Actuals - debug critical error

When running the interface return a **Critical Error**, the **interface stops working** until this one is removed.

This works as a **hook**, that give you the chance to debug and **find the root that generated the Error**.

To know before starting

run_id is increasing +1 each time an interface runs

Can we also use it to identify when a single actual was imported by the interface??

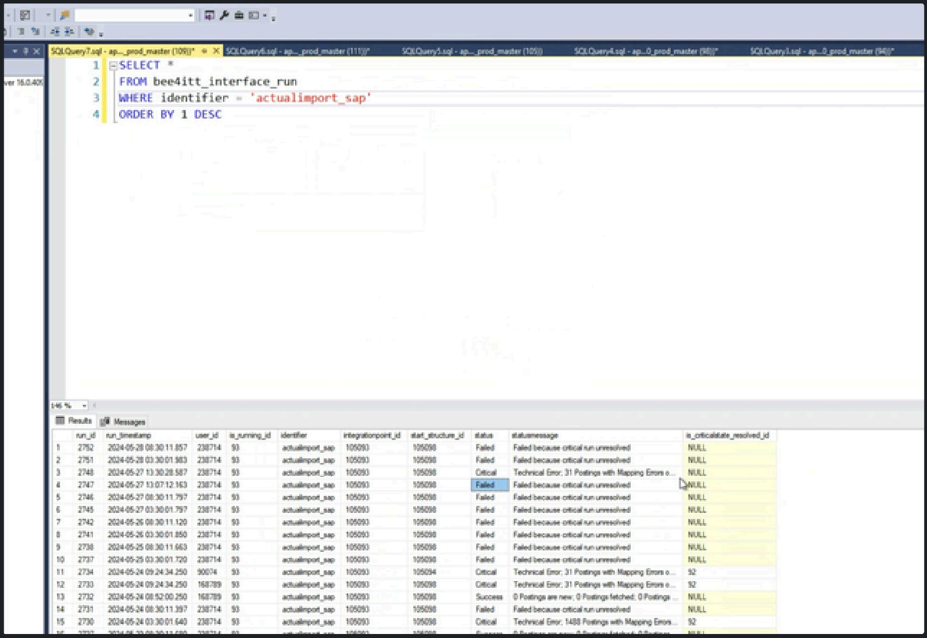
Find Critical Error’s id

we need to find the run_id of the Critical Error, so we can identify when it happened

Two options

- You can check on bee360 here
- Or by database as follow

```sql
SELECT *
FROM bee4itt_interface_run
WHERE identifier = 'actualimport_sap'
ORDER BY 1 DESC
```



SQLQuery7.sql - ap..._prod_master (109))*
SQLQuery6.sql - ap..._prod_master (111))*
SQLQuery5.sql - ap..._prod_master (105))
SQLQuery4.sql - ap...0_prod_master (98))*
SQLQuery3.sql - ap...0_prod_master (94))*
146 %
Results
Messages

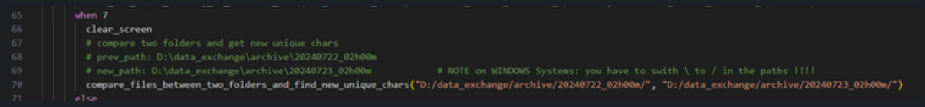
```sql
SELECT *
FROM bee4itt_interface_run
WHERE identifier = 'actualimport_sap'
ORDER BY 1 DESC
```

	run_id	run_timestamp	user_id	is_running_id	identifier	integrationpoint_id	start_structure_id	status	statusmessage	is_criticalstate_resolved_id
1	2752	2024-05-28 08:30:11.857	238714	93	actualimport_sap	105093	105098	Failed	Failed because critical run unresolved	NULL
2	2751	2024-05-28 03:30:01.983	238714	93	actualimport_sap	105093	105098	Failed	Failed because critical run unresolved	NULL
3	2748	2024-05-27 13:30:28.587	238714	93	actualimport_sap	105093	105098	Critical	Technical Error; 31 Postings with Mapping Errors o...	NULL
4	2747	2024-05-27 13:07:12.163	238714	93	actualimport_sap	105093	105098	Failed	Failed because critical run unresolved	NULL
5	2746	2024-05-27 08:30:11.797	238714	93	actualimport_sap	105093	105098	Failed	Failed because critical run unresolved	NULL
6	2745	2024-05-27 03:30:01.797	238714	93	actualimport_sap	105093	105098	Failed	Failed because critical run unresolved	NULL
7	2742	2024-05-26 08:30:11.120	238714	93	actualimport_sap	105093	105098	Failed	Failed because critical run unresolved	NULL
8	2741	2024-05-26 03:30:01.850	238714	93	actualimport_sap	105093	105098	Failed	Failed because critical run unresolved	NULL
9	2738	2024-05-25 08:30:11.663	238714	93	actualimport_sap	105093	105098	Failed	Failed because critical run unresolved	NULL
10	2737	2024-05-25 03:30:01.720	238714	93	actualimport_sap	105093	105098	Failed	Failed because critical run unresolved	NULL
11	2734	2024-05-24 09:24:34.250	90074	93	actualimport_sap	105093	105094	Critical	Technical Error; 31 Postings with Mapping Errors o...	92
12	2733	2024-05-24 09:24:34.250	168789	93	actualimport_sap	105093	105098	Critical	Technical Error; 31 Postings with Mapping Errors o...	92
13	2732	2024-05-24 08:52:00.250	168789	93	actualimport_sap	105093	105098	Success	0 Postings are new; 0 Postings fetched; 0 Postings ...	NULL
14	2731	2024-05-24 08:30:11.397	238714	93	actualimport_sap	105093	105098	Failed	Failed because critical run unresolved	NULL
15	2730	2024-05-24 03:30:01.640	238714	93	actualimport_sap	105093	105098	Critical	Technical Error; 1488 Postings with Mapping Errors...	92

Identify the cause

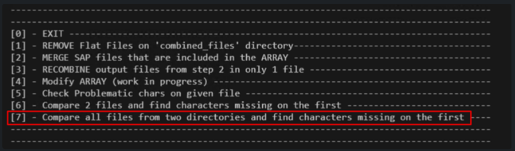
Compare SAP files between the day of the Critical Error and the previous one.

- Find when the critical error happened
- **Adjust the script** `D:\data_exchange\actuals\CSV_MERGER\menu.rb` with the right paths (check pic below)



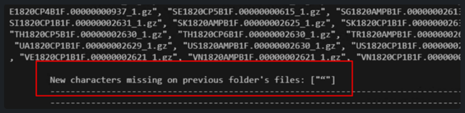
```ruby
when 7
  clear_screen
  # compare two folders and get new unique chars
  # prev_path: D:\data_exchange\archive\20240722_02h00m
  # new_path: D:\data_exchange\archive\20240723_02h00m            # NOTE on WINDOWS Systems: you have to swith \ to / in the paths !!!!
  compare_files_between_two_folders_and_find_new_unique_chars("D:/data_exchange/archive/20240722_02h00m/", "D:/data_exchange/archive/20240723_02h00m/")
else
```

- **Run Script to find NEW characters, comparing the files from SAP that caused the Critical errors VS files from SAP from Previous day.**
  - From the terminal: `ruby D:\data_exchange\actuals\CSV_MERGER\menu.rb`



```
-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
[0] - EXIT ---------------------------------------------------------------------------------
[1] - REMOVE Flat Files on 'combined_files' directory---------------------------------------
[2] - MERGE SAP files that are included in the ARRAY ---------------------------------------
[3] - RECOMBINE output files from step 2 in only 1 file -----------------------------------
[4] - Modify ARRAY (work in progress) ------------------------------------------------------
[5] - Check Problematic chars on given file ------------------------------------------------
[6] - Compare 2 files and find characters missing on the first ----------------------------
[7] - Compare all files from two directories and find characters missing on the first ----
-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
```

- We get an array with NEW chars



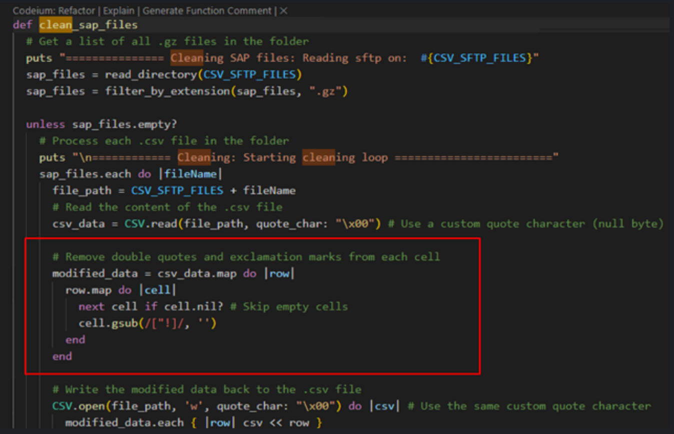
```
E1820CP4B1F.00000000937_1.gz", "SE1820CP5B1F.00000000615_1.gz", "SG1820AMPB1F.0000000261
SI1820CP1B1F.00000002631_1.gz", "SK1820AMPB1F.00000002625_1.gz", "SK1820CP1B1F.000000026
"TH1820CP5B1F.00000002630_1.gz", "TH1820CP6B1F.00000002630_1.gz", "TR1820AMPB1F.00000002
 "UA1820CP1B1F.00000002629_1.gz", "US1820AMPB1F.00000002630_1.gz", "US1820CP1B1F.0000000
, "VE1820CP1B1F.00000002621_1.gz", "VN1820AMPB1F.00000002621_1.gz", "VN1820CP1B1F.000000

        New characters missing on previous folder's files: ["“"]
        ----------------------------------------------------------------------------
        ----------------------------------------------------------------------------
```

- Remove Critical Error as explained before
- Open all the SAP files on Notepad++ that should contain the NEW problematic char
- Clean the NEW characters from SAP Files (using “search on all opened files”)
- Run the interface again
- The Critical Error should be solved now.
    - If this didn't solved the error, the new character was not the cause, or there is an additional cause.

Make it permanent

Add the new problematic characters to be cleaned every day

- The function that cleans the files is located on `D:\data_exchange\actuals\CSV_MERGER\lib\library.rb`
- function is called `clean_sap_files`
- On the pic below you can find the line `cell.gsub(/["!]/, '')` that contains the symbols to be removed. In this case Double quotes `"` and Exclamation marks `!`
    - You can add additional symbols here. For example to add `~` you would have to set the line as: `cell.gsub(/["!~]/, '')`
- ATENTION ! This script is called every day by the interface to clean, move and combine the files.
    - >>>> TEST the changes deeply before finishing
    - Compare the results before and after Cleaning.
- You can run the script from the task scheduler or by running from Terminal `ruby D:\data_exchange\actuals\CSV_MERGER\combine_and_archive.rb`
    - The script will use the files located on `D:\SFTP-Files` so be careful to make Backup if needed



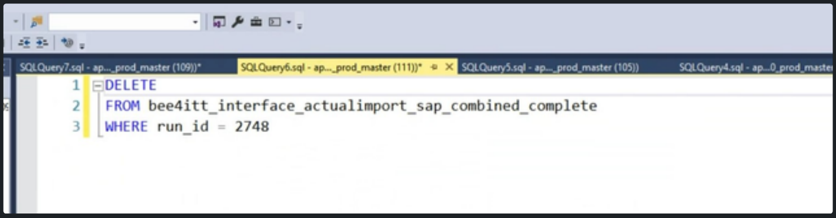
```ruby
def clean_sap_files
  # Get a list of all .gz files in the folder
  puts "=============== Cleaning SAP files: Reading sftp on:  #{CSV_SFTP_FILES}"
  sap_files = read_directory(CSV_SFTP_FILES)
  sap_files = filter_by_extension(sap_files, ".gz")

  unless sap_files.empty?
    # Process each .csv file in the folder
    puts "\n============ Cleaning: Starting cleaning loop ========================"
    sap_files.each do |fileName|
      file_path = CSV_SFTP_FILES + fileName
      # Read the content of the .csv file
      csv_data = CSV.read(file_path, quote_char: "\x00") # Use a custom quote character (null byte)

      # Remove double quotes and exclamation marks from each cell
      modified_data = csv_data.map do |row|
        row.map do |cell|
          next cell if cell.nil? # Skip empty cells
          cell.gsub(/["!]/, '')
        end
      end

      # Write the modified data back to the .csv file
      CSV.open(file_path, 'w', quote_char: "\x00") do |csv| # Use the same custom quote character
        modified_data.each { |row| csv << row }
```

Delete the Critical Error by run_id

Two options:

- You can do it from database as below

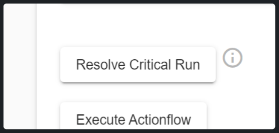
```sql
DELETE
FROM bee4itt_interface_actualimport_sap_combined_complete
WHERE run_id = xxxxxx
```



SQLQuery7.sql - ap..._prod_master (109))*
SQLQuery6.sql - ap..._prod_master (111))*
SQLQuery5.sql - ap..._prod_master (105))
SQLQuery4.sql - ap...0_prod_maste

```sql
DELETE
FROM bee4itt_interface_actualimport_sap_combined_complete
WHERE run_id = 2748
```

- or from bee360 integration point



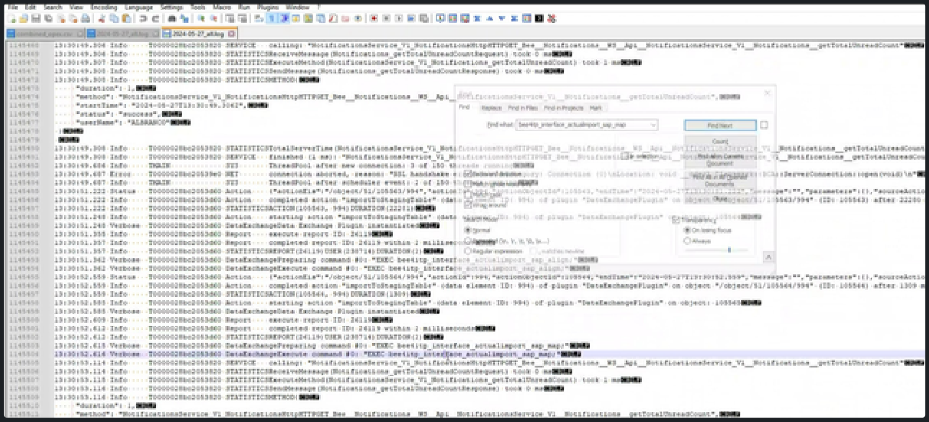
Debugging logs (OPTIONAL)

NOTE This step might be not needed

Now that we know when it happened, we can check the right log file.

Note that the error might have happened on APT or APP servers, depending of the load balancer. So you need to **check both server's logs**

- We go to **D:\bee360_prod\server_bee360\log**
- Open the right **[date]_all.log** according the date you look for
- Search for **bee4itp_interface_actualimport_sap_map**
- If the interface runs more than once per day, you will find it here.
  - **Choose the line** with the right time
  - **Copy the identifier** that look something like `T0000017ad9594570`



- Now adjust the identifier `T0000017ad9594570` to `Error    T0000017ad9594570`
  - You have to keep the right number of blank spaces between both words, so it match correctly. (4 blank spaces for 'Error'). It might be different if different tag is used.
- You should be able to find some error like on below pictures, close to the time where the interface was executed