowerShell and CSOM.

Don't get us wrong, when list data is not too big, we prefer importing data to Excel but when data volume is very large, we will be using PowerShell and CSOM.

So let's see the first method … Excel Import

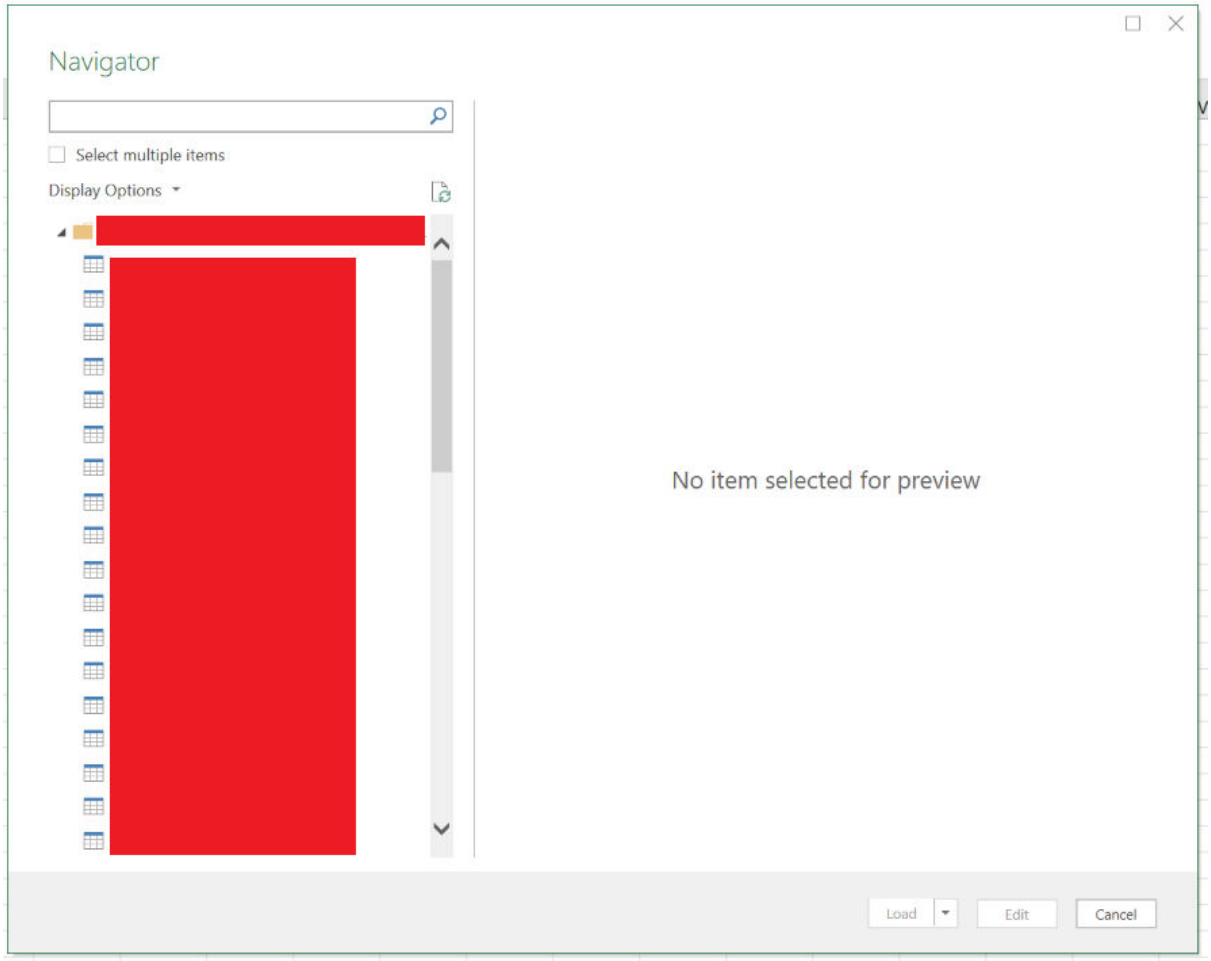
Let's start with importing SharePoint Online list data to Excel. Start Excel and go to select "Data" tab. Click on "Get Data" -> "From Online Services" -> "From SharePoint Online List".



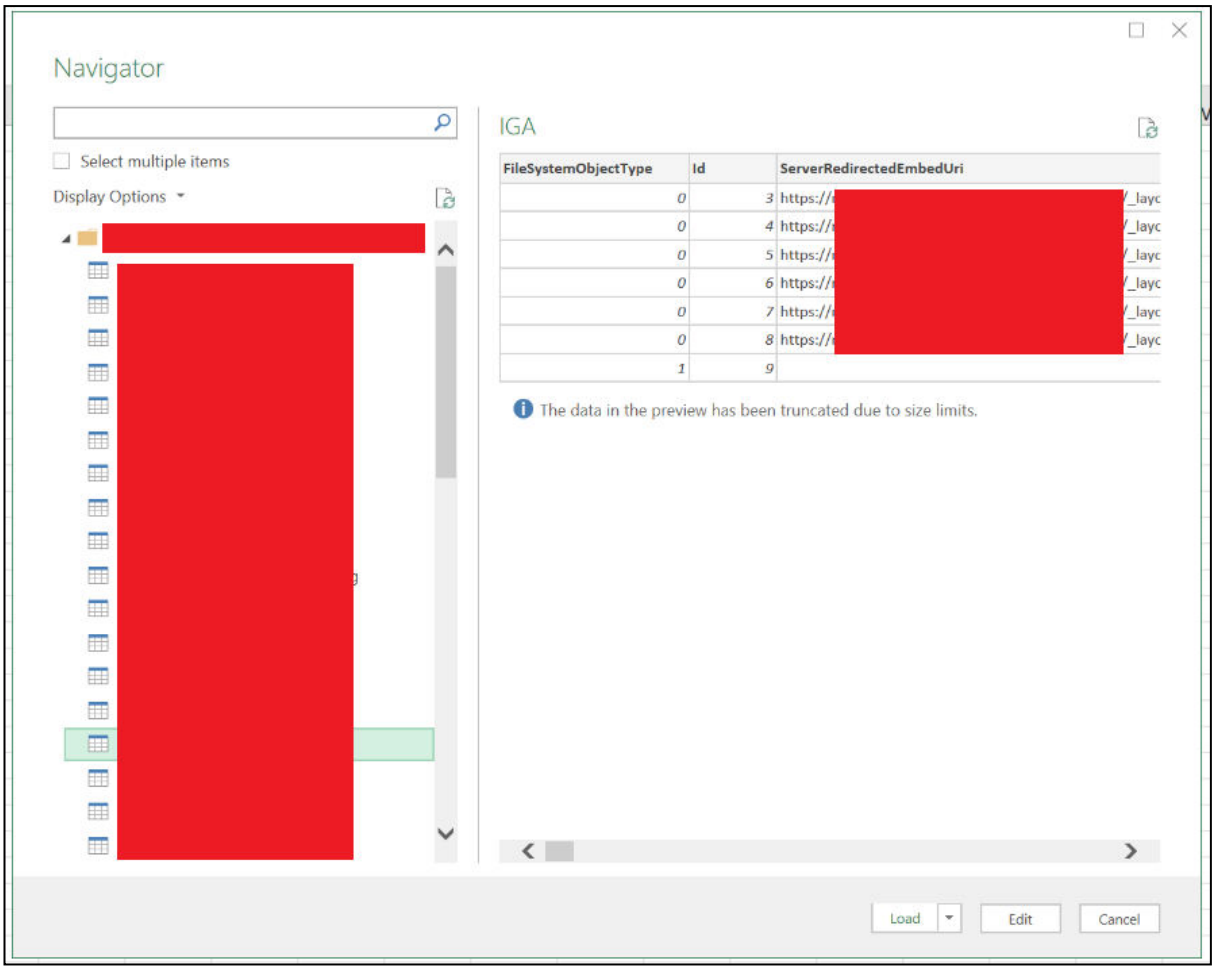
Pop up will appear with the heading "SharePoint lists". It's a little misleading so don't get fooled by the heading and enter URL of your SharePoint Online *site*, not the URL of SharePoint Online *list*.

The following Pop up will confirm that you are on the right track

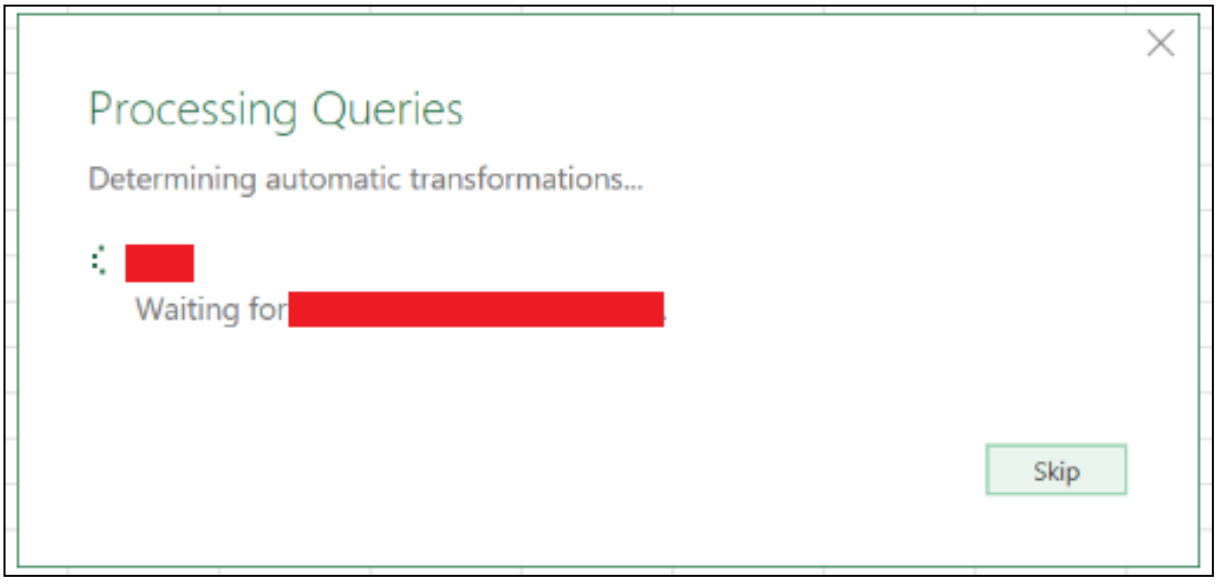
Once connected you will see all your available lists


Navigator
Select multiple items
Display Options
[redacted]
No item selected for preview
Load
Edit
Cancel

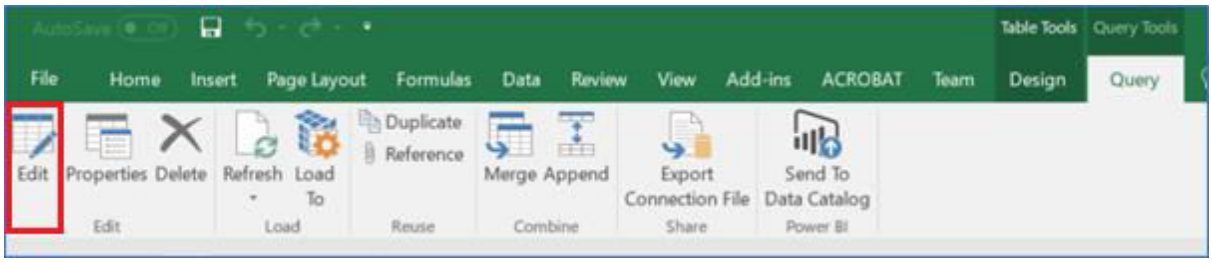
Select desired list and data will appear in preview window



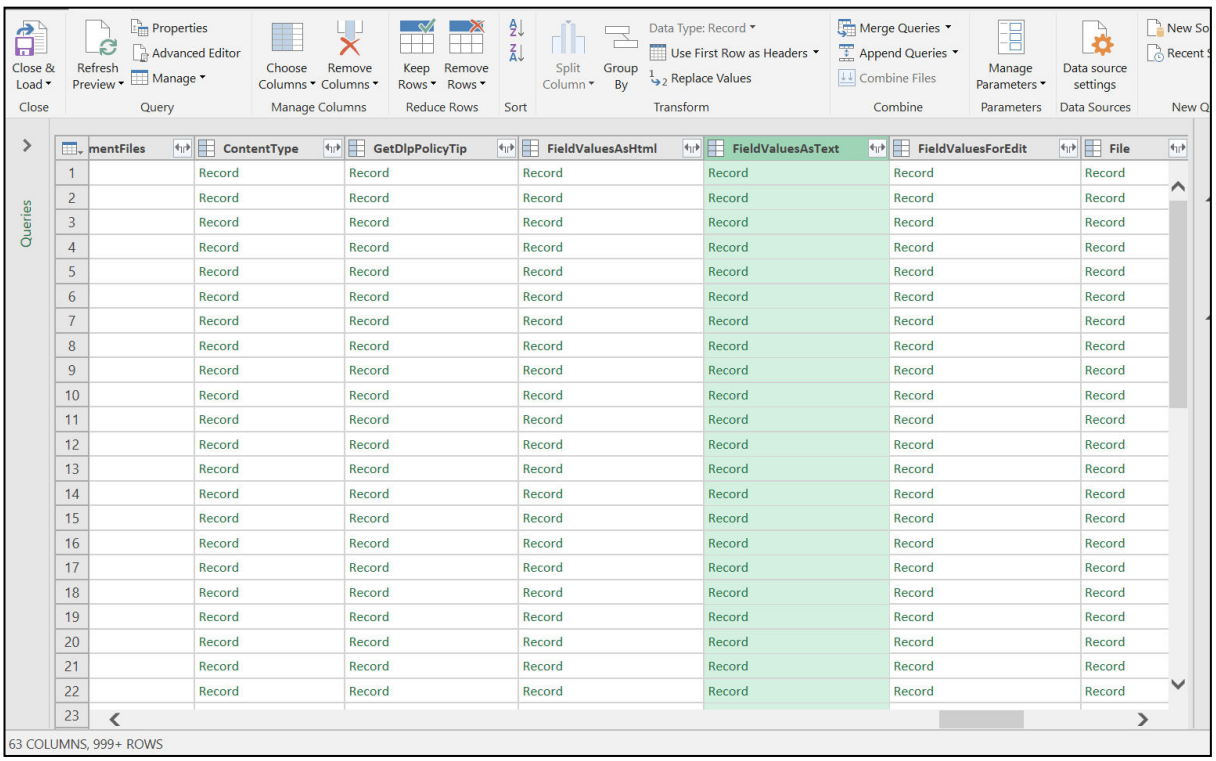
Click on “Load” and the a pop up will appear



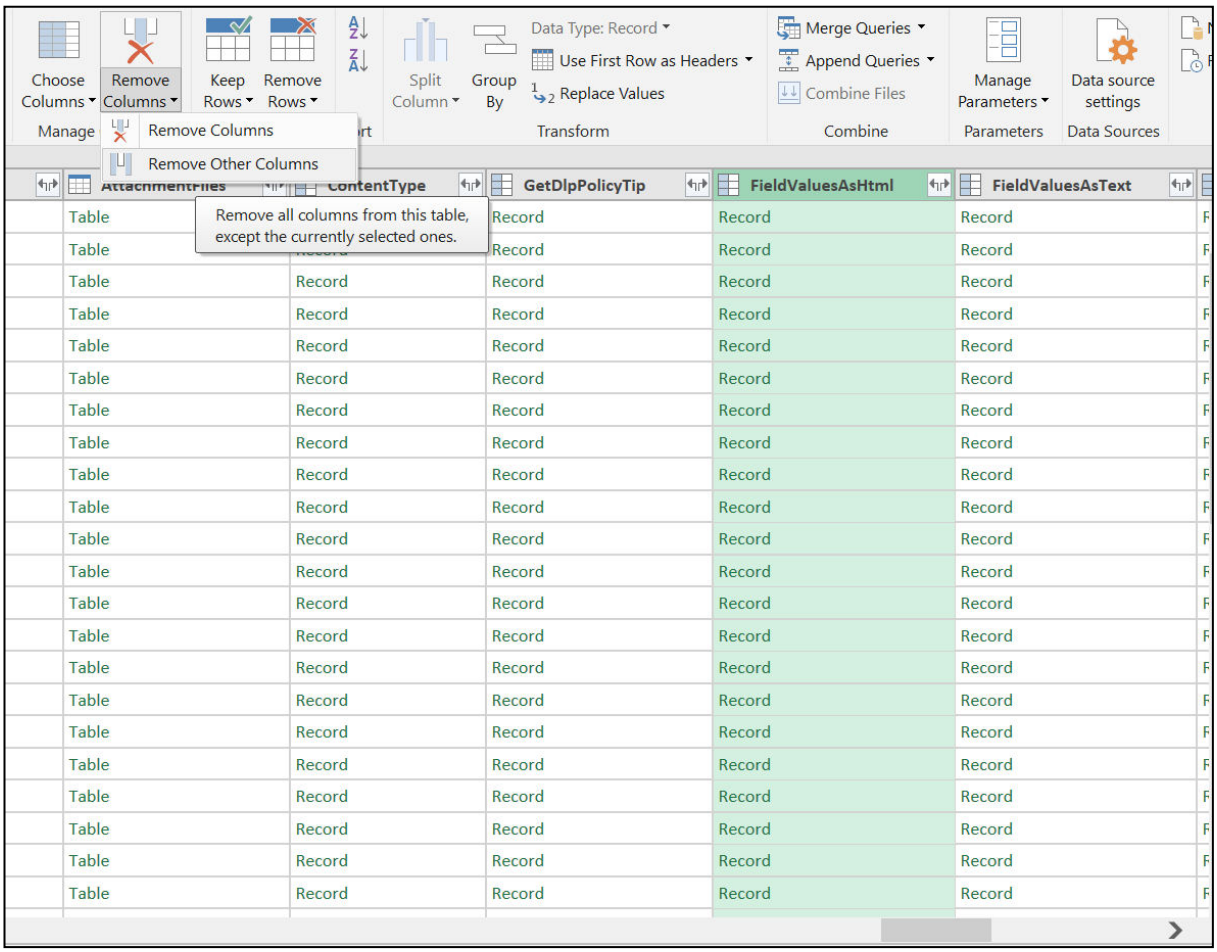
Eventually, you will be able to see the data from the SharePoint Online list in Excel. Don't be alarmed by all the data that appeared in Excel. Go to "Query" tab under "Query Tools" and click on "Edit".



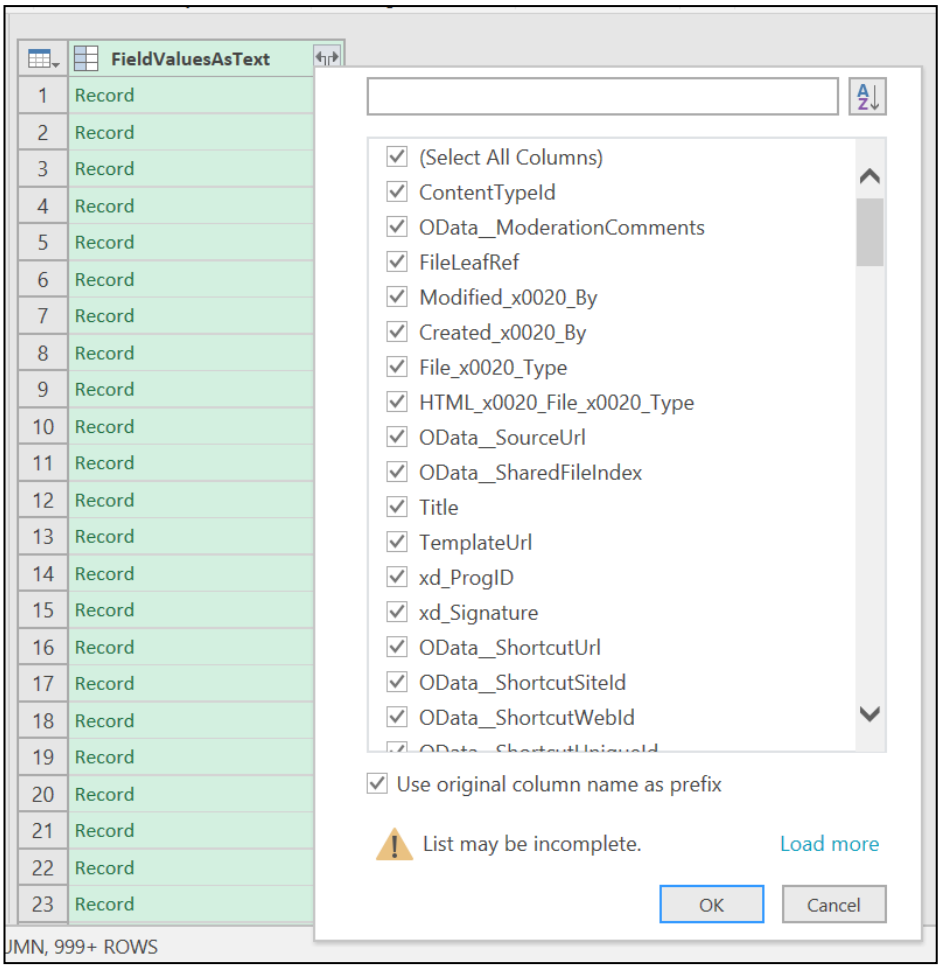
"Query Editor" window will appear. Scroll horizontally and find column "FieldValuesAsText".



Make sure “FieldValuesAsText” is selected and click on “Remove Columns” -> “Remove Other Columns”. This will remove all columns from “Query Editor”.

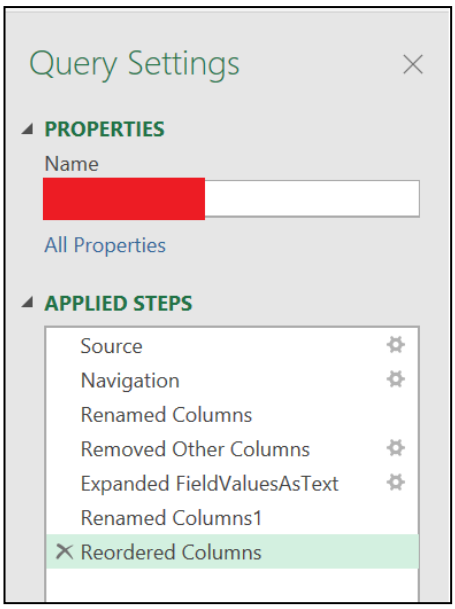


Once all columns are removed, click on the right corner on “FieldValuesAsText” and the following will appear:



Select the columns that you need. Excel will take time to load and you will have all columns from the SharePoint List along with the relevant data. Rename the columns as needed and click “Save & Load”. This will take time as Excel will update itself based on the changes made and we have imported data from SharePoint Online list to Excel.

TIP: *One little tip about the “Query Settings” window. This view displays all the changes performed in sequential order so can undo any change at any point!*



This approach works fine when we are working with small data size but takes a lot time with each step added in Query Settings. One way to try and speed this up is to not add any steps in Query Settings and try to with work with the data populated in Excel from the SharePoint Online list in first place. Again, this works OK for smaller data volumes.

The Other method… PowerShell

Other way that works much better for larger data volumes is to write a PowerShell script and export data from SharePoint Online list to CSV or Excel. We will be using CSV format for simplicity.

1 First step is to make sure that `Microsoft.SharePoint.Client.dll` and `Microsoft.SharePoint.Client.Runtime.dll` are at the following location:

```
Add-Type -Path "C:\Program Files\Common Files\microsoft shared\Web Server
Extensions\15\ISAPI\Microsoft.SharePoint.Client.dll" -ErrorAction Stop
Add-Type -Path "C:\Program Files\Common Files\microsoft shared\Web Server
Extensions\15\ISAPI\Microsoft.SharePoint.Client.Runtime.dll" -ErrorAction Stop
```

2 Now, let's record the values of the following parameters to get the context of SharePoint Online site:

```
$username = "XXXXXX"
$userPassword = "XXXXXXXXX"
$siteURL = "XXXXXXX"
$listtitle = "XXXXXX"
```

3 We will be using the following CAML query which will have OOTB columns. We can add as many columns as desired

```
$qCommand = @"
<View Scope="RecursiveAll">
    <Query>
        <OrderBy Override='True'><FieldRef Name='Modified' /></OrderBy>
    </Query>
    <ViewFields>
        <FieldRef Name='Modified' /><FieldRef Name='Created' />
    </ViewFields>
    <RowLimit Paged="TRUE">5000</RowLimit>
</View>
"@
```

4 Accessing SharePoint Online site and list in script below

```
$secpasswd = ConvertTo-SecureString $userPassword -AsPlainText -Force
$context = New-Object Microsoft.SharePoint.Client.ClientContext($siteURL)
$context.Credentials = New-Object
Microsoft.SharePoint.Client.SharePointOnlineCredentials($UserName, $secpasswd)

$list = $context.Web.Lists.GetByTitle($listtitle)
$context.Load($list)
$context.ExecuteQuery()
```

5 A couple of important parts of the script:

A First is `$position` variable which will store the current position

B and `$itemsinfo` is array of PSObject

```
$position = $null
$itemsinfo = @()
```

6 We will be using Do Until loop and will be terminating when $position variable is null and adding data in $itemsinfo

```
Do
{
    $camlQuery = New-Object Microsoft.SharePoint.Client.CamlQuery
    $camlQuery.ListItemCollectionPosition = $position
    $currentCollection = $list.GetItems($qCommand)
    $context.Load($currentCollection)
    $context.ExecuteQuery()

    $position = $currentCollection.ListItemCollectionPosition

    foreach($listitem in $currentCollection)
    {
        try
        {
        $fieldvalue = @{
            Created =
[System.TimeZoneInfo]::ConvertTimeFromUtc($listitem["Created"], $TZ)
            Modified =
[System.TimeZoneInfo]::ConvertTimeFromUtc($listitem["Modified"], $TZ)
            }
        }catch
        {
            Write-Host $_
        }

        $itemsinfo += New-Object psobject -Property $fieldvalue
    }
}
Until($position -eq $null)
```

7 Last step is to create CSV file

```
$itemsinfo | Select-Object Created, Modified | export-csv
"C:\DataFromSharePointOnlineList.csv" -NoTypeInformation
```

…and the full script…

```
Add-Type -Path "C:\Program Files\Common Files\microsoft shared\Web Server Exten-
sions\15\ISAPI\Microsoft.SharePoint.Client.dll" -ErrorAction Stop
Add-Type -Path "C:\Program Files\Common Files\microsoft shared\Web Server Exten-
sions\15\ISAPI\Microsoft.SharePoint.Client.Runtime.dll" -ErrorAction Stop

$strCurrentTimeZone = (Get-WmiObject win32_timezone).StandardName
$TZ = [System.TimeZoneInfo]::FindSystemTimeZoneById($strCurrentTimeZone)

$username = "XXXXXX"
$userPassword = "XXXXXXXXX"
$siteURL = "XXXXXXX"
$listtitle = "XXXXXX"

$qCommand = @"
<View Scope="RecursiveAll">
    <Query>
        <OrderBy Override='True'><FieldRef Name='Modified' /></OrderBy>
    </Query>
    <ViewFields>
        <FieldRef Name='Modified' /><FieldRef Name='Created' />
    </ViewFields>
    <RowLimit Paged="TRUE">5000</RowLimit>
</View>
"@

$secpasswd = ConvertTo-SecureString $userPassword -AsPlainText -Force
$context = New-Object Microsoft.SharePoint.Client.ClientContext($siteURL)
$context.Credentials = New-Object Microsoft.SharePoint.Client.SharePointOnlineCre-
dentials($UserName, $secpasswd)

$list = $context.Web.Lists.GetByTitle($listtitle)
$context.Load($list)
$context.ExecuteQuery()

$position = $null
$itemsinfo = @()
Do
{
    $camlQuery = New-Object Microsoft.SharePoint.Client.CamlQuery
    $camlQuery.ListItemCollectionPosition = $position
    $currentCollection = $list.GetItems($qCommand)
    $context.Load($currentCollection)
    $context.ExecuteQuery()

    $position = $currentCollection.ListItemCollectionPosition

    foreach($listitem in $currentCollection)
    {
        try
        {
        $fieldvalue = @{
            Created = [System.TimeZoneInfo]::ConvertTimeFromUtc($listitem["Cre-
ated"], $TZ)
            Modified = [System.TimeZoneInfo]::ConvertTimeFromUtc($listitem["Modi-
fied"], $TZ)
            }
        }catch
        {
            Write-Host $_
        }

        $itemsinfo += New-Object psobject -Property $fieldvalue
    }
}
Until($position -eq $null)

$itemsinfo | Select-Object Created, Modified | export-csv "C:\DataFromSharePoin-
tOnlineList.csv" -NoTypeInformation
```

One last workaround if want to create Excel, following code will generate Excel from CSV

```
$excel = New-Object -ComObject excel.application
$excel.visible=$false
$excel.DisplayAlerts = $false

$reportOut = $excel.Workbooks.Add()
$wb = $excel.WorkBooks.Open("C:\DataFromSharePointOnlineList.csv")
$wb.Worksheets.Item(1).Name = "DataFromSharePointOnlineList"
$wb.Worksheets.Copy($reportOut.WorkSheets.Item(1))
$wb.Close(0)

$reportOut.worksheets.item("Sheet1").Delete()
$strdate = get-date
$filename = "C:\DataFromSharePointOnlineList.xlsx"
$reportOut.SaveAs($filename,[Microsoft.Office.Interop.Excel.XlFileFormat]::xlOpenXM
LWorkbook)
$reportOut.Close(0)
$excel.Quit()
```