Calculations Taxes

This page describes how we calculate the tax amounts (import duties / excise / VAT etc).

Source of Products & Measures data

The data needed to calculate the tax amounts is refreshed every night using the following API for each unique taric code that we created in the CMS system. The code that uses this data to calculate the taxes is available in our package cu_tax_calculation.

Example API call to Bzctrl:

Gets the data for 2 different products, including that of their parents:

https://acc.bzctrl.com/bzctrl-core-api/api/v1/taric/export-data?includeParents=true&goodsNomenclatureCodes=2710124510&goodsNomenclatureCodes=3803009000

We always need the data of the parent also because any measures etc on the parent are inherited by the childs.

The response is a whole datamodel including codelists, footnotes, goodsnomenclatures, geograpahical areas, measures etc.

The input data

The input for the calculation is:

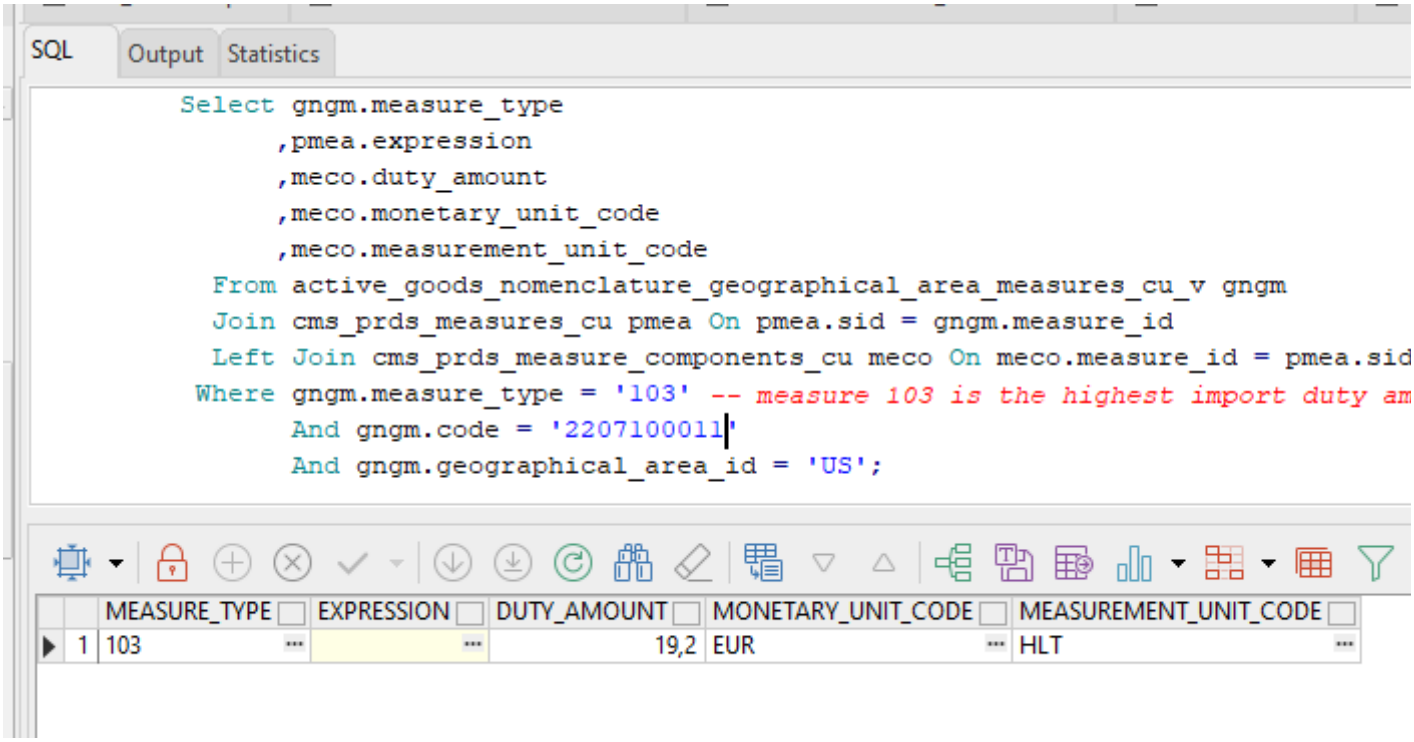
- Taric code
- Origin
- Customs Value
- Quantity in kilo's
- Quantity in Liters at 15 degrees C
- Quantity in Liters at 20 degrees C

- optional: alcohol percentage (ASV) for our ethanol products

Calculation of Import duties

Based on the taric code and origin, we now always get the measure 103 (third country duties). For this measure we get the expression and the measure components:

Then based on the measurement_unit_code that we find, we fetch the correct quantity from the input. So in this example its HLT(hectoliters). Customs does not specify if HLT is at 15 degrees C or at 20 degrees C. In this case we grab the liters at 20 degrees C from the input by default (if not specified by customs for the unit) and divide it by 100.



This is the logic for getting the correct quantity from the input:

```
l_measurement_unit_code := upper(i_measurement_unit_code);

-- For some volume units from customs it is not specified at which temperature they
-- Using the preference we can choose which one to pick in that case
If i_volume_at_unspecified_temperature_preference = cu_cms_tax_calculation.k_volume_
Then
   l_volume_at_unspecified_temperature_quantity := coalesce(i_quantity_l20
                                                           ,i_quantity_l15);
Elsif i_volume_at_unspecified_temperature_preference = cu_cms_tax_calculation.k_volu
Then
   l_volume_at_unspecified_temperature_quantity := coalesce(i_quantity_l15
                                                           ,i_quantity_l20);
Else
   l_volume_at_unspecified_temperature_quantity := coalesce(i_quantity_l20
                                                           ,i_quantity_l15);
End If;

If l_measurement_unit_code = k_codelist_value_l20
Then
   l_quantity := i_quantity_l20;
Elsif l_measurement_unit_code = k_codelist_value_l15
Then
   l_quantity := i_quantity_l15;
Elsif l_measurement_unit_code = k_codelist_value_ton
Then
   l_quantity := i_quantity_kgm / 1000;
Elsif l_measurement_unit_code = k_codelist_value_kgm
Then
   l_quantity := i_quantity_kgm;
Elsif l_measurement_unit_code = k_codelist_value_htl
Then
   l_quantity := l_volume_at_unspecified_temperature_quantity / 100; -- Customs does
Elsif l_measurement_unit_code = k_codelist_value_ltr
Then
   l_quantity := l_volume_at_unspecified_temperature_quantity; -- Customs does not s
Elsif l_measurement_unit_code = k_codelist_value_mtq
Then
   l_quantity := l_volume_at_unspecified_temperature_quantity / 1000; -- Customs doe
Elsif l_measurement_unit_code = k_codelist_value_asv
Then
   l_quantity := coalesce(i_quantity_asv
                         ,k_default_asv_value);
End If;
```

If there is a measurement unit like HLT then we multiply the quantity specified in this unit with the duty amount.

If there is no measurement unit like HLT then we multiply the Customs Value with the duty amount percentage.

```
-- Determine which quantity to use
l_quantity := fn_get_correct_quantity_for_customs_uom(i_measurement_unit_code =>
                                                     ,i_quantity_kgm          =>
                                                     ,i_quantity_l15          =>
                                                     ,i_quantity_l20          =>
                                                     ,i_quantity_asv          =>

-- Import duties can be calculated by:
--   A percentage of the monetary value of the product
--   A fixed amount of euros per quantity of product
If i_measurement_unit_code Is Not Null
   And l_quantity Is Not Null
   And i_duty_amount Is Not Null
Then
   -- Import duties are a fixed amount per quantity
   l_import_duties_amount := l_quantity * i_duty_amount;
Elsif i_customs_value Is Not Null
      And i_duty_amount Is Not Null
Then
   -- Import duties are a percentage of the monetary value of the product
   l_import_duties_amount := i_customs_value * (i_duty_amount / 100);
End If;

l_import_duties_amount := coalesce(l_import_duties_amount
                                  ,0);
```

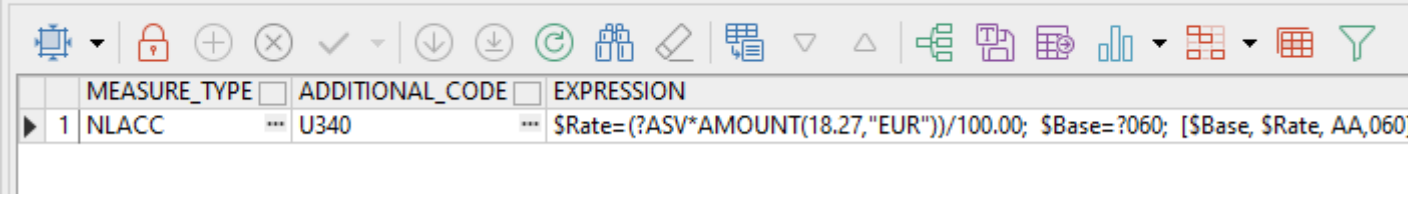
Calculation of Excise and Stock Tax

For excise and stock tax we calculate it in a similar way, but we get the measure based on origin and taric code and type=NLACC for excise or type=NLACVH for stock tax and the additional code like U340. This U-code is set by the system if there is only 1 available for the taric code, but if there are more than one U-codes the user has to set the correct one for the taric code in our product setup.

The difference here is that the expression can be filled.

SQL Output Statistics

```
Select gngm.measure_type
      ,mtad.additional_code
      ,pmea.expression
      ,meco.duty_amount
      ,meco.monetary_unit_code
      ,meco.measurement_unit_code
  From active_goods_nomenclature_geographical_area_measures_cu_v gngm
  Join active_measure_type_additional_codes_cu_v mtad On mtad.measure_id = gngm.measure
  Join cms_prds_measures_cu pmea On pmea.sid = gngm.measure_id
  Left Join cms_prds_measure_components_cu meco On meco.measure_id = pmea.sid
 Where gngm.measure_type = 'NLACC'
       And mtad.additional_code = 'U340'
       And gngm.code = '2207100011'
       And gngm.geographical_area_id = 'US';
```



	MEASURE_TYPE	ADDITIONAL_CODE	EXPRESSION
1	NLACC	U340	$Rate=(?ASV*AMOUNT(18.27,"EUR"))/100.00; $Base=?060; [$Base, $Rate, AA,060

The expression can be something like ?$Rate=(?ASV*AMOUNT(18.27,"EUR"))/100.00; $Base=?060; [$Base, $Rate, AA,060]?

To dynamically be able to calculate the tax based on the expression (we can now handle all the different expressions) we first extract the base unit of measure:

```
-- example expressions found in customs product database:
-- '$Rate=(?ASV*AMOUNT(18.27,"EUR"))/100.00;  $Base=?060;
-- '$BASE1=?063;  $RATE1=AMOUNT(789.10,"EUR")/1000.00;  [$BASE1,$RATE1,ED,063]'
If i_expression Is Not Null
Then
   -- Split the expression on ; and [] and pick the string with %BASE% in it, this w
   -- examples:
   -- '[$Base, $Rate, AA,060]' -> $Base
   -- '[$BASE1,$RATE1,ED,063]' -> $BASE1
   Select Replace(Replace(Trim(str2.column_value)
                          ,'[')
                  ,']') As base_string
     Into l_base_string
     From Table(apex_string.split((Select Trim(str.column_value)
                                     From Table(apex_string.split(i_expression
                                                                  ,';')) str
                                    Where Trim(str.column_value) Like '[%'
                                          And Trim(str.column_value) Like '%]')
                                  ,',')) str2
    Where upper(str2.column_value) Like '%BASE%';

   -- Split the expression on ; and ? and pick the string with the base string in it
   -- examples:
   -- '$BASE1=?063' -> 063
   -- '$Base=?060' -> 060
   Select Trim(str2.column_value) As base_string
     Into l_uom_string
     From Table(apex_string.split((Select Trim(str.column_value)
                                     From Table(apex_string.split(i_expression
                                                                  ,';')) str
                                    Where Trim(str.column_value) Like '%' || l_base_s
                                  ,'?')) str2
    Where str2.column_value Not Like '%' || l_base_string || '%';

End If;
```

Also this is not always the same format, sometime its &BASE1, sometimes its $Base etc. The unit can be 063 or 060 for example, which we then map on Liters at 15 degrees C or Liters at 20 degrees C from the input.

Then we transform the expression into a usable calculation:

```
-- example expressions found in customs product database:
-- '$Rate=(?ASV*AMOUNT(18.27,"EUR"))/100.00;  $Base=?060;  [$Base, $Rate, AA,060]'
-- '$BASE1=?063;  $RATE1=AMOUNT(789.10,"EUR")/1000.00;  [$BASE1,$RATE1,ED,063]'
If i_expression Is Not Null
Then
   -- Split the expression on ; and = and pick the string with AMOUNT in it, this wi
   -- examples:
   -- '(?ASV*AMOUNT(18.27,"EUR"))/100.00'
   -- 'AMOUNT(789.10,"EUR")/1000.00'
   Select upper(str2.column_value) As calculation_string
     Into l_calculation_string
     From Table(apex_string.split((Select str.column_value
                                     From Table(apex_string.split(i_expression
                                                                  ,';')) str
                                    Where upper(str.column_value) Like '%AMOUNT%')
                                  ,'=')) str2
    Where upper(str2.column_value) Like '%AMOUNT%';

   --set ASV
   -- example: '(99.8*AMOUNT(18.27,"EUR"))/100.00'
   l_calculation_string := Replace(l_calculation_string
                                  ,'?' || cu_cms_tax_calculation.k_codelist_value_as
                                  ,l_quantity_asv);

   -- remove Amount part
   -- exmaples:
   -- '(99.8*18.27)/100.00'
   -- '789.10/1000.00'
   l_calculation_string := Replace(l_calculation_string
                                  ,'AMOUNT(');
   l_calculation_string := Replace(l_calculation_string
                                  ,',"EUR")');

End If;
```

Then based on the calculation and the quantity we calculate the amount to be paid.

Excise is not always based on an expression, if its not we calculate it in the same way as the import duties (fetch the duty amount and multiply it by the quantity).

Calculation of VAT

For VAT we get the percentage based on the type=NLBTW and the additional code (U340 in this case) and the taric code and origin.

Then we just apply this percentage.

```
Select Max(meco.duty_amount) As duty_amount
  From active_goods_nomenclature_geographical_area_measures_cu_v gngm
  Left Join active_measure_type_additional_codes_cu_v mtad On mtad.measure_id = gngm.
  Join cms_prds_measures_cu pmea On pmea.sid = gngm.measure_id
  Left Join cms_prds_measure_components_cu meco On meco.measure_id = pmea.sid
 Where gngm.measure_type = 'NLBTW'
       And (mtad.additional_code = 'U340')
       And gngm.code = '2207100011'
       And gngm.geographical_area_id = 'US';
```

	DUTY_AMOUNT
1	21

Calculation of Obligation Guarantee

For the calculation of the obligation guarantee we combine the calculated Import duties, Excise, Stock tax and VAT like so:

```
-- The obligation guarantee is the sum of:
--  Import Duties
--  Excise
--  Stock tax
--  VAT

l_customs_value := coalesce(i_customs_value
                           ,0);

l_import_duty_amount := fn_calculate_import_duties(i_taric_code           => i_taric_c
                                                  ,i_geographical_area_id => i_geograp
                                                  ,i_customs_value        => l_customs
                                                  ,i_quantity_kgm         => i_quantit
                                                  ,i_quantity_l15         => i_quantit
                                                  ,i_quantity_l20         => i_quantit
                                                  ,i_quantity_asv         => i_quantit

l_excise_amount := fn_calculate_excise(i_taric_code           => i_taric_code
                                      ,i_geographical_area_id => i_geographical_area_i
                                      ,i_quantity_kgm         => i_quantity_kgm
                                      ,i_quantity_l15         => i_quantity_l15
                                      ,i_quantity_l20         => i_quantity_l20
                                      ,i_quantity_asv         => i_quantity_asv);

l_stock_tax_amount := fn_calculate_stock_tax(i_taric_code           => i_taric_code
                                            ,i_geographical_area_id => i_geographical_
                                            ,i_quantity_kgm         => i_quantity_kgm
                                            ,i_quantity_l15         => i_quantity_l15
                                            ,i_quantity_l20         => i_quantity_l20
                                            ,i_quantity_asv         => i_quantity_asv)

l_vat_calculation_amount := l_customs_value + l_import_duty_amount + l_excise_amount;

l_vat_amount := fn_calculate_vat(i_taric_code           => i_taric_code
                                ,i_geographical_area_id => i_geographical_area_id
                                ,i_amount               => l_vat_calculation_amount);

l_obligation_guarantee_amount := round(l_import_duty_amount
                                      ,2) + round(l_excise_amount
                                                 ,2) + round(l_stock_tax_amount
                                                            ,2) + round(l_vat_amount
                                                                       ,2);
```

We calculate the vat based on the customs value + import duty amount + excise amount multiplied by the VAT percentage.

Then we round all the taxes on 2 decimals just like the customs website does and sum it all up.

Obligation guarantee = Import Duties + Excise + Stock tax + VAT

Revision #1
Created 2 April 2025 15:37:01 by Admin
Updated 5 September 2025 12:47:58 by Admin