Avalara AvaTax for Microsoft Dynamics NAV 2013 R2

Specific Code Change Document

Version 01
Revision date: 2/10/17
Product release: Microsoft NAV 7.10.35473.12.04

Making sales tax less taxing

Release: Microsoft NAV 7.10.35473.12.01

Details

Fixed the issue related Transaction Statistics / Release / Post operation, where transaction contains same jurisdiction level taxes for same line.

Changes in Avalara Tables

Table 14073504 AVA Detail

Changes in Version List:

New Value	AVANA7.08.12.01

Changes in Keys:

Old Value	Module,Document Type,Document No.,Document Line No.,Juris.Code
New Value	Module,Document Type,Document No.,Document Line No.,Juris.Code,Tax Rate

Table 14073507 AVA Detail History

Changes in Version List:

New Value	AVANA7.08.12.01

Changes in Keys:

Old Value	Module,Document Type,Document No.,Document Line No.,Juris.Code
New Value	Module,Document Type,Document No.,Document Line No.,Juris.Code,Tax Rate

Table 14073508 AVA Juris. History

Changes in Version List:

New Value	AVANA7.08.12.01

Add new field:

Enabled	Field No.	Field Name	Caption	Data Type	Length	Description
Yes	54	Juris.Code	Juris.Code	Text	30	

Changes in field Caption:

Enabled	Field No.	Field Name	Caption	Data Type	Length	Description
Yes	6	Tax Description	Tax Description	Text	250	

Changes in Keys:

Old Value	Module,Document Type,Document No.,Document Line No.,Tax Name
New Value	Module,Document Type,Document No.,Document Line No.,Tax Name,Juris.Code

Table 14073509 AVA Juris.

Changes in Version List:

New Value	AVANA7.08.12.01

Add new field:

Enabled	Field No.	Field Name	Caption	Data Type	Length	Description
Yes	54	Juris.Code	Juris.Code	Text	30	

Changes in field Caption:

Enabled	Field No.	Field Name	Caption	Data Type	Length	Description
Yes	6	Tax Description	Tax Description	Text	250	

Changes in Keys:

Old Value	Module,Document Type,Document No.,Document Line No.,Tax Name
New Value	Module,Document Type,Document No.,Document Line No.,Tax Name,Juris.Code

Changes in Avalara CodeUnits

Codeunit 14073301 AVA Tax Engine

Changes in Version List:

New Value	AVANA7.08.12.01

Changes in function→ fnObjToTbl

Search for below first line and add colored line

```
AvaDtl.SETRANGE("Document Line No.",AvaLine."Document Line No.");
AvaDtl.SETRANGE("Juris.Code",oTaxLine.TaxDetails.Item(j).JurisCode);
AvaDtl.SETRANGE("Tax Rate",oTaxLine.TaxDetails.Item(j).Rate); //AVANA7.08.12.01
IF NOT AvaDtl.FINDFIRST THEN BEGIN
 AvaDtl.INIT;
 AvaDtl.Module := AvaLine.Module;
 AvaDtl."Document Type" := DocumentType;
 AvaDtl."Document No." := oResult.DocCode;
 AvaDtl."Document Line No." := AvaLine."Document Line No.";
 AvaDtl."Tax Base":=oTaxLine.TaxDetails.Item(j).Base*DCoefficient;
 AvaDtl."Tax Exempt Amount" := oTaxLine.TaxDetails.Item(j).Exemption*DCoefficient;
 AvaDtl."Tax Rate" := oTaxLine.TaxDetails.Item(j).Rate;
 AvaDtl."Tax Tax Amount" := oTaxLine.TaxDetails.Item(j).Tax*DCoefficient;
 AvaDtl.Taxability := oTaxLine.TaxDetails.Item(j).Taxable*DCoefficient;
 AvaDtl."Tax Type" := oTaxLine.TaxDetails.Item(j).TaxType;
 AvaDtl."Non Taxable" := oTaxLine.TaxDetails.Item(j).NonTaxable;
 AvaDtl."Juris.Code" := oTaxLine.TaxDetails.Item(j).JurisCode;
 AvaDtl."Juris.Name" := oTaxLine.TaxDetails.Item(j).JurisName;
 JurisTypeTxt := UPPERCASE(FORMAT(oTaxLine.TaxDetails.Item(j).JurisType));
 CASE JurisTypeTxt OF
   'STATE'   : AvaDtl."Juris.Type" := AvaDtl."Juris.Type"::STATE;
   'COUNTY'  : AvaDtl."Juris.Type" := AvaDtl."Juris.Type"::COUNTY;
   'CITY'    : AvaDtl."Juris.Type" := AvaDtl."Juris.Type"::CITY;
   'SPECIAL' : AvaDtl."Juris.Type" := AvaDtl."Juris.Type"::SPECIAL;
 END;
 AvaDtl."Tax Name" := oTaxLine.TaxDetails.Item(j).TaxName;
 TmpCityTax+=oTaxLine.TaxDetails.Item(j).Rate;
 AvaDtl.INSERT;
END
ELSE BEGIN
 AvaDtl."Tax Exempt Amount" := oTaxLine.TaxDetails.Item(j).Exemption*DCoefficient;
 AvaDtl."Tax Rate" := oTaxLine.TaxDetails.Item(j).Rate;
 //AvaDtl."Tax Tax Amount" := oTaxLine.TaxDetails.Item(j).Tax*DCoefficient; //AVANA7.08.12.01
 AvaDtl."Tax Tax Amount" += oTaxLine.TaxDetails.Item(j).Tax*DCoefficient; //AVANA7.08.12.01
 AvaDtl.Taxability := oTaxLine.TaxDetails.Item(j).Taxable*DCoefficient;
 AvaDtl."Tax Type" := oTaxLine.TaxDetails.Item(j).TaxType;
 AvaDtl."Non Taxable" := oTaxLine.TaxDetails.Item(j).NonTaxable;
 AvaDtl."Juris.Name" := oTaxLine.TaxDetails.Item(j).JurisName;
 JurisTypeTxt := UPPERCASE(FORMAT(oTaxLine.TaxDetails.Item(j).JurisType));
 CASE JurisTypeTxt OF
   'STATE'   : AvaDtl."Juris.Type" := AvaDtl."Juris.Type"::STATE;
   'COUNTY'  : AvaDtl."Juris.Type" := AvaDtl."Juris.Type"::COUNTY;
   'CITY'    : AvaDtl."Juris.Type" := AvaDtl."Juris.Type"::CITY;
   'SPECIAL' : AvaDtl."Juris.Type" := AvaDtl."Juris.Type"::SPECIAL;
 END;
```

```
      AvaDtl."Tax Name" := oTaxLine.TaxDetails.Item(j).TaxName;
      TmpCityTax+=oTaxLine.TaxDetails.Item(j).Rate;
      AvaDtl.MODIFY;
     END;

     AvaJurs.RESET;
     AvaJurs.SETRANGE(Module,AvaLine.Module);
     AvaJurs.SETRANGE("Document Type",DocumentType);
     AvaJurs.SETRANGE("Document No.",oResult.DocCode);
     AvaJurs.SETRANGE("Tax Name",oTaxLine.TaxDetails.Item(j).TaxName);
     //AvaJurs.SETRANGE("Tax Description",oTaxLine.TaxDetails.Item(j).JurisName); //AVANA7.08.12.01
     AvaJurs.SETRANGE("Juris.Code",oTaxLine.TaxDetails.Item(j).JurisCode); //AVANA7.08.12.01
     IF NOT AvaJurs.FINDFIRST THEN BEGIN
      AvaJurs.INIT;
      AvaJurs.Module := AvaLine.Module;
      AvaJurs."Document Type"  := AvaLine."Document Type";
      AvaJurs."Document No." := AvaLine."Document No.";
      AvaJurs."Document Line No." := oResult.TaxLines.Item(i).No;
      AvaJurs."Tax Name" := oTaxLine.TaxDetails.Item(j).TaxName;
      AvaJurs."Juris.Code" :=  oTaxLine.TaxDetails.Item(j).JurisCode; //AVANA7.08.12.01
      AvaJurs."Tax Description" := oTaxLine.TaxDetails.Item(j).JurisName;
      AvaJurs."Tax Name Total" := oTaxLine.TaxDetails.Item(j).Tax*DCoefficient;
      AvaJurs."Tax Rate" := oTaxLine.TaxDetails.Item(j).Rate;
      AvaJurs.INSERT;
     END ELSE BEGIN
      AvaJurs."Tax Name Total" := AvaJurs."Tax Name Total" + oTaxLine.TaxDetails.Item(j).Tax*DCoefficient;
      AvaJurs.MODIFY;
```

Changes in function→ fnObjToHistTbl

Search for below first line and add colored line

```
    AvaJurs.SETRANGE("Document No.",fAvaHead."Document No.");
   IF AvaJurs.FINDFIRST THEN BEGIN
    FOR i := 1 TO AvaJurs.COUNT DO BEGIN
     WITH AvaJursHIst DO BEGIN
      INIT;
      Module := fAvaHead.Module;
      "Document Type" := fAvaHead."Document Type";
      "Document No." := NewDocCode;
      "Document Line No." := AvaJurs."Document Line No.";
      "Tax Name"  := AvaJurs."Tax Name";
      "Juris.Code" := AvaJurs."Juris.Code"; //AVANA7.08.12.01
      "Tax Description" := AvaJurs."Tax Description";
      "Tax Name Total" := AvaJurs."Tax Name Total";
      "Tax Rate" := AvaJurs."Tax Rate";
      INSERT;
     END;
     AvaJurs.NEXT;
```

Release: Microsoft NAV 7.10.35473.12.02

Details

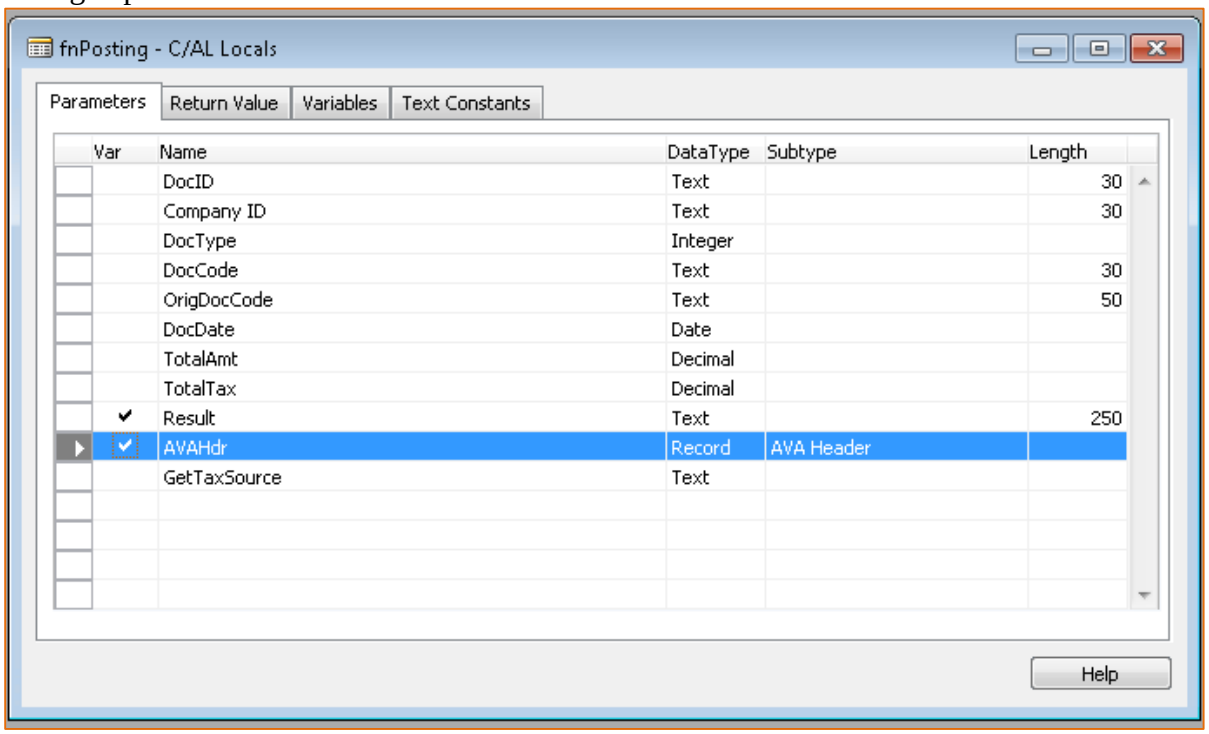
Fixed the issue related to deletion of all records from Ava header after posting single transaction.
Note: Developer needs to import 12.01 before importing 12.02 objects or below changes can be manually aaplied on 12.00 object from design mode.

Changes in Avalara codeunit

OBJECT Codeunit 14073301 AVA Tax Engine

Changes in parameter function fnPosting:

Changed parameter for AVAHdr



Changes in function → fnCommiting

Changes highlighted in green color

```
fnCommiting(DocId : Text[30];"Company ID" : Text[30];DocType : Integer;Doc
fnCommitTax(DocId,"Company ID",DocType,DocCode,iResult,oMessages,GetTaxSou

//AVANA7.08.12.02 start
//IF iResult = 0 THEN BEGIN
//  IF AvaHead.FINDFIRST THEN
//    fnObjToHistTbl(AvaHead,DocCode);
//END;
//AVANA7.08.12.02 end
AvaHead.RESET;

AvaValidCall.fnConnectorTraceLog(0,2,DocCode,Text15073212,GetTaxSource,0,6

fnCanceling(DocId : Text[30];"Company ID" : Text[30];DocType : Integer;Can
fnCancelTax(DocId,"Company ID",DocType,DocCode,CancelType,iResult,oMessage
```

Release: Microsoft NAV 7.10.35473.12.03

Details

Fixed an issue where the Outstanding Amount and the Amount Including Tax for a transaction were incorrect even when the sales tax was properly calculated

Changes in Avalara codeunit

OBJECT Codeunit 14073301 AVA Tax Engine

Create new function→ UpdateAmountIncTaxSales

Parameters:

Var	Name	DataType	Subtype	Length
Yes	SalesHeader	Record	Sales Header	

Variables:

Name	DataType	Subtype	Length
AVALine	Record	AVA Line	
RecCurrency	Record	Currency	
PrepLineAmt	Decimal		
SalesLine	Record	Sales Line	
AmountInclVAT	Decimal		
BaseAmountExVAT	Decimal		
BaseAmountIncVAT	Decimal		
AmountInclVATPerQty	Decimal		
OutstandingAmountIncVAT	Decimal		
CompleteAmountIncVAT	Decimal		
TaxExemptionAmount	Decimal		

```
    IF Avaconfig.GET AND (NOT Avaconfig."Disable Tax Calc.") AND Avaconfig.Initiate AND (SalesHeader."Tax Area Code" =
COPYSTR(FORMAT(SalesHeader."Tax Area Code"),1,6)) THEN BEGIN
      IF SalesHeader."Currency Code" = '' THEN
        RecCurrency.InitRoundingPrecision
      ELSE BEGIN
        SalesHeader.TESTFIELD("Currency Factor");
        RecCurrency.GET(SalesHeader."Currency Code");
        RecCurrency.TESTFIELD("Amount Rounding Precision");
      END;

      SalesLine.RESET;
      SalesLine.SETRANGE("Document Type",SalesHeader."Document Type") ;
      SalesLine.SETRANGE("Document No.",SalesHeader."No.");
      SalesLine.SETFILTER(Type,'<>%1',SalesLine.Type::" ");
      SalesLine.SETFILTER("Qty. to Invoice",'>=0');
      IF SalesLine.FINDFIRST THEN BEGIN
        REPEAT
          IF SalesLine."Document Type" IN [SalesLine."Document Type"::"Return Order",SalesLine."Document Type"::"Credit
Memo"] THEN BEGIN
            SalesLine."Outstanding Quantity" := SalesLine.Quantity - SalesLine."Return Qty. Received";
            SalesLine."Outstanding Qty. (Base)" := SalesLine."Quantity (Base)" - SalesLine."Return Qty. Received (Base)";
            SalesLine."Return Qty. Rcd. Not Invd." := SalesLine."Return Qty. Received" - SalesLine."Quantity Invoiced";
```

```
          SalesLine."Ret. Qty. Rcd. Not Invd.(Base)" := SalesLine."Return Qty. Received (Base)" - SalesLine."Qty. Invoiced
(Base)";
        END ELSE BEGIN
          SalesLine."Outstanding Quantity" := SalesLine.Quantity - SalesLine."Quantity Shipped";
          SalesLine."Outstanding Qty. (Base)" := SalesLine."Quantity (Base)" - SalesLine."Qty. Shipped (Base)";
          SalesLine."Qty. Shipped Not Invoiced" := SalesLine."Quantity Shipped" - SalesLine."Quantity Invoiced";
          SalesLine."Qty. Shipped Not Invd. (Base)" := SalesLine."Qty. Shipped (Base)" - SalesLine."Qty. Invoiced (Base)";
        END;
        SalesLine."Completely Shipped" := (SalesLine.Quantity <> 0) AND (SalesLine."Outstanding Quantity" = 0);

        AVALine.RESET;
        AVALine.SETRANGE(Module,AVALine.Module::Sales);
        AVALine.SETRANGE("Document Type",SalesLine."Document Type");
        AVALine.SETRANGE("Document No.",SalesLine."Document No.");
        AVALine.SETRANGE("Document Line No.",FORMAT(SalesLine."Line No."));
        AVALine.SETRANGE("Item No.",FORMAT(SalesLine."No."));
        IF AVALine.FINDFIRST THEN BEGIN
          IF AVALine."Total Taxable Amount" <> 0 THEN
            //SalesLine."VAT %" := AvaLine."Tax Rate" * 100;
             SalesLine."VAT %" := (AVALine."Tax Tax Amount" / AVALine."Total Taxable Amount") * 100;

          BaseAmountExVAT := SalesLine."Line Amount" - SalesLine."Inv. Discount Amount";
          BaseAmountIncVAT := BaseAmountExVAT + ((BaseAmountExVAT - AVALine."Tax Exempt Amount") *
(SalesLine."VAT %" / 100));

          IF (BaseAmountIncVAT <> 0) THEN
            SalesLine."Amount Including VAT" := BaseAmountIncVAT;

          SalesLine."Prepmt. Line Amount" := ROUND((SalesLine."Prepayment %" * SalesLine."Line Amount") /
100,RecCurrency."Amount Rounding Precision");
          PrepLineAmt := AVALine."Total Taxable Amount" + AVALine."Tax Discount Amount" + AVALine."Tax Exempt
Amount";
          PrepLineAmt += PrepLineAmt * AVALine."Tax Rate" ;
          IF SalesHeader."Prepmt. Include Tax" THEN
            SalesLine."Prepmt. Amt. Incl. VAT" := ROUND((SalesLine."Prepayment %" * PrepLineAmt) /
100,RecCurrency."Amount Rounding Precision")
          ELSE
            SalesLine."Prepmt. Amt. Incl. VAT" := SalesLine."Prepmt. Line Amount";
          SalesLine."Prepayment Amount" := SalesLine."Prepmt. Amt. Incl. VAT";

          IF SalesLine.Quantity <> 0 THEN
            OutstandingAmountIncVAT := ROUND((BaseAmountIncVAT * SalesLine."Outstanding Quantity") /
SalesLine.Quantity,RecCurrency."Amount Rounding Precision");
          SalesLine.VALIDATE("Outstanding Amount",OutstandingAmountIncVAT);
          SalesLine.MODIFY;
        END
        ELSE BEGIN
          IF SalesLine.Quantity <> 0 THEN BEGIN
            BaseAmountExVAT := SalesLine."Line Amount" - SalesLine."Inv. Discount Amount";
            BaseAmountIncVAT := BaseAmountExVAT + (BaseAmountExVAT * (SalesLine."VAT %" / 100));
            AmountInclVATPerQty := ROUND((BaseAmountIncVAT /SalesLine.Quantity),RecCurrency."Amount Rounding
Precision");
            OutstandingAmountIncVAT := AmountInclVATPerQty * SalesLine."Outstanding Quantity";
            SalesLine.VALIDATE("Outstanding Amount",OutstandingAmountIncVAT);
            CompleteAmountIncVAT := AmountInclVATPerQty * SalesLine.Quantity;
            SalesLine."Amount Including VAT" := CompleteAmountIncVAT;
```

```
          SalesLine.MODIFY;
        END;
      END;
    UNTIL SalesLine.NEXT = 0;
  END;
END;
```

Create new function→ UpdateAmountIncTaxService

Parameters:

Var	Name	DataType	Subtype	Length
Yes	ServiceHeader	Record	Service Header	

Variables:

Name	DataType	Subtype	Length
AVALine	Record	AVA Line	
RecCurrency	Record	Currency	
PrepLineAmt	Decimal		
ServiceLine	Record	Service Line	
AmountInclVAT	Decimal		
BaseAmountExVAT	Decimal		
BaseAmountIncVAT	Decimal		
AmountInclVATPerQty	Decimal		
OutstandingAmountIncVAT	Decimal		
CompleteAmountIncVAT	Decimal		

```
  IF Avaconfig.GET AND (NOT Avaconfig."Disable Tax Calc.") AND Avaconfig.Initiate AND (ServiceHeader."Tax Area Code"
= COPYSTR(FORMAT(ServiceHeader."Tax Area Code"),1,6)) THEN BEGIN
    IF ServiceHeader."Currency Code" = '' THEN
      RecCurrency.InitRoundingPrecision
    ELSE BEGIN
      ServiceHeader.TESTFIELD("Currency Factor");
      RecCurrency.GET(ServiceHeader."Currency Code");
      RecCurrency.TESTFIELD("Amount Rounding Precision");
    END;

    ServiceLine.RESET;
    ServiceLine.SETRANGE("Document Type",ServiceHeader."Document Type") ;
    ServiceLine.SETRANGE("Document No.",ServiceHeader."No.");
    ServiceLine.SETFILTER(Type,'<>%1',ServiceLine.Type::" ");
    ServiceLine.SETFILTER("Qty. to Invoice",'>=0');
    IF ServiceLine.FINDFIRST THEN BEGIN
      REPEAT
        IF ServiceLine."Document Type" = ServiceLine."Document Type"::"Credit Memo" THEN BEGIN
          ServiceLine."Outstanding Quantity" := ServiceLine.Quantity;
          ServiceLine."Outstanding Qty. (Base)" := ServiceLine."Quantity (Base)";
        END ELSE BEGIN
          ServiceLine."Outstanding Quantity" := ServiceLine.Quantity - ServiceLine."Quantity Shipped";
          ServiceLine."Outstanding Qty. (Base)" := ServiceLine."Quantity (Base)" - ServiceLine."Qty. Shipped (Base)";
          ServiceLine."Qty. Shipped Not Invoiced" := ServiceLine."Quantity Shipped" - ServiceLine."Quantity Invoiced" -
ServiceLine."Quantity Consumed";
```

```
          ServiceLine."Qty. Shipped Not Invd. (Base)" := ServiceLine."Qty. Shipped (Base)" - ServiceLine."Qty. Invoiced (Base)"
- ServiceLine."Qty. Consumed (Base)";
        END;
        ServiceLine."Completely Shipped" := (ServiceLine.Quantity <> 0) AND (ServiceLine."Outstanding Quantity" = 0);

        AVALine.RESET;
        AVALine.SETRANGE(Module,AVALine.Module::Service);
        AVALine.SETRANGE("Document Type",ServiceLine."Document Type");
        AVALine.SETRANGE("Document No.",ServiceLine."Document No.");
        AVALine.SETRANGE("Document Line No.",FORMAT(ServiceLine."Line No."));
        AVALine.SETRANGE("Item No.",FORMAT(ServiceLine."No."));
        IF AVALine.FINDFIRST THEN BEGIN
          IF AVALine."Total Taxable Amount" <> 0 THEN
            //ServiceLine."VAT %" := AVALine."Tax Rate" * 100;
            ServiceLine."VAT %" := (AVALine."Tax Tax Amount" / AVALine."Total Taxable Amount") * 100;


          BaseAmountExVAT := ServiceLine."Line Amount" - ServiceLine."Inv. Discount Amount";
          BaseAmountIncVAT := BaseAmountExVAT + ((BaseAmountExVAT - AVALine."Tax Exempt Amount") *
(ServiceLine."VAT %" / 100));
          IF (BaseAmountIncVAT <> 0) THEN
            ServiceLine."Amount Including VAT" := BaseAmountIncVAT;
          IF ServiceLine.Quantity <> 0 THEN
          OutstandingAmountIncVAT := ROUND((BaseAmountIncVAT * ServiceLine."Outstanding Quantity") /
ServiceLine.Quantity,RecCurrency."Amount Rounding Precision");
          ServiceLine.VALIDATE("Outstanding Amount",OutstandingAmountIncVAT);
          ServiceLine.MODIFY;
        END
        ELSE BEGIN
          IF ServiceLine.Quantity <> 0 THEN BEGIN
            BaseAmountExVAT := ServiceLine."Line Amount" - ServiceLine."Inv. Discount Amount";
            BaseAmountIncVAT := BaseAmountExVAT + (BaseAmountExVAT * (ServiceLine."VAT %" / 100));
            AmountInclVATPerQty := ROUND((BaseAmountIncVAT /ServiceLine.Quantity),RecCurrency."Amount Rounding
Precision");
            OutstandingAmountIncVAT := AmountInclVATPerQty * ServiceLine."Outstanding Quantity";
            ServiceLine.VALIDATE("Outstanding Amount",OutstandingAmountIncVAT);
            CompleteAmountIncVAT := AmountInclVATPerQty * ServiceLine.Quantity;
            ServiceLine."Amount Including VAT" := CompleteAmountIncVAT;
            ServiceLine.MODIFY;
          END;
        END;
      UNTIL ServiceLine.NEXT = 0;
    END;
  END;
```

Change to function→ fnCalcTax

Search for below first line and add colored line

```
                IF AvaCallGetTax = FALSE THEN IF (AvalLineDiffAmtInv <> 0) THEN AvaCallGetTax:= TRUE;
                 IF AvaCallGetTax = FALSE THEN BEGIN
                 IF ((AvaLine."Tax Tax Amount" + AvaLine."Total Taxable Amount" + AvaLine."Tax Exempt Amount") <> 0) THEN
                  SalesLine."Amount Including VAT" := AvaLine."Tax Tax Amount" + AvaLine."Total Taxable Amount" +
AvaLine."Tax Exempt Amount";
                  IF (SalesLine."Prepayment %" > 0) AND (AvaLine."Tax Tax Amount" > 0) THEN
                    //SalesLine."VAT %" := AvaLine."Tax Rate" * 100;
```

```
                IF AvaLine."Total Taxable Amount" <> 0 THEN
                SalesLine."VAT %" := (AvaLine."Tax Tax Amount" / AvaLine."Total Taxable Amount") * 100;
              SalesLine."Prepmt. Line Amount" := ROUND((SalesLine."Prepayment %" * SalesLine."Line Amount") /
100,RecCurrency."Amount Rounding Precision");

              //PrepLineAmt := AvaLine."Total Taxable Amount" + AvaLine."Tax Tax Amount" + AvaLine."Tax Exempt
Amount";
              PrepLineAmt := (AvaLine."Total Taxable Amount" + AvaLine."Tax Discount Amount" + AvaLine."Tax Exempt
Amount");
```

Search for below first line and add colored line

```
        ERROR(Text14073209 + Text14073204);
      END;
    END;
    AvaWindowOpen(StrMessage+Text14073223);
    AvaModule := AvaModule::Sales;

//fnObjToTbl(oGetTaxRequest,oGetTaxResult,oGetTaxResult.DocCode,TDocType,DT2DATE(oGetTaxResult.DocDate),Docum
//entType,AvaDocumentNo,GetTaxSource,AvaModule);
fnObjToTbl(oGetTaxRequest,oGetTaxResult,oGetTaxResult.DocCode,TDocType,DT2DATE(oGetTaxResult.DocDate),Docum//
entType,AvaDocumentNo,GetTaxSource,AvaModule,DocTypeSetToInv);

    IF NOT DocTypeSetToInv THEN
      fnQtyAmtOrd(SalesHead,SalesLine,FALSE)
    ELSE
      fnQtyAmtOrd(SalesHead,SalesLine,TRUE);

    UpdateAmountIncTaxSales(SalesHead);

AvaValidCall.fnConnectorTraceLog(0,2,SalesHeaderNo,Text15073210,GetTaxSource,SalesLine.COUNT,2,FunctionName,1);

    IF NOT Avaconfig."Show Message Dialog" THEN
      Window.CLOSE
    ELSE

    AvaWindowClose;
    fnClear;
```

Change to function→ fnCalcTaxBatch

Search for below first line and add colored line

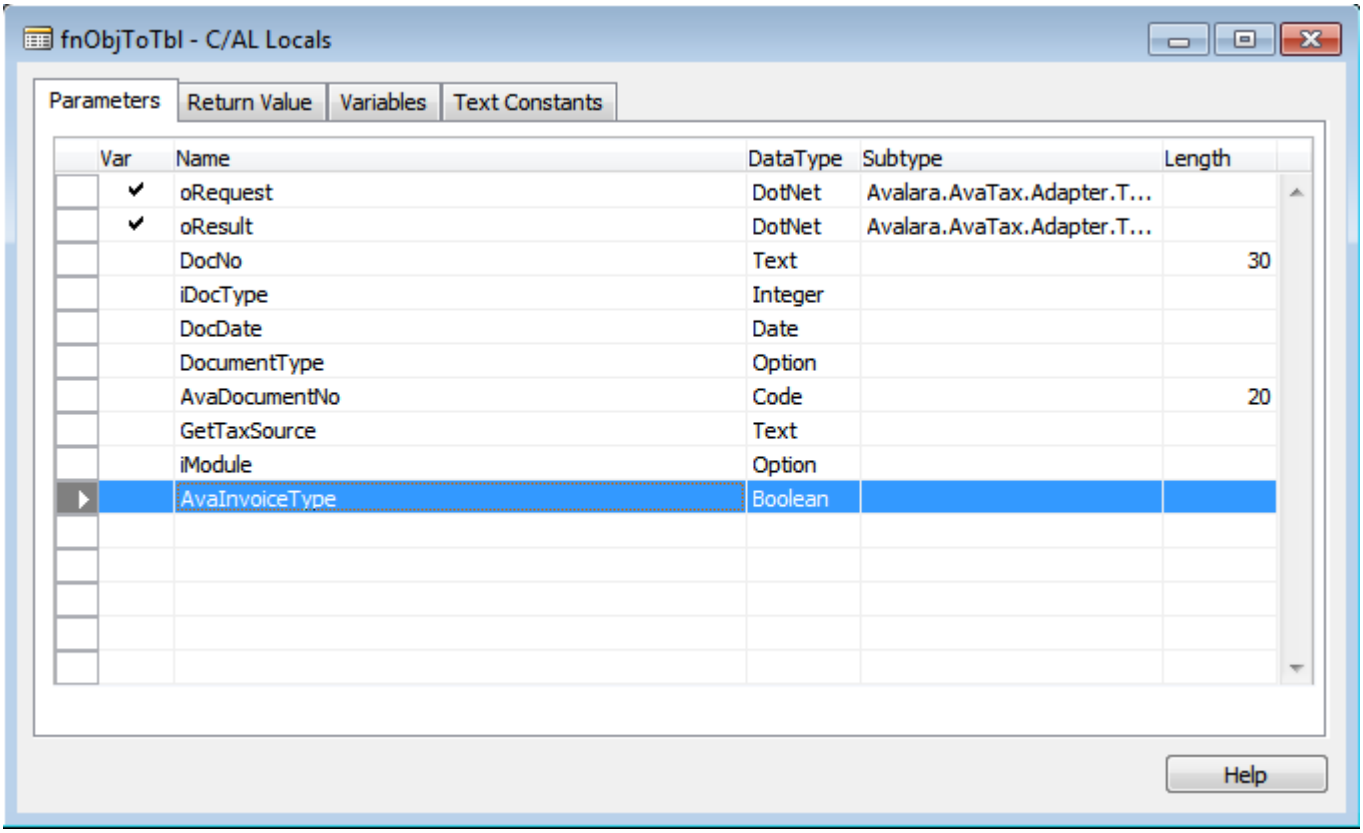
```
              AvalLineDiffAmtInv := (AvaLine."Total Taxable Amount" + AvaLine."Tax Exempt Amount" + AvaLine."Tax
Discount Amount")-
                                    ((SalesLine."Line Amount"*SalesLine."Qty. to Invoice")/SalesLine.Quantity);
            IF AvaCallGetTax = FALSE THEN IF (AvalLineDiffAmtInv <> 0) THEN AvaCallGetTax:= TRUE;
             IF AvaCallGetTax = FALSE THEN BEGIN
              IF ((AvaLine."Tax Tax Amount" + AvaLine."Total Taxable Amount" + AvaLine."Tax Exempt Amount") <> 0)
THEN
              SalesLine."Amount Including VAT" := AvaLine."Tax Tax Amount" + AvaLine."Total Taxable Amount" +
AvaLine."Tax Exempt Amount";
              IF (SalesLine."Prepayment %" > 0) AND (AvaLine."Tax Tax Amount" > 0) THEN
               //SalesLine."VAT %" := AvaLine."Tax Rate" * 100;
               IF AvaLine."Total Taxable Amount" <> 0 THEN
               SalesLine."VAT %" := (AvaLine."Tax Tax Amount" / AvaLine."Total Taxable Amount") * 100;
```

```
                SalesLine."Prepmt. Line Amount" := ROUND((SalesLine."Prepayment %" * SalesLine."Line Amount") /
100,RecCurrency."Amount Rounding Precision");

                PrepLineAmt := AvaLine."Total Taxable Amount" + AvaLine."Tax Tax Amount" + AvaLine."Tax Exempt
Amount";
                IF SalesOrderHdr."Prepmt. Include Tax" THEN
```

Search for below first line and add colored line

```
          AVABatchTaxFill(SalesHead,tResultMessage,Status);
         EXIT(FALSE);
       END;
    END;

    AvaWindowOpen(StrMessage+Text14073223);
    AvaModule := AvaModule::Sales;

//fnObjToTbl(oGetTaxRequest,oGetTaxResult,oGetTaxResult.DocCode,TDocType,DT2DATE(oGetTaxResult.DocDate),Docum
//entType,AvaDocumentNo,GetTaxSource,AvaModule);

fnObjToTbl(oGetTaxRequest,oGetTaxResult,oGetTaxResult.DocCode,TDocType,DT2DATE(oGetTaxResult.DocDate),Docume
ntType,AvaDocumentNo,GetTaxSource,AvaModule,FALSE);
    fnQtyAmtOrd(SalesHead,SalesLine,FALSE);

    UpdateAmountIncTaxSales(SalesHead);

AvaValidCall.fnConnectorTraceLog(0,2,SalesHeaderNo,Text15073210,GetTaxSource,SalesLine.COUNT,2,FunctionName,1);
     IF NOT Avaconfig."Show Message Dialog" THEN
```

Change to function→ fnObjToTbl
Add Parameters:



Search for below first line and add colored line

```
            RecCurrency.GET(SalesOrderHdr."Currency Code");
            RecCurrency.TESTFIELD("Amount Rounding Precision");
          END;

          //SalesLine."VAT %" := AvaLine."Tax Rate" * 100;
          IF AvaLine."Total Taxable Amount" <> 0 THEN
          SalesLine."VAT %" := (AvaLine."Tax Tax Amount" / AvaLine."Total Taxable Amount") * 100;
          SalesLine."Prepmt. Line Amount" := ROUND((SalesLine."Prepayment %" * SalesLine."Line Amount") /
100,RecCurrency."Amount Rounding Precision");

          PrepLineAmt := (AvaLine."Total Taxable Amount" + AvaLine."Tax Discount Amount" + AvaLine."Tax Exempt
Amount");
          IF AvaLine."Tax Tax Amount" <> 0 THEN
          PrepLineAmt += (AvaLine."Total Taxable Amount" + AvaLine."Tax Discount Amount" + AvaLine."Tax Exempt
Amount")* AvaLine."Tax Rate" ;
          IF SalesOrderHdr."Prepmt. Include Tax" THEN
            SalesLine."Prepmt. Amt. Incl. VAT" := ROUND((SalesLine."Prepayment %" * PrepLineAmt) /
100,RecCurrency."Amount Rounding Precision")
          ELSE
            SalesLine."Prepmt. Amt. Incl. VAT" := SalesLine."Prepmt. Line Amount";
          SalesLine."Prepayment Amount" := SalesLine."Prepmt. Amt. Incl. VAT";
    //IF SalesLine."Document Type" IN [SalesLine."Document Type"::"Return Order",SalesLine."Document Type"::"Credit
//Memo"] THEN BEGIN
  //          SalesLine."Outstanding Quantity" := SalesLine.Quantity - SalesLine."Return Qty. Received";
  //          SalesLine."Outstanding Qty. (Base)" := SalesLine."Quantity (Base)" - SalesLine."Return Qty. Received (Base)";
  //          SalesLine."Return Qty. Rcd. Not Invd." := SalesLine."Return Qty. Received" - SalesLine."Quantity Invoiced";
  //          SalesLine."Ret. Qty. Rcd. Not Invd.(Base)" := SalesLine."Return Qty. Received (Base)" - SalesLine."Qty. Invoiced
//(Base)";
```

```
  //          END ELSE BEGIN
   //           SalesLine."Outstanding Quantity" := SalesLine.Quantity - SalesLine."Quantity Shipped";
   //           SalesLine."Outstanding Qty. (Base)" := SalesLine."Quantity (Base)" - SalesLine."Qty. Shipped (Base)";
    //          SalesLine."Qty. Shipped Not Invoiced" := SalesLine."Quantity Shipped" - SalesLine."Quantity Invoiced";
    //          SalesLine."Qty. Shipped Not Invd. (Base)" := SalesLine."Qty. Shipped (Base)" - SalesLine."Qty. Invoiced (Base)";
    //          END;
     //         SalesLine."Completely Shipped" := (SalesLine.Quantity <> 0) AND (SalesLine."Outstanding Quantity" = 0);
    //          IF SalesLine.Quantity > 0 THEN
    //          AmountInclVAT := ROUND( (SalesLine."Line Amount" + (SalesLine."Line Amount" * AvaLine."Tax Rate")) *
//SalesLine."Outstanding Quantity" / SalesLine.Quantity
  //                        ,RecCurrency."Amount Rounding Precision");
  //          SalesLine.VALIDATE("Outstanding Amount",AmountInclVAT);
            SalesLine.MODIFY;
          END ELSE BEGIN
            SalesInvLine.SETRANGE("Document No.",AvaHead."Document No.");
            SalesInvLine.SETFILTER("Line No.",oTaxLine.No);
            IF SalesInvLine.FINDFIRST THEN
              AvaLine."Tax Area" := SalesInvLine."Tax Area Code"
            ELSE BEGIN
              recObjPostedCrLine.SETRANGE("Document No.",AvaHead."Document No.");
              recObjPostedCrLine.SETFILTER("Line No.",oTaxLine.No);
              IF recObjPostedCrLine.FINDFIRST THEN
                AvaLine."Tax Area" := recObjPostedCrLine."Tax Area Code"
            END;
          END;
        END ELSE BEGIN
          ServiceLine.SETRANGE("Document No.",AvaHead."Document No.");
          ServiceLine.SETRANGE("Document Type",ServiceHeader."Document Type");
          ServiceLine.SETFILTER("Line No.",oTaxLine.No);
          IF  ServiceLine.FINDFIRST THEN BEGIN
            AvaLine."Tax Area" := ServiceLine."Tax Area Code";
            IF (AvaLine.Quantity <> 0)  THEN
            ServiceLine."Amount Including VAT" := AvaLine."Tax Tax Amount" + AvaLine."Total Taxable Amount" + AvaLine."Tax
Exempt Amount";
            IF ServiceHeader."Currency Code" = '' THEN
              RecCurrency.InitRoundingPrecision
            ELSE BEGIN
              ServiceHeader.TESTFIELD("Currency Factor");
              RecCurrency.GET(ServiceHeader."Currency Code");
              RecCurrency.TESTFIELD("Amount Rounding Precision");
            END;
            //ServiceLine."VAT %" := AvaLine."Tax Rate" * 100;
              IF AvaLine."Total Taxable Amount" <> 0 THEN
              ServiceLine."VAT %" := (AvaLine."Tax Tax Amount" / AvaLine."Total Taxable Amount") * 100;

            //IF ServiceLine."Document Type" = ServiceLine."Document Type"::"Credit Memo" THEN BEGIN
             // ServiceLine."Outstanding Quantity" := ServiceLine.Quantity;
             // ServiceLine."Outstanding Qty. (Base)" := ServiceLine."Quantity (Base)";
            //END ELSE BEGIN
            //  ServiceLine."Outstanding Quantity" := ServiceLine.Quantity - ServiceLine."Quantity Shipped";
             // ServiceLine."Outstanding Qty. (Base)" := ServiceLine."Quantity (Base)" - ServiceLine."Qty. Shipped (Base)";
             // ServiceLine."Qty. Shipped Not Invoiced" := ServiceLine."Quantity Shipped" - ServiceLine."Quantity Invoiced" -
//ServiceLine."Quantity Consumed";
  //          ServiceLine."Qty. Shipped Not Invd. (Base)" := ServiceLine."Qty. Shipped (Base)" - ServiceLine."Qty. Invoiced
//(Base)" - ServiceLine."Qty. Consumed (Base)";
  //          END;
```

```
  //        ServiceLine."Completely Shipped" := (ServiceLine.Quantity <> 0) AND (ServiceLine."Outstanding Quantity" = 0);
  //        AmountInclVAT := ROUND( (ServiceLine."Line Amount" + (ServiceLine."Line Amount" * AvaLine."Tax Rate")) *
//ServiceLine."Outstanding Quantity" / ServiceLine.Quantity
//                        ,RecCurrency."Amount Rounding Precision");
  //        ServiceLine.VALIDATE("Outstanding Amount",AmountInclVAT);
          ServiceLine.MODIFY;
         END ELSE BEGIN
          ServInvLine.SETRANGE("Document No.",AvaHead."Document No.");
```

Change to function→ fnGetTaxHistory

Search for below first line and add colored line

```
     oMessages:=oGetTaxHistoryResult.Messages;
    Result:=oGetTaxHistoryResult.ResultCode;

    IF (Result = 0) OR (Result = 1) THEN BEGIN
     TMessages:='';

//fnObjToTbl(oGetTaxRequest,oGetTaxResult,oGetTaxResult.DocCode,DocType,DT2DATE(oGetTaxResult.DocDate),Docume
//ntType,AvaDocumentNo,GetTaxSource,iModule);

fnObjToTbl(oGetTaxRequest,oGetTaxResult,oGetTaxResult.DocCode,DocType,DT2DATE(oGetTaxResult.DocDate),Document
Type,AvaDocumentNo,GetTaxSource,iModule,FALSE);
      AvaHead.RESET;
      AvaHead.SETRANGE("Document Type",DocType);
      AvaHead.SETRANGE("Document No.",DocCode);
```

Change to function→ fnServiceCalcTax

Search for below first line and add colored line

```
          ERROR(Text14073209 + Text14073204);
        END;
      END;
      AvaWindowOpen(StrMessage+Text14073223);
      AvaModule := AvaModule::Service;

//fnObjToTbl(oGetTaxRequest,oGetTaxResult,oGetTaxResult.DocCode,TDocType,DT2DATE(oGetTaxResult.DocDate),Docum
//entType,AvaDocumentNo,GetTaxSource,AvaModule);

fnObjToTbl(oGetTaxRequest,oGetTaxResult,oGetTaxResult.DocCode,TDocType,DT2DATE(oGetTaxResult.DocDate),Docume
ntType,AvaDocumentNo,GetTaxSource,AvaModule,FALSE);
      fnServiceQtyAmtOrd(ServiceHead,ServiceLine,FALSE);
      UpdateAmountIncTaxService(ServiceHead);


AvaValidCall.fnConnectorTraceLog(0,2,ServiceHeaderNo,Text15073210,GetTaxSource,ServiceLine.COUNT,2,FunctionName,
1);

    IF NOT Avaconfig."Show Message Dialog" THEN
      Window.CLOSE
      ELSE
      AvaWindowClose;
      fnClear;
```

Change to function→ fnServiceCalcTax1

Search for below first line and add colored line

```
          ERROR(Text14073209 + Text14073204);
        END;
    END;
    AvaWindowOpen(StrMessage+Text14073223);
    AvaModule := AvaModule::Service;

//fnObjToTbl(oGetTaxRequest,oGetTaxResult,oGetTaxResult.DocCode,TDocType,DT2DATE(oGetTaxResult.DocDate),Docum
//entType,AvaDocumentNo,GetTaxSource,AvaModule);

fnObjToTbl(oGetTaxRequest,oGetTaxResult,oGetTaxResult.DocCode,TDocType,DT2DATE(oGetTaxResult.DocDate),Docume
ntType,AvaDocumentNo,GetTaxSource,AvaModule,FALSE);

    ServiceLine.RESET;
    ServiceLine.SETRANGE("Document Type",ServiceHeaderType);
    ServiceLine.SETRANGE("Document No.",ServiceHeaderNo);
    ServiceLine.SETFILTER(Type, '<>%1', ServiceLine.Type::" ");
    fnServiceQtyAmtOrd(ServiceHead,ServiceLine,TRUE);
    UpdateAmountIncTaxService(ServiceHead);
    AvaWindowClose;
    fnClear;
    AvaValidCall.fnConnectorTraceLog(0,2,ServiceHeaderNo,Text15073210,GetTaxSource,0,2,FunctionName,1);
```

Change to function→ fnServCalcTaxBatch

Search for below first line and add colored line

```
           Status  := 'Error';
          AVAServiceBatchTaxFill(ServiceHead,tResultMessage,Status);
          EXIT(FALSE);
        END;
      END;
      AvaWindowOpen(StrMessage+Text14073223);
      AvaModule := AvaModule::Service;

//fnObjToTbl(oGetTaxRequest,oGetTaxResult,oGetTaxResult.DocCode,TDocType,DT2DATE(oGetTaxResult.DocDate),Docum
//entType,AvaDocumentNo,GetTaxSource,AvaModule);

fnObjToTbl(oGetTaxRequest,oGetTaxResult,oGetTaxResult.DocCode,TDocType,DT2DATE(oGetTaxResult.DocDate),Docume
ntType,AvaDocumentNo,GetTaxSource,AvaModule,FALSE);
      fnServiceQtyAmtOrd(ServiceHead,ServiceLine,FALSE);
      UpdateAmountIncTaxService(ServiceHead);
    IF NOT Avaconfig."Show Message Dialog" THEN
      Window.CLOSE
      ELSE

      AvaWindowClose;
      Remarks := 'Success';
      Status  := 'Success';
      AVAServiceBatchTaxFill(ServiceHead,Remarks,Status);
      fnClear;
```

Release: Microsoft NAV 7.10.35473.12.04

Details

Fixed an issue where the sales tax amount was incorrectly displayed in native Statistics for transactions having identical Document No. For example, the sales tax calculated for Document No. 1000A was displayed in Document No. 1000.

Changes in Avalara codeunit

Codeunit 14073310 AVA Validate Call

Changes in function→ AvaFillTempAvaLines

Search for below first line and add colored line

```
IF NOT(AvaCheckInstStatus(0)) THEN
      EXIT(FALSE);

    "AvaLine 1".RESET;
    "AvaLine 1".DELETEALL;
    "AvaHead 1".RESET;
    "AvaHead 1".DELETEALL;

    CASE DocType OF
      0,1,2,4: BEGIN TDocType := 1; Coefficient:= 1; END;
      3,5: BEGIN TDocType := 5; Coefficient:= -1; END;
    END;

    //AVANA7.08.12.04 Start
    //AvaHead.RESET;
    //AvaHead.SETFILTER("Status Code",'<>%1','OK');
    //IF fModule = fModule::Sales THEN BEGIN
    //  AvaHead.SETRANGE(Module,AvaHead.Module::Sales);
    //  AvaHead.SETRANGE("Document Type",SalesHead."Document Type");
    //  AvaHead.SETFILTER("Document No.",SalesHead."No."+'*');
    //  AvaHead.SETFILTER("Document Id",SalesHead."No."+'*');
    //END;
    //IF fModule = fModule::Service THEN BEGIN
    //  AvaHead.SETRANGE(Module,AvaHead.Module::Service);
    //  AvaHead.SETRANGE("Document Type",ServiceHead."Document Type");
    //  AvaHead.SETFILTER("Document No.",ServiceHead."No."+'*');
    //  AvaHead.SETFILTER("Document Id",ServiceHead."No."+'*');
    //END;
    //
    //IF AvaHead.FINDLAST THEN BEGIN

    AvaHead.RESET;
    AvaHead.SETRANGE(Module,fModule);
    AvaHead.SETFILTER("Status Code",'<>%1','OK');
    IF fModule = fModule::Sales THEN BEGIN
      AvaHead.SETRANGE("Document Type",SalesHead."Document Type");
      IF AvaConfig.GET AND AvaConfig."Enable AsynchronousAvaTax Post" THEN BEGIN
        AvaHead.SETFILTER("Document No.",SalesHead."No."+'*');
        AvaHead.SETFILTER("Document Id",SalesHead."No."+'*');
      END ELSE BEGIN
        AvaHead.SETFILTER("Document No.",SalesHead."No.");
        AvaHead.SETFILTER("Document Id",SalesHead."No.");
```

```
  END;
 END
 ELSE IF fModule = fModule::Service THEN BEGIN
  AvaHead.SETRANGE("Document Type",ServiceHead."Document Type");
  IF AvaConfig.GET AND AvaConfig."Enable AsynchronousAvaTax Post" THEN BEGIN
    AvaHead.SETFILTER("Document No.",ServiceHead."No."+'*');
    AvaHead.SETFILTER("Document Id",ServiceHead."No."+'*');
  END ELSE BEGIN
    AvaHead.SETFILTER("Document No.",ServiceHead."No.");
    AvaHead.SETFILTER("Document Id",ServiceHead."No.");
  END;
 END;
 IF AvaHead.FINDFIRST THEN BEGIN
 //AVANA7.08.12.04 End
  "AvaHead 1".COPY(AvaHead);
  "AvaHead 1".INSERT;
  AvaLine.RESET;
  AvaLine.SETRANGE(Module,AvaHead.Module);
  AvaLine.SETRANGE("Document No.",AvaHead."Document No.");
  AvaLine.SETRANGE("Document Type",AvaHead."Document Type");
  IF AvaLine.FINDFIRST THEN
    REPEAT
      "AvaLine 1".COPY(AvaLine);
      "AvaLine 1".INSERT;
    UNTIL AvaLine.NEXT=0;
 END;
```