# Deaths Routine Dataload on SQL and Google

Contents

[Deaths Routine Dataload on SQL and Google 1](#_Toc164677738)

[**Introduction** 1](#_Toc164677739)

[Stage 1: Uploading to SQL Database 1](#_Toc164677740)

[Resize columns 2](#_Toc164677741)

[Export Data 2](#_Toc164677742)

**Stage 2: Uploading to Google** 2

Appendix – if issues arise at each stage**…………………………………………………………………………..**2

# **Introduction**

This document details what must be done to ensure that Death’s data on SQL and google platform are properly loaded and modified.

The following stages were followed in loading the death datasets into SQL database:

## Stage 1: Uploading to SQL Database

* From my space, right click on tasks,
* click on import data,
* on Data source, click on Flat file source,
* browse, and pick the file name,
* change Text qualifier from none to \*(remove speechmarks),
* check the columns on the left-hand side to inspect input in each rows,
* then click on advanced and highlight all rows to set the OutputColumnwidth.
* After that stage, SQL Server Native Client 11.0 was selected as Destination Database.
* Then, click on next until finish which will execute. If there are no error messages, then it will start copying rows.
* Once the data is loaded, then add field cb\_datasetref varchar(35)
* update cb\_datasetref to be the same as the zipfilename for each set.

(For instance, using the Alter Table, Update table)

## 

## Resize columns

After that process, Use the ResizeColumns script to remove spaces and set the columns to its original characters. Resizer scripts can be found on: bhts-conydevwd2: \ConnectedBradford\Scripts\ src\_StagingDatabase\_BDCT \Scripts\ ResizeColumnsBasedONMaxDataLength2

Append into StagingDatabase

At completion, append from your database into: src\_StagingDatabase into the tables NEC\_Deaths\* etc.

Once completed move the files into D:\ConnectedBradford\Scripts\src\_StagingDatabase\Loaded\.

## 

## Export Data

The datasets were exported using the export.bat file. (Example in folder ConnectedBradford\Scripts\src\_StagingDatabase\Export\_Scripts\9p\_ExportDeathsData.bat (this may need amending based on the tbl names) was used.

## 

## **Stage 2: Uploading to Google**

* 1. Each table was uploaded to bucket, and a python script was executed for each to load them into my space. [CB\_FDM\_DeathCertificates/code/PAA\_NEC\_Deaths\_Load4romGCP.ipynb at main · ConnectedBradford/CB\_FDM\_DeathCertificates (github.com)](https://github.com/ConnectedBradford/CB_FDM_DeathCertificates/blob/main/code/PAA_NEC_Deaths_Load4romGCP.ipynb)
  2. Then, these new tables were placed in CB\_STAGING\_DATABASE\_FDM\_Format

using the script " PAA\_NEC\_DeathsFDMFormat BQRun.txt" (This will need amending based on the tbl names) which can be found in the GitHub repo.

[CB\_FDM\_DeathCertificates/code/PAA\_NEC\_DeathsFDMFormat BQRun.txt.ipynb at main · ConnectedBradford/CB\_FDM\_DeathCertificates (github.com)](https://github.com/ConnectedBradford/CB_FDM_DeathCertificates/blob/main/code/PAA_NEC_DeathsFDMFormat%20BQRun.txt.ipynb)

* 1. Finally, the spreadsheet <https://bthftnhsuk-my.sharepoint.com/:x:/g/personal/john_birkinshaw_bthft_nhs_uk/ETwHO0-qi3pFkb7vZ5XE3MoB2sZoN_casKrLkhKpUOxeIg?e=peB1Dr> was updated (lines 360-365)

**Appendix – if issues arise at each stage.**

## Stage 1: Uploading to SQL Database

Normally, we set at 255 but the death data had columns with long characters. So, I troubleshooted so many time for a suitable columnwidth (500, 1000, 1,500, etc).

## Note: The larger the ColumnWidth is, the more time it takes to copy.

## Resize columns

(For instance, the first and second columns in Dealths\_QWO\_20240201\_RELEASE3.7 are 64, 6, and 1, respectively. So, if you read it in with a columnwidth of 1000, you need to resize it to its original character).

Append into StagingDatabase

**Note**: You may need to inspect the columns if any issue arises. It is possible the tables have different name. You may need to rename the same way using the renamer script.

* I also noticed output\_specification\_1 columns do not have the same column names, so I ran a query to rename and set the columns 'Column 0', 'Column 1', and 'Column 2' as 'Field Name', 'Format', and 'Field Description' respectively in the table [dbo].[Output Specification\_1].
* When I inspected the reference\_PostcodeGeography table columns, some characters were larger than what we have in the destination table. So, there is need to verify. To do this, I executed an alter script. This code help to identify the maximum length of each column and decide whether to alter them accordingly. The alter script was included within the stored procedure `AlterColumnLengths. After executing the stored procedure `AlterColumnLengths`, this stored procedure does not return any results because SQL Server Management Studio or our SQL client did not display any output. However, to verify if the alterations were successful by checking the column lengths in the **[dbo].[reference.PostcodeGeography]** table. I simply ran a query to check the maximum lengths of the columns. This query then confirms whether the alterations were successful and if the column lengths are within the desired limits.
* After running each of these queries individually to get the maximum length of characters for each column in the [dbo].[reference.PostcodeGeography] table. Once we have verified these lengths, we can decide whether to proceed with altering the columns based on the requirements. Then, proceed to append into the tables, but need to adjust (increase) the columns that are limited in the destination table to accommodate the new data to append.
* To modify the table, we need proper access/ permission to modify the NEC\_Deaths table. After access was granted, a message was received saying saving changes is not permitted. It says “the changes you makes requires the following tables to be dropped and re-created. You can either update a table that cannot be re-created or enabled the option Prevent saving changes that require the table to be re-created.”
* The message we encounter is due to a setting in SQL Server Management Studio (SSMS) called "Prevent saving changes that require the table to be re-created." This setting is enabled by default to prevent accidental loss of data or changes that require table recreation, such as altering the data type or dropping columns.

To resolve this issue and allow SSMS to recreate the table when necessary, I disabled this option by:

* Open SQL Server Management Studio (SSMS).
* Go to the "Tools" menu.
* Select "Options."
* In the Options window, expand "Designers."
* Select "Table and Database Designers."
* Uncheck the option "Prevent saving changes that require the table to be re-created."
* Click "OK" to save the changes.

Once this was done, the tables were created and modified without restriction. Then I proceeded to creating the NEC\_Death1 table and necessary modifications were made.

**Note**: Remember to enable this option again after you have finished making your changes to ensure data integrity and prevent accidental table recreation.

* To overcome the issues of appending reference.PostcodeGeography and  reference.PostcodeGeography\_1 to src\_StagingDatabase because the table design will not change (Because of its size). A latest version was created with the correct sizing by right clicking the table name and select “Script table as”).

Then, all existing data from the old table was automatically appended into the new table which I renamed NEC\_reference\_PostcodeGeography1. When the data is appended, I removed the original table and rename it using the old name.

Note: Drive D can be access by signing into Remote Desktop connection. The text files were moved to the loaded folder.

Export data

This part could be achieved by checking for the death batch file in the I drive, copy it to desktop for easy access, right click to edit. Edit by putting all the dataset names in the file to export at once. This will export the files in text format and the CSV format would be deleted to reduce size during google upload.