Creating MFNs

[Introduction 1](#_Toc14366207)

[Business requirements 1](#_Toc14366208)

[Inputs to the process 2](#_Toc14366209)

[UK non-liberalised measures 2](#_Toc14366210)

[List of authorised codes 3](#_Toc14366211)

[List of all current commodity codes 4](#_Toc14366212)

[Step by step guidance to produce XML data set 6](#_Toc14366213)

[Anatomy of the application 9](#_Toc14366214)

# Introduction

The MFN creation process has been radically adjusted in preparation for November 2019’s EU Exit in order to bring the MFN duties higher up the commodity code tree. In March / April’s abortive attempts, the application had been set such that the measures would always be set at the very tips of each of the commodity tree’s branches: the leaves.

While this was highly effective at ensuring that there were no ME32 errors (when measures are only assigned to leaf nodes, there are no higher or lower nodes against which to conflict), this led to a slight challenge, in that, if the EU introduces late new commodity codes, which is possible, then the commodity will have no duty assigned against it by default – an additional data load is required to assign a new duty to that new commodity code. In most cases, the commodity code has been introduced not to vary the MFN duty, rather to add some other trade function such as a trade remedy, a suspension or an import / export control. Therefore, the assignment of duties to the leaves is a hindrance rather than a benefit.

In addition, assigning duties to all leaves creates bigger data loads and a greater operational burden at a later date.

# Business requirements

| ID | Requirement |
| --- | --- |
| 1 | Parameterise Brexit date, so that the process can be repeated if required |
| 2 | End date all of the EU’s measures on Brexit date -1 |
| 3 | Start all required UK measures on Brexit date |
| 4 | Receive a list of non-liberalised duties from the analyst team in a specified format in order to facilitate the load of data into TAP and downstream systems |
| 5 | Abide by the EU’s assignment of 105 (end use) and 103 (standard third country duty) measures according to the commodity code’s nature |
| 6 | Ensure that no business rules are violated, most notably ME32 in the process of creating and loading data |
| 7 | Ensure that there is full coverage across all commodity codes for an MFN duty |
| 8 | Ensure that no duties are assigned:  at chapter level (e.g. 0100 00 00 00) *or*  at HS heading level (e.g. 0101 00 00 00) |
| 9 | Create XML to load to TAP and downstream systems |

# Inputs to the process

The following inputs are required to ensure that this process can run smoothly – where appropriate, these are detailed beneath the table.

| Item | Description |
| --- | --- |
| Resource | DDaT staff that can run the necessary Python script |
| Database | Offline version of the TAP database |
| Script | Necessary Python script for XML generation |
| UK non-liberalised measures | Excel spreadsheet of UK non-liberalised commodity codes in specified format |
| List of authorised use codes | A full list of the commodity codes that are to be treated as subject to authorised use (end use), so that the measure types assigned can be differentiated according to projected use of commodity code |
| List of all current commodity codes | A full list of the entire commodity tree, in a prescribed format as described below |

## UK non-liberalised measures

The structure of the document **Copy of 181207 OFFSEN - 2018 Tariff schedule DFID and Cheese.xlsx** as initially provided in March / April is suitable for this round of Brexit. This Excel spreadsheet contains:

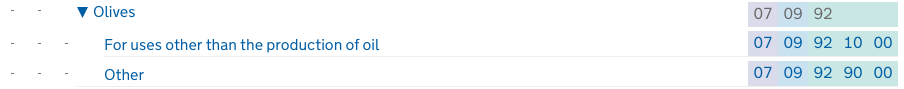
* a full list of the commodity codes that are current (as of a given date – it needs to be refreshed)
* a definition as to whether those commodity codes are to have duties retained against them. In the March / April version (which worked), column AB listed the Boolean values (TRUE / FALSE) as to whether duties are to be retained against those commodities. If set to TRUE, then the commodity is going to be included in the UK tariff as a non-liberalised line. If set to FALSE, then the duty will be liberalised (set to zero).
* the actual duties to be retained: these are split into two columns: Ad valorem and Specific (columns L & M). In fact, as will be seen, the Specific column is later split further into Specific and Minimum, as on occasion the floor (minimum) tariff duty is included with the specific, which complicates the data extraction.
* By filtering to TRUE on column AB, the non-liberalised duties are displayed
* The process for managing this data is described below
* Importantly, **we are only interested in non-zero duties in this sheet** – the script will work out the rest: in the previously supplied version, the data did contain some zeros: these are removed manually prior to running the script.

**Critical note on content and granularity of file**  
Critically, the file used in March / April only listed retained duties down to the 8-digit level, whereas in fact 10 digits were needed. The column containing the commodity code identifier, column J is entitled **HS8 (with 0).** This needs to contain the 10-digit code in all cases, regardless of whether the duty is assigned at the 8- or 10-digit level.

## List of authorised codes

The **list of authorised codes** is required in order to correctly differentiate the duties that are applicable to commodity codes of varying types. There are c. 480 commodities that have been explicitly set up to provide typically reduced duty (not always zero) by the European Commission for traders who are importing goods for authorised / end use, i.e. they are going to be using the imported good in the construction of something else (as an ingredient or raw material), materially changing the good into a different commodity: they must be able to prove this in order to take advantage of the reduced rate. Typically, in introducing a commodity of authorised use type, the EU has taken a parent commodity code and split it into two subsidiary codes.

The example below illustrates this. For Olives, the code “0709920000” has been split into two subsidiary codes. Of these codes, one “0709921000” is designed for end-use (the olives will be transformed into olive oil); whereas the other “0709929000” has been designed for olives that will be sold as olives with no transformation.



The following measure types are used:

| Measure type ID | Description |
| --- | --- |
| 103 | Third country duty (i.e. the basic MFN tariff duty) – this is used for all non-end-use products |
| 105 | Non preferential duty under end-use *or*  Non preferential duty under authorised use (at it will be called in the UK). This is used for all end-use products. |

The differentiation between the authorised use and the standard third country duty is largely one of labelling. The way in which the duty is calculated is the same regardless. The differentiation is especially unimportant, given that the UK is expected to liberalise nearly all tariff duties: in March, only c. 460 duties were retained, the rest were set to zero: this is expected to be near-repeated in November.

However, **from a legislative perspective** it is important to **retain the differentiation**.

The rules that are applied are as follows:

If the EU had applied a 105 duty to the commodity code instead of a 103, then the 105 will be retained in the UK tariff.  
  
If the EU had assigned as 103 measure type, then the 103 measure type will be retained.

The simple data file to support this requirement has the following properties:

| Property | Description |
| --- | --- |
| Filename | measure\_type\_105.csv |
| Stored in … | /source subfolder (relative to the executable script) |
| Header row | A header row is required, with the single column header "goods\_nomenclature\_item\_id" |
| Columns | There is just a single column in this file: the "goods\_nomenclature\_item\_id" or commodity code identifier for the commodities that are subject to authorised use – in this file we do not care about the duties, as these are to be assigned in the analyst-provided data file (see above in section UK non-liberalised measures) |
| Row count | c. 480 |
| SQL to reproduce | select distinct goods\_nomenclature\_item\_id from measures where measure\_type\_id = '105' and validity\_end\_date is null order by 1 |
| Database against which to reproduce | The EU master database – up to date with all latest incremental data files. |

All other commodities that are not included in this file

## List of all current commodity codes

In order to be able to determine how to assign the relevant measures against the commodity codes, it is necessary to have a full list of all the commodity codes on the EU system, as is correct at the latest possible date.

Creating this list manually is hugely complex, but simple by running (on a very powerful machine) the script

select \* from ml.goods\_nomenclature\_export\_brexit ('01%')  
order by goods\_nomenclature\_item\_id, producline\_suffix

and then exporting the resultant data to a CSV file. This file contains the following data fields:

| Field | Description |
| --- | --- |
| goods\_nomenclature\_sid | The unique SID of the nomenclature item – not used |
| goods\_nomenclature\_item\_id | The 10-digit commodity code – used |
| producline\_suffix | The product line suffix – used: a product line suffix of "80" indicates that a measure can be assigned to the commodity code, any other values dictates that a measure cannot be assigned to the commodity code. Please note the deliberate mis-spelling to match the misspelling in the Taric format |
| validity\_start\_date | The date the commodity code first appeared – not used (though a parameter for the creation of this document is that the commodity code's validity start date must have passed) |
| validity\_end\_date | The end date – if the end date has passed, then the data not included in this file. |
| description | The verbatim description of the commodity code |
| number\_indents | The hierarchical indentation of the commodity code, where:   * 0 is the HS chapter * 0 is also the HS heading (stet.) * 1 is the HS subheading * etc.   This is critical in working out the parent and child relationship between commodity codes. |
| goods\_nomenclature\_description\_period\_sid | Not used |
| desc\_validity\_start\_date | Not used |
| desc\_validity\_end\_date | Not used |
| nice\_description | The description indented with leading hyphens to ease reading– not used |
| hs\_chapter | The HS chapter in which the commodity code resides– not used |
| hs\_section | The HS section in which the commodity code resides– not used |
| node | The commodity code with insignificant trailing zeroes removed – not used |
| leaf | Set to "1" for declarable commodities at the extremity of branches - used |
| significant\_digits | Determines the number of digits in the 10-digit commodity code that are not trailing zeros. Digits are **dealt with in pairs**, e.g.  01 00 00 00 00 has 2 significant digits  01 01 00 00 00 has 4 significant digits  01 01 01 00 00 has 6 significant digits  01 01 01 01 00 has 8 significant digits  01 01 01 01 01 has 10 significant digits |

The file to support this requirement has the following properties:

| Property | Description |
| --- | --- |
| Filename | mfns\_nov\_19.csv |
| Stored in … | /source subfolder (relative to the executable script) |
| Header row | Required (as per columns noted above) |
| Columns | Noted above |
| Row count | c. 24,450 |
| SQL to reproduce | select \* from ml.goods\_nomenclature\_export\_brexit ('01%')  order by goods\_nomenclature\_item\_id, producline\_suffix |
| Database against which to reproduce | The EU master database – up to date with all latest incremental data files. |

# Step by step guidance to produce XML data set

### Prior steps required

Due to the ME32 rule, it is required that all MFN data that might clash with this MFN data is end-dated before the new data is loaded. This is documented elsewhere.

**ME32 states:**  
"There may be no overlap in time with other measure occurrences with a goods code in the same nomenclature hierarchy which references the same measure type, geo area, order number, additional code and reduction indicator. This rule is not applicable for Meursing additional codes."

Essentially this means that, at any given time, a measure of any given type may not and must not appear more than once anywhere up and down a branch in the commodity code tree. Ambiguity is anathema to the border systems: they do not like to have to make a value judgment on a choice of two or more duties and will simply reject any data that causes them to think. ME32 is a killer – beware.

| Step | Activity |
| --- | --- |
| 1 | End date all existing MFNs – described elsewhere |
| 2 | Ensure that the local version of the EU staging database is up to date with the latest incremental loads – needed to ensure that the subsequent steps use the latest EU data, which is critical |
| 3 | Generate an updated version of the list of authorised duties and store in the correct location |
| 4 | Generate an updated version of the complete commodity code list and store in the correct location |
| 5 | Receive a full, complete and verified version of the final UK tariff non-liberalised lines from the analyst team |
| 6 | Supply the file also to the HMRC CHIEF team for validation and data entry |
| 7 | Validate that the data within the file looks correct, for example:  Do all commodity codes specified exist?  Do all the duties match to the EU's duties? It may not be problematic if the duties are different from the EU's, however in the March / April Brexit, there were mistakes that were spotted by validating against the EU tariff  Are all duties expressed in a usable way – the duties are expressed in the 2 columns (ad valorem and specific) in a human-readable way – this human-readable content has to be migrated to the Taric 3 XML format, therefore a set style is mandated  Validating against these tasks is described in more detail below. |
| 8 | Copy the relevant columns to a new file, ensuring that the selected rows are just those where there is a need to retain a duty. We only need columns J, L and M |
| 9 | Split the Specific column into two columns entitled "Specific" and "Minimum". In splitting the column, split according to the following instructions:  only applies to rows where the term "MIN" is included in the existing "Specific" column cell – this was just two records in the March / April version of the UK tariff.  where "MIN" is included, copy any content **after the word MIN** to the Minimum column  Remove this content from the Specific column, also removing the term MIN |
| 10 | Save the file as a CSV in MS Excel in the specified location and with the specified filename.  Filename = mfns\_nov\_19.csv  Folder = /source (relative to executable script) |
| 11 | Ensure that the latest envelope ID is set correctly – *needs completing* |
| 12 | Run the executable script:  Run the script **python3 analyse\_mfns.py** on a Mac  Run the script **py analyse\_mfns.py** on a PC |
| 13 | This generates an extract file entitled "**uk\_mfn\_data.xml**" which can then be run through the standard data load process – not documented here |
| 14 | In addition, a second file is generated and placed in the /log subfolder which allows you to see what has been decided in the generation of the XML file (XMLs are very hard to read), so this file makes it simpler. The file verify\_mfn\_data.csv contains the following fields:  10-digit commodity code  Product line suffix  Leaf (1 or 0)  Measure type ID – 103 or 105  Action to take. The options are as follows:  **Applied** – where a specific tariff duty has been assigned to a commodity code in the tree, having been explicitly derived from the supplied Excel spreadsheet  **Barred** – where the product line suffix is not 80, therefore no measures can be assigned  **Liberalise** – where a 0% duty is to be assigned  **Protect** – where, in the process of applying an explicit duty or a liberalised duty, the process to avoid ME32 is run, in which the script runs up and down each branch to "protect" commodity codes against having duties assigned: this would contravene ME32.  Ad valorem duty  Specific duty  Minimum duty  "Nice" description with indented hyphens, for debugging purposes  What this file is intended to show is that there is no ME32 conflict and that the duties have been assigned correctly as defined in the supplied UK duty spreadsheet  If this file is correct, it does not mean that the XML data is correct though – it all needs to be tested thoroughly. |

# Anatomy of the application

This application is a one-off, expressly developed to support the creation of MFN data. MFN data differs from all other data, inasmuch as it requires complete coverage of the entire commodity tree. All other measure types just require that data is applied explicitly to the specific commodity codes that are deserving of the data.

Therefore, the code has not been productionised – it is intended to be throwaway once Brexit is out of the way.

The application performs the following tasks:

| Task | Description |
| --- | --- |
| Environment setup | **application object instantiation**  Determine all the necessary paths, filename and database connectivity – run via the instantiation of the application object |
| Get minimum SIDs | **Function get\_minimum\_sids**  Works out the highest used SID for various objects in the database – it is critical that any new data that is produced off-system does not use the same SIDs (unique, auto-incrementing IDs) as any existing objects: if they do, then the data will fail.  This process uses two yardsticks to determine what SID to use.  first, we range our SIDs so that they do not conflict with what the EU has already produced or may produce prior to EU Exit. These are typically set some significant way higher than the EU's data. These are set in the file config\_common.json JSON file  we then also do a check in the database to ensure that we are not overwriting any files that we have already created off-system in the UK |
| Get XML templates | **Function get\_templates**  In constructing XML output files, the data creation process uses templates with placeholders in place instead of the actual values - these placeholders are replaced with actual data once the script is run, however the templates are initially loaded into memory. There are three templates used, as follows:  **envelope.xml** – the wrapper to the entire file  **measure.xml** – repeated for each measure: one measure per MFN duty  **measure.component.xml** – repeated once or multiple times to cater for each of the components that are assigned to each of the measures (for example where there are specific or minimum duties, there may be multiple components assigned to a measure) |
| Read in the list of commodities that need authorised use measures | **Function get\_measure\_type\_105**  Import the list of commodity codes that are to be assigned authorised use measures instead of third country duty measures  Reads from file **measure\_type\_105.csv** |
| Read in the complete nomenclature list | **Function get\_nomenclature**  Read in the full list of up to date commodity codes  Reads from file **complete\_nomenclature.csv**  Work out the parent / child relationship between each of the commodity codes |
| Get a list of the UK MFN duties | **Function get\_uk\_mfns**  Get a full list of the UK MFNs  Read the file mfns\_nov\_19.csv  Apply the appropriate measures to the commodity codes as required |
| Apply protection against ME32 to related commodities | **Function protect\_commodities**  For all commodities where a duty has been assigned, go up and down the tree and "protect" other commodities that exist within that tree, to ensure that there are no ME32 errors caused.  This just assigns a status of "Protect", so that the subsequent function, which then applies a liberalised duty (0%) to commodities that have not had their duties retained, are exempted. |
| Liberalise any commodities that are left over | **Function create\_liberalised\_duties**  The whole purpose of this application to:  apply the required duties to the commodity codes as required  apply liberalisation to other commodities whose duties are not retained  ensure full coverage across the entire tariff  not cause ME32 conflicts  This is of considerable impact on this function, which takes c. 3-10 minutes to run, depending on the machine on which this script is run.  The script does the following:  Starting at the top of the tree (0100 00 00 00), look down the list for any commodities that have a status of "", i.e.  they have not yet had an explicit duty applied  they have not had a protect status applied due to being in a tree where a duty has been explicitly applied  they are lower than HS heading in the hierarchy  they have a product line suffix of 80  If there is a commodity that has a status if "", then liberalise it, by applying a duty of 0%  Then in all cases, look back up and down the tree and "Protect" any commodities that that are in that branch  At the end of this process, there should be no commodities left over with a status of "" |
| Document progress | **Function write\_measure\_count**  Write to the console the count of measures of different statuses |
| Get the SIDs of all of the commodity codes against which duties have been assigned | **Function get\_nomenclature\_sids**  Go back to the EU database and find the latest SIDs for all commodities for which measures are to be written.  The SID is required in measure nodes as well as the item\_id for goods nomenclature |
| Perform ME 32 checks | **Function perform\_me32\_checks**  This is optional |
| Write the XML | **Function write\_xml**  For each of the measures that need to get produced, write a node for measure and associated one or more nodes for measure\_components |
| Validate the XML | **Function validate**  sdfd |
| Write the verification CSV | **Function write\_verification\_csv**  Write verification CSV |