**Contents:**

[NHTSA VIN Decoding guidance document 2](#_Toc158564116)

[Example output: 2](#_Toc158564117)

[Steps for Decoding: 4](#_Toc158564118)

[Importing NHTSA’s Database into SQL Server Management Studio 5](#_Toc158564119)

[Creating a Stored Procedure to run Updated VIN Decoder 8](#_Toc158564120)

[2022 SQL Code 10](#_Toc158564121)

# NHTSA VIN Decoding guidance document

[Product Information Catalog and Vehicle Listing (vPIC) Analytical User's Manual 2020 (dot.gov)](https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/813252)

# Example output:

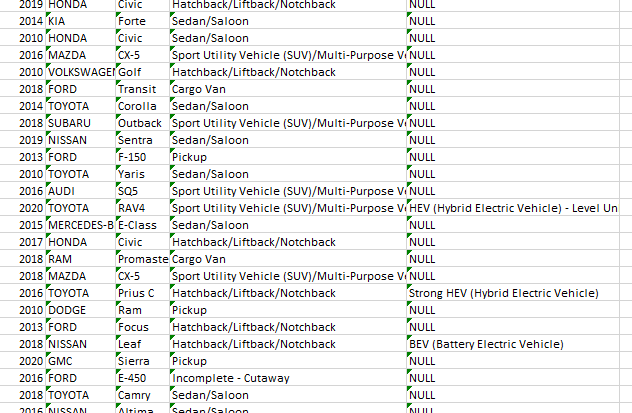
A screenshot of a computer

Description automatically generated

Data will always export as a 2 row x 8 column table. This allows sorting in excel

* In the Excel VIN Decode Form, Import (from text file) the SQL query file (the RPT file can be read as a text file). Use the blue highlighted cell in the form to import the data to
* Columns A-F will populate: VIN, Year, Make, Model, Body Style, EV Label (in this order)
* If error code #4 or #11 is associated with the VIN decode process from the SQL query, the cells should read "error".

Screen shot from a VIN decode form:



# Steps for Decoding:

* Use NHTSA’s standalone database for VIN decoding. This version works for the database from year 2022
  + Link to file for download:
* Download Microsoft SQL Server management studio
* Run the SQL script to create a new stored procedure.
* Use notepad ++ to copy 10,000 records at a time into Microsoft SQL Server Management Studio
  + Example:

A screenshot of a computer

Description automatically generated

* Type the command :

USE [vPICList\_lite]

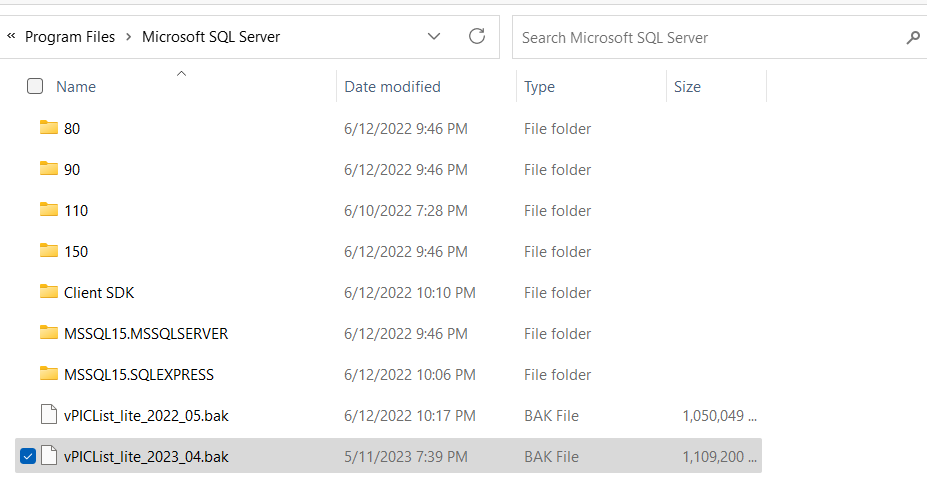
GO

In the query before the set of 10,000 VIN decode commands

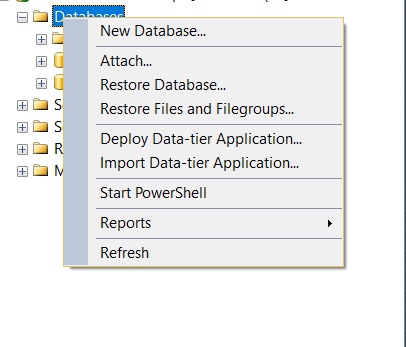
* The SQL stored procedure should be able to decode at a minimum rate of 1000 VINs/minute.
  + Process 10,000 VINs at a time to optimize speed
* In Microsoft SQL Server Management Studio, Query results will need to be set so that output is to a bar “ | “ delimited file. This prevents problems with columns that are comma delimited (the output text can contain commas which causes issues in the alignment of the data)

# Importing NHTSA’s Database into SQL Server Management Studio

After downloading NHTSA’s database, unzip the file and place in the C drive (Can place in Program Files --> Microsoft SQL Server)

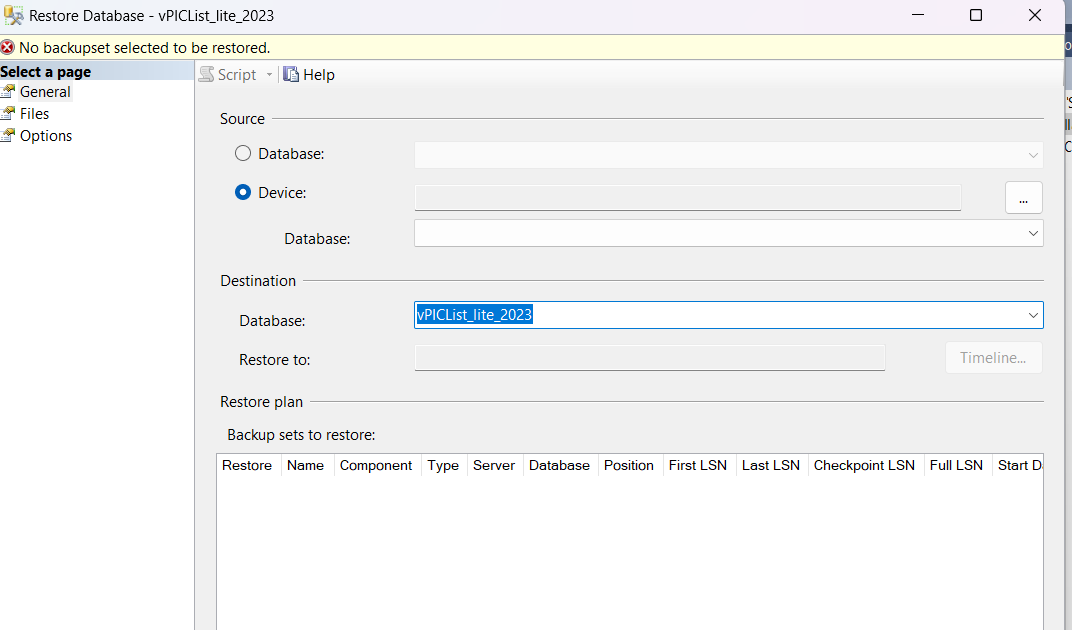


Open SQL Server Management Studio. Right click Databases, --> select Restore Database...

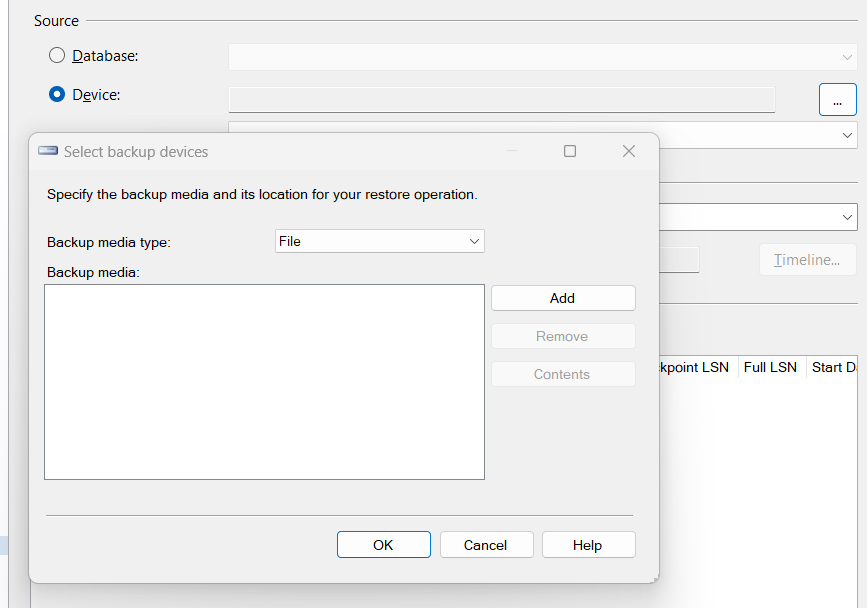


Under “Destination” name the database.

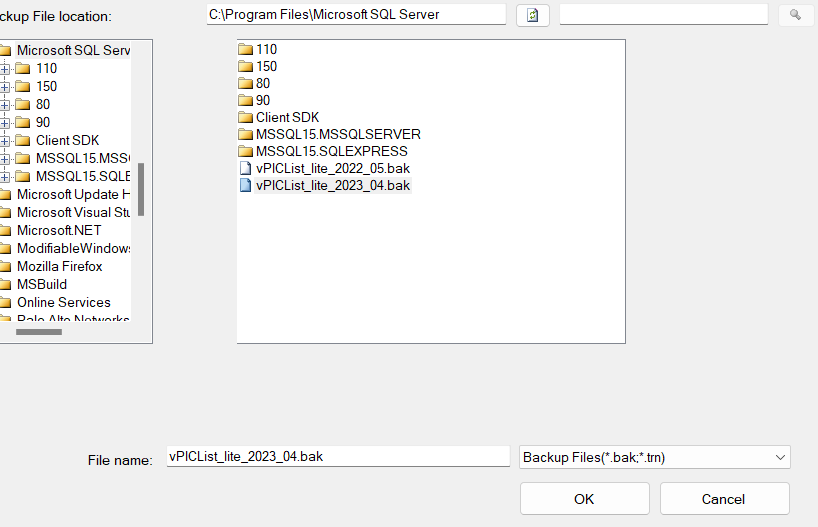
select device --> click 



Click Add

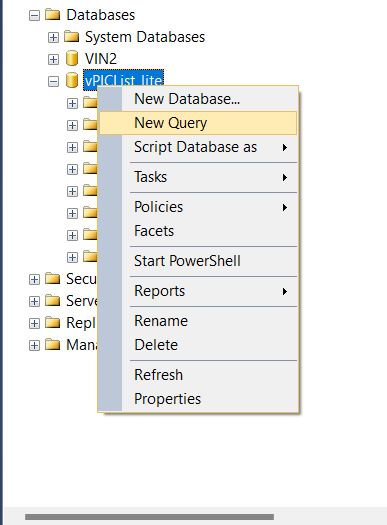


Locate the .bak file, select, click ok

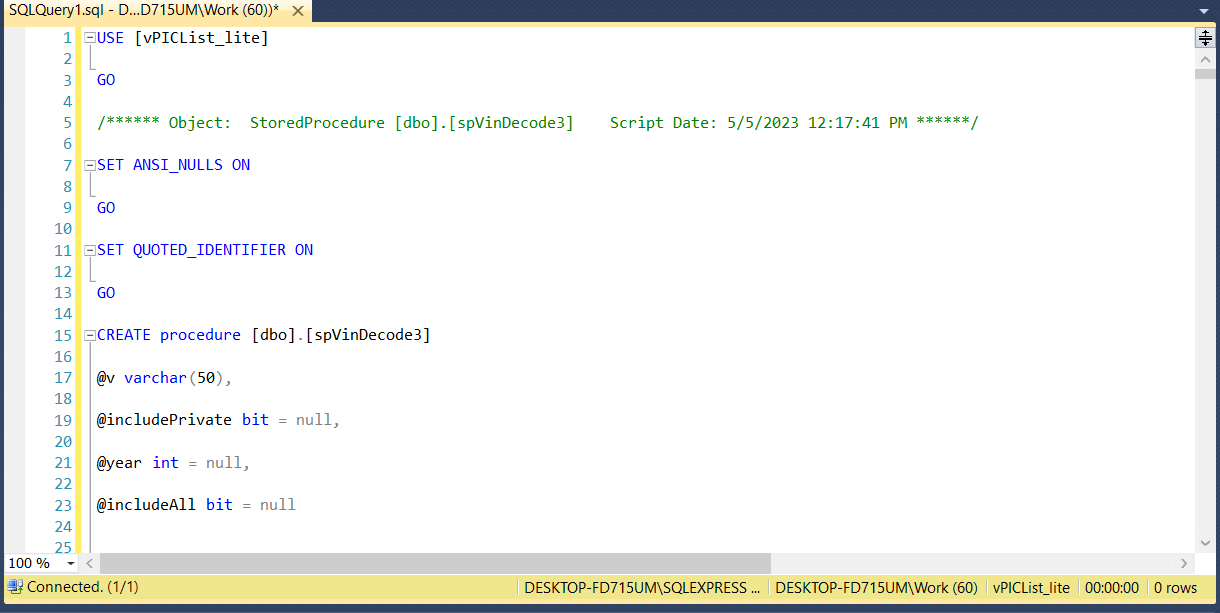


# Creating a Stored Procedure to run Updated VIN Decoder

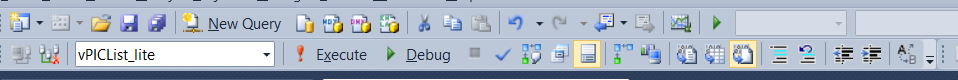
Click on NHTSA's database, select New Query



Paste the SQL Code into the Query tab that opens



Select Execute



# 2022 SQL Code

SQL script is copied below. This needs to be run in Microsoft SQL Server management Studio to create a new stored procedure that processes VINs properly. Use with the 2022 database from NHTSA (not compatible with the 2023 database)

USE [vPICList\_lite]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[spVinDecode3] Script Date: 5/5/2023 12:17:41 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE procedure [dbo].[spVinDecode3]

@v varchar(50),

@includePrivate bit = null,

@year int = null,

@includeAll bit = null

as

begin

SET NOCOUNT ON;

declare

@make varchar(50) = null,

@includeNotPublicilyAvailable bit = null,

@NoOutput bit = 0,

@vin varchar(17) = '',

@wmi varchar(6) = '',

@wmiId int,

@patternId int,

@vinSchemaId int,

@keys varchar(14) = '',

@modelYear int,

@formulaKeys nvarchar(14) = '',

@modelYearPos varchar(20) = '',

@conclusive bit = 0,

@v1 varchar(50) = @v -- adding a variable to store the original VIN For output

declare @ReturnCode varchar(100) = '', @CorrectedVIN varchar(17), @ErrorBytes varchar(500), @AdditionalDecodingInfo varchar(500), @UnUsedPositions varchar(500)

declare @doNotRetry bit = 0

set @vin = upper(LTRIM(RTRIM(@v)))

set @wmi = dbo.fVinWMI(@vin)

if @year is null

begin

declare @my int, @descr varchar(17)

select @my = ModelYear, @descr = Descriptor from VinDescriptor where Descriptor = dbo.fVinDescriptor(@vin )

if @my >= 1980

select @modelYear = @my, @conclusive = 1, @modelYearPos = @descr, @doNotRetry = 1

else

begin

select @modelYear = dbo.fVinModelYear2 (upper(@vin)), @conclusive = 1, @modelYearPos = '\*\*\*X\*|Y'

if @modelYear < 0

select @modelYear = -@modelYear, @conclusive = 0, @modelYearPos = '\*\*\*\*\*|Y'

end

end

else

begin

select @modelYear = @year, @conclusive = 1

end

if LEN(@vin) > 3

Begin

set @keys = SUBSTRING(@vin, 4, 5)

if LEN(@vin) > 9

set @keys = @keys + '|' + SUBSTRING(@vin, 10, 8)

end

declare @did int = 1

IF OBJECT\_ID('tempdb..#DecodingItem') IS NOT NULL

drop table #DecodingItem

CREATE TABLE #DecodingItem(

[Id] [bigint] IDENTITY(1,1) NOT NULL,

[DecodingId] [int] NOT NULL,

[CreatedOn] [datetime] NULL ,

[PatternId] [int] NULL,

[Keys] [varchar](50) NULL,

[VinSchemaId] [int] NULL,

[WmiId] [int] NULL,

[ElementId] [int] NULL,

[AttributeId] [varchar](500) NULL,

[Value] [varchar](500) NULL,

[Source] [varchar](50) NULL,

[Priority] [int] NULL,

[TobeQCed][bit] null

)

declare @pass int = 0;

start\_again:

set @pass = @pass + 1

select @wmiId = Id from Wmi where Wmi = @wmi and (@includeNotPublicilyAvailable = 1 or (PublicAvailabilityDate <= getdate()))

if @wmiid is null

begin

select @ReturnCode = @ReturnCode + ' 7 ', @CorrectedVIN = '', @ErrorBytes = ''

end

else

begin

INSERT INTO #DecodingItem ([DecodingId], [Source], [CreatedOn], [Priority], [PatternId], [Keys], [VinSchemaId], [WmiId], [ElementId], [AttributeId], [Value], TobeQCed)

SELECT

@did, 'Pattern', isnull(p.UpdatedOn, p.CreatedOn), wvs.YearFrom,

p.Id, upper(p.Keys), p.VinSchemaId, wvs.WmiId, p.ElementId,

p.AttributeId, dbo.fElementAttributeValue (p.ElementId, p.AttributeId) as Value, vs.TobeQCed

FROM

dbo.Pattern AS p

INNER JOIN dbo.Element E ON P.ElementId = E.Id

INNER JOIN dbo.VinSchema VS on p.VinSchemaId = vs.Id

INNER JOIN dbo.Wmi\_VinSchema AS wvs ON vs.Id = wvs.VinSchemaId and ((@modelYear is null) or (@modelYear between wvs.YearFrom and isnull(wvs.YearTo, 2999)))

INNER JOIN dbo.Wmi AS w ON wvs.WmiId = w.Id and w.Wmi = @wmi

WHERE

@keys like replace(p.Keys, '\*', '\_') + '%'

and not p.ElementId in (26, 27, 29, 39)

and not E.Decode is null

and (isnull(e.IsPrivate, 0) = 0 or @includePrivate = isnull(e.IsPrivate, 0))

and (@includeNotPublicilyAvailable = 1 or (w.PublicAvailabilityDate <= getdate()))

and (@includeNotPublicilyAvailable = 1 or (isnull(vs.TobeQCed, 0) = 0))

declare @EngineModel varchar(500), @k varchar(50)

select top 1 @EngineModel = attributeid, @patternId = PatternId, @vinSchemaId = VinSchemaId, @k = Keys

from #DecodingItem

where DecodingId = @did and ElementId = 18

order by [Priority] desc

if not @EngineModel is null

INSERT INTO #DecodingItem ([DecodingId], [Source], [CreatedOn], [Priority], [PatternId], [Keys], [VinSchemaId], [WmiId], [ElementId], [AttributeId], [Value])

SELECT

@did, 'EngineModelPattern', isnull(p.UpdatedOn, p.CreatedOn), 50,

@patternId, @k, @vinSchemaId, @wmiId, p.ElementId,

p.AttributeId, dbo.fElementAttributeValue (p.ElementId, p.AttributeId) as Value

FROM

EngineModel em

inner join dbo.EngineModelPattern AS p on em.Id = p.EngineModelId

INNER JOIN dbo.Element E ON P.ElementId = E.Id

WHERE

em.Name = @EngineModel

INSERT INTO #DecodingItem ([DecodingId], [Source], CreatedOn, [Priority], [PatternId], [Keys], [VinSchemaId], [WmiId], [ElementId], [AttributeId], [Value])

select

@did, 'VehType', isnull(w.UpdatedOn, w.CreatedOn), 100,

null, upper(@wmi) as keys , null, w.Id as WmiId, 39,

CAST(t.Id as varchar), upper(t.Name) as Value

from wmi w

join VehicleType t on t.Id = w.VehicleTypeId

where Wmi = @wmi

and (@includeNotPublicilyAvailable =1 or (w.PublicAvailabilityDate <= getdate()))

declare @MfrId int, @MfrName varchar(500)

select @MfrId = t.Id, @MfrName = upper(t.Name)

from wmi w

join Manufacturer t on t.Id = w.ManufacturerId

where Wmi = @wmi

and (@includeNotPublicilyAvailable =1 or (w.PublicAvailabilityDate <= getdate()))

INSERT INTO #DecodingItem ([DecodingId], [Source], [Priority], [PatternId], [Keys], [VinSchemaId], [WmiId], [ElementId], [AttributeId], [Value])

select @did, 'Manuf. Name', 100, null, upper(@wmi) as keys, null, @WmiId as WmiId, 27, CAST(@MfrId as varchar), @MfrName as Value

INSERT INTO #DecodingItem ([DecodingId], [Source], [Priority], [PatternId], [Keys], [VinSchemaId], [WmiId], [ElementId], [AttributeId], [Value])

select @did, 'Manuf. Id', 100, null, upper(@wmi) as keys, null, @WmiId AS wMIiD, 157, CAST(@MfrId as varchar), CAST(@MfrId as varchar)

INSERT INTO #DecodingItem ([DecodingId], [Source], [Priority], [PatternId], [Keys], [VinSchemaId], [WmiId], [ElementId], [AttributeId], [Value])

select

@did, 'ModelYear', 100,

null, @modelYearPos , null, null, 29,

CAST(@modelYear as varchar), CAST(@modelYear as varchar) as Value

where not @modelYear is null

set @formulaKeys = @keys

set @formulaKeys = replace(@formulaKeys,1,'#')

set @formulaKeys = replace(@formulaKeys,2,'#')

set @formulaKeys = replace(@formulaKeys,3,'#')

set @formulaKeys = replace(@formulaKeys,4,'#')

set @formulaKeys = replace(@formulaKeys,5,'#')

set @formulaKeys = replace(@formulaKeys,6,'#')

set @formulaKeys = replace(@formulaKeys,7,'#')

set @formulaKeys = replace(@formulaKeys,8,'#')

set @formulaKeys = replace(@formulaKeys,9,'#')

set @formulaKeys = replace(@formulaKeys,0,'#')

INSERT INTO #DecodingItem ([DecodingId], [Source], CreatedOn, [Priority], [PatternId], [Keys], [VinSchemaId], [WmiId], [ElementId], [AttributeId], [Value])

select

@did, 'Formula Pattern', isnull(p.UpdatedOn, p.CreatedOn), 100,

p.Id, p.Keys as Keys, p.VinSchemaId, null, p.ElementId,

p.AttributeId, SUBSTRING(@keys, CHARINDEX('#', p.keys), ((len(p.keys) - charindex('#', REVERSE(p.Keys)) + 1) - (CHARINDEX('#', p.keys)) + 1)) as value

FROM

dbo.Pattern AS p

INNER JOIN dbo.Element E ON P.ElementId = E.Id

WHERE

p.VinSchemaId in

(

SELECT wvs.VinSchemaId

FROM dbo.Wmi AS w

INNER JOIN dbo.Wmi\_VinSchema AS wvs ON w.Id = wvs.WmiId and ((@modelYear is null) or (@modelYear between wvs.YearFrom and isnull(wvs.YearTo, 2999)))

WHERE w.Wmi = @wmi and ((@modelYear is null) or (@modelYear between wvs.YearFrom and isnull(wvs.YearTo, 2999)))

and (@includeNotPublicilyAvailable =1 or (w.PublicAvailabilityDate <= getdate()))

)

and CHARINDEX('#', p.keys) > 0

and not p.ElementId in (26, 27, 29, 39)

and @formulaKeys like replace(p.Keys, '\*', '\_') + '%'

delete

from #DecodingItem

where Id IN

(

SELECT Id FROM

(

SELECT d.Id, RANK() OVER (PARTITION BY ElementId ORDER BY Priority DESC, createdon DESC, LEN(REPLACE(ISNULL(D.Keys, ''), '\*', '')), NEWID() desc) AS RankResult

FROM #DecodingItem D

WHERE D.ElementId NOT IN (121, 129, 150, 154, 155, 114, 169, 186)

) t WHERE t.RankResult > 1

)

declare @modelId int

select @modelId = attributeid from #DecodingItem where DecodingId = @did and ElementId = 28

if not @modelId is null

begin

INSERT INTO #DecodingItem ([DecodingId], [Source], [Priority], [PatternId], [Keys], [VinSchemaId], [WmiId], [ElementId], [AttributeId], [Value])

SELECT

@did, 'pattern - model', 1000,

di.PatternId, di.Keys, di.VinSchemaId, null as WmiId, 26 AS ElementId,

mk.Id AS AttributId, upper(mk.Name) AS Value

FROM

dbo.Make\_Model AS mm

INNER JOIN dbo.Make AS mk ON mm.MakeId = mk.Id

INNER JOIN #DecodingItem AS di ON mm.ModelId = di.AttributeId

WHERE

(di.ElementId = 28)

AND (di.DecodingId = @did)

end

else

begin

INSERT INTO #DecodingItem ([DecodingId], [Source], [CreatedOn], [Priority], [PatternId], [Keys], [VinSchemaId], [WmiId], [ElementId], [AttributeId], [Value])

select

@did, 'Make', isnull(w.UpdatedOn, w.CreatedOn), -100,

null, @wmi as keys , null, w.Id as WmiId, 26,

CAST(t.Id as varchar), upper(t.Name) as Value

from wmi w

join Wmi\_Make wm on wm.WmiId = w.Id

join Make t on t.Id = wm.MakeId

where Wmi = @wmi

and (@includeNotPublicilyAvailable =1 or (w.PublicAvailabilityDate <= getdate()))

end

exec spVinDecode\_Conversions @did

declare @tVehicleType int

select top 1 @tVehicleType = attributeid from #DecodingItem where DecodingId = @did and elementid = 39

declare @tmpPatterns table (id int, TobeQCed bit null)

declare @tmpPatternsEx table (id int, a int, b int)

insert into @tmpPatterns

select distinct sp.id, s.TobeQCed

from VehicleSpecSchema s

inner join VSpecSchemaPattern sp on s.id = sp.SchemaId

inner join VehicleSpecPattern p on sp.Id = p.VSpecSchemaPatternId

inner join VehicleSpecSchema\_Model vssm on vssm.VehicleSpecSchemaId = s.id

left outer join VehicleSpecSchema\_Year vssy on vssy.VehicleSpecSchemaId = s.id

inner join Wmi\_Make wm on wm.MakeId = s.makeid

inner join wmi on wmi.id = wm.WmiId

where 1 = 1

and wmi.wmi = @wmi

and s.VehicleTypeId = @tVehicleType

and vssm.ModelId = @modelId

and (vssy.Year = @modelYear or vssy.Id is null)

and p.IsKey=1

and (@includeNotPublicilyAvailable = 1 or (isnull(s.TobeQCed, 0) = 0))

insert into @tmpPatternsEx (id, a, b)

select

p.VSpecSchemaPatternId, count(\*) as cntTotal, count (distinct d.id) as cntMatch

from

VehicleSpecPattern p

inner join @tmpPatterns ptrn on p.VSpecSchemaPatternId = ptrn.id

left outer join #DecodingItem d on p.ElementId = d.ElementId and p.AttributeId = d.AttributeId

where

p.IsKey = 1

group by p.VSpecSchemaPatternId

having count(\*) <> count(distinct d.id)

delete from @tmpPatterns where id in (select id from @tmpPatternsEx)

declare @tbl1 table (

IsKey bit,

vSpecSchemaId int,

vSpecPatternId int,

ElementId int,

AttributeId varchar(500),

ChangedOn datetime null,

TobeQCed bit null

)

insert into @tbl1

(iskey, vSpecSchemaId, vSpecPatternId, ElementId, AttributeId, ChangedOn, TobeQCed)

SELECT distinct

vsp.IsKey, vsvp.SchemaId, vsp.vspecschemapatternid, vsp.ElementId, vsp.AttributeId, isnull(vsp.UpdatedOn, vsp.CreatedOn), ptrn.TobeQCed

FROM

VehicleSpecPattern vsp

inner join VSpecSchemaPattern vsvp on vsvp.id = vsp.vspecschemapatternid

inner join @tmpPatterns ptrn on vsvp.id = ptrn.id

WHERE

vsp.IsKey = 0

and vsp.ElementId not in (select elementid from #DecodingItem)

; WITH cte AS (

SELECT elementid,

row\_number() OVER(PARTITION BY elementid order by attributeid) AS [rn]

FROM @tbl1

)

DELETE cte WHERE [rn] > 1

INSERT INTO

#DecodingItem ([DecodingId], [Source], [CreatedOn], [Priority], [PatternId], [Keys], [VinSchemaId], [WmiId], [ElementId], [AttributeId], [Value], TobeQCed)

SELECT distinct

@did, 'Vehicle Specs', ChangedOn, -100, vSpecPatternId, '', vSpecSchemaId, null, ElementId, AttributeId, dbo.fElementAttributeValue(ElementId, AttributeId), TobeQCed

FROM

@tbl1

if (select COUNT(\*) from #DecodingItem where DecodingId = @did and not PatternId is null) = 0

begin

select @ReturnCode = @ReturnCode + ' 8 ', @CorrectedVIN = '', @ErrorBytes = ''

end

else

begin

exec spVinDecode\_ErrorCode @vin, @modelYear, @ReturnCode OUTPUT, @CorrectedVIN OUTPUT, @ErrorBytes OUTPUT, @UnUsedPositions OUTPUT

end

end

if exists(select \* from #DecodingItem where ElementId = 5 and AttributeId = 64 and DecodingId = @did)

begin

select @ReturnCode = @ReturnCode + ' 9 '

end

declare @isOffRoad bit = 0

if exists(select \* from #DecodingItem where ElementId = 5 and AttributeId in (69, 84, 86, 88, 97, 105, 113, 124, 126, 127) and DecodingId = @did)

begin

select @ReturnCode = @ReturnCode + ' 10 '

set @isOffRoad = 1

end

If (@modelYear is null)

begin

select @ReturnCode = @ReturnCode + ' 11 '

end

declare @vehicleType varchar(500) = (select AttributeId from #DecodingItem where ElementId = 39)

if @modelYear >= 2008

and @vehicleType in('2', '7')

and not exists(select 1 from #DecodingItem where ElementId = 168)

begin

INSERT INTO #DecodingItem ([DecodingId], [Source], [Priority], [PatternId], [Keys], [VinSchemaId], [WmiId], [ElementId], [AttributeId], [Value])

values (@did, 'code', 500, null, null, null, null, 168, 1, 'Direct')

end

DECLARE @invalidChars VARCHAR(500) = ''

DECLARE @startPos INT = 13

, @x\_vehicleTypeId INT, @x\_truckTypeId INT, @j INT = 0, @chr CHAR(10) = ''

, @isCarMpvLT bit = 0

IF SUBSTRING(@vin, 3, 1) = '9'

SET @startPos = 15

ELSE

begin

SELECT @x\_vehicleTypeId = vehicleTypeId, @x\_truckTypeId = truckTypeId FROM dbo.Wmi WHERE wmi = @wmi

IF @x\_vehicleTypeId IN (2, 7) OR (@x\_vehicleTypeId = 3 AND @x\_truckTypeId = 1)

select @startPos = 13, @isCarmpvLT = 1

else

SET @startPos = 14

end

WHILE @j < LEN(@vin)

BEGIN

SET @j = @j + 1

SET @chr = SUBSTRING(@vin, @j, 1)

IF

@j <> 9 AND @j < @startPos AND @chr NOT LIKE '[0-9ABCDEFGHJKLMNPRSTUVWXYZ\*]'

OR

@j <> 9 AND @j >= @startPos AND @chr NOT LIKE '[0-9\*]'

OR

@j = 9 AND @chr NOT LIKE '[0-9X\*]'

OR

@j = 10 AND @chr NOT LIKE '[1-9ABCDEFGHJKLMNPRSTVWXY]'

BEGIN

IF @chr = ' '

SET @chr = 'space'

IF @CorrectedVIN = ''

SET @CorrectedVIN = @vin

SET @invalidChars = @invalidChars + ', ' + CAST(@j AS VARCHAR) + ':' + @chr

SET @CorrectedVIN = LEFT(@CorrectedVIN, @j-1) + '!' + SUBSTRING(@CorrectedVIN, @j+1, 100)

END

END

IF @invalidChars <> ''

set @ReturnCode = @ReturnCode + ' 400 '

if not @year is null

begin

declare @mdlyr int = abs(dbo.fVinModelYear2 (upper(@vin)))

declare @diff int = abs(@year - @mdlyr)

if (@diff <> 0 and @diff <> 30)

select @ReturnCode = @ReturnCode + ' 12 '

end

INSERT INTO #DecodingItem ([DecodingId], [Source], [CreatedOn], [Priority], [PatternId], [Keys], [VinSchemaId], [WmiId], [ElementId], [AttributeId], [Value])

SELECT

@did,

'Default',

isnull(dv.UpdatedOn, dv.CreatedOn),

10,

null,

null,

null,

null,

dv.ElementId,

dv.DefaultValue,

case when e.datatype='lookup' and dv.DefaultValue = '0' then 'Not Applicable' else dbo.fElementAttributeValue (dv.ElementId, dv.DefaultValue) end

FROM

DefaultValue dv

inner join element e on dv.ElementId = e.id

where dv.VehicleTypeId = @vehicleType and dv.DefaultValue is not null and dv.elementid not in (select distinct elementid from #decodingitem)

if LEN(@vin) < 17

select @ReturnCode = @ReturnCode + ' 6 '

else

begin

declare @CD char(1) = SUBSTRING(@vin, 9, 1)

declare @calcCD char(1) = ''

set @calcCD = dbo.[fVINCheckDigit2](@vin, @isCarmpvLT)

IF @cd <> @calcCD

begin

set @ReturnCode = @ReturnCode + ' 1 '

end

END

declare @errors varchar(100) = @ReturnCode

set @errors = replace(@errors, ' 9 ', '')

set @errors = replace(@errors, ' 10 ', '')

set @errors = replace(@errors, ' 12 ', '')

set @errors = ltrim(rtrim(@errors))

if @errors = '' or @errors = '14'

set @ReturnCode = ' 0 ' + @ReturnCode

if @ReturnCode like '% 4 %'

select @AdditionalDecodingInfo = isnull(additionalerrortext,'') from ErrorCode where id = 4

if @ReturnCode like '% 5 %'

select @AdditionalDecodingInfo = isnull(additionalerrortext,'') from ErrorCode where id = 5

if @ReturnCode like '% 14 %'

select @AdditionalDecodingInfo = rtrim(ltrim(isnull(@AdditionalDecodingInfo, '') + ' Unused position(s): ' + @UnUsedPositions + '; '))

if @ReturnCode like '% 400 %'

select @AdditionalDecodingInfo = rtrim(ltrim(isnull(@AdditionalDecodingInfo, '') + ' Invalid character(s): ' + SUBSTRING(@invalidChars, 3, LEN(@invalidChars)-2) + '; '))

if @conclusive = 0

set @AdditionalDecodingInfo = @AdditionalDecodingInfo + case when @AdditionalDecodingInfo = '' then '' else char(13) end + 'The Model Year decoded for this VIN may be incorrect. If you know the Model year, please enter it and decode again to get more accurate information.'

declare @offRoadNote varchar(100) = ' NOTE: Disregard if this is an off-road vehicle PIN, as check digit calculation may not be accurate.'

declare @errorMessages varchar(max) = null

declare @errorCodes varchar(500) = null

declare @oneError varchar(10) = ''

select

@errorMessages = isnull(ltrim(rtrim(@errorMessages)) + '; ' + name, name),

@errorCodes = isnull(ltrim(rtrim(@errorCodes)) + ',' + cast(id as varchar), cast(id as varchar)),

@oneError = Id

from

(select id, Name + case when @isOffRoad = 1 and id = 1 then @offRoadNote else '' end as Name from ErrorCode ) as t

where @ReturnCode like '% ' + cast(id as varchar) + ' %'

order by id

select @errorMessages = left(@errorMessages, 500)

INSERT INTO #DecodingItem ([DecodingId], [Source], [Priority], [PatternId], [Keys], [VinSchemaId], [WmiId], [ElementId], [AttributeId], [Value])

SELECT

@did, 'Corrections', 999,

null, '', null, null, p.ElementId,

p.AttributeId, p.Value as Value

FROM

(

select 142 as ElementId, @CorrectedVIN as AttributeId, @CorrectedVIN as Value

union

select 143, @errorCodes, @errorCodes

union

select 191, @errorMessages, @errorMessages

union

select 144, @ErrorBytes, @ErrorBytes

union

select 156, @AdditionalDecodingInfo, @AdditionalDecodingInfo

) p

declare @tryagain bit = 0, @maxYear int = year(getdate())+1

if

@doNotRetry = 0

and @ReturnCode like '% 8 %'

and @pass = 1 and @modelYear between 1980 and @maxYear

begin

if @modelYear >= 2010

begin

select @modelYear = @modelYear - 30, @tryagain = 1

end

else if @modelYear + 30 <= @maxYear

begin

select @modelYear = @modelYear + 30, @tryagain = 1

end

if @tryagain = 1

begin

truncate table #DecodingItem

delete from @tbl1

select @ReturnCode = ''

GOTO start\_again;

end

end

update #DecodingItem

set TobeQCed = vs.TobeQCed

from #DecodingItem d inner join VinSchema vs on d.VinSchemaId = vs.Id and vs.TobeQCed = 1

where lower(left(isnull(d.Source, ''), 7)) in ('pattern', 'formula', 'enginem', 'convers')

if isnull(@includeNotPublicilyAvailable, 0) = 0

delete

from #DecodingItem

where TobeQCed = 1

/\*The original code below (lines 590 to 630) has been commented out to remove its output from the query

if @NoOutput = 0

begin

select

e.GroupName, e.Name as Variable, REPLACE(REPLACE(REPLACE(t.Value, CHAR(9), ' '), CHAR(13), ' '), CHAR(10), ' ') as Value,

t.PatternId, t.VinSchemaId, t.Keys, e.id as ElementId, t.AttributeId, t.CreatedOn as CreatedOn, t.WmiId,

e.Code, e.DataType , e.Decode, t.Source, t.ToBeQCed as ToBeQCd

from

Element e

left outer join #DecodingItem t on t.ElementId = e.Id

where

(isnull(e.Decode, '') <> '')

and ((@includeAll) = 1 or (isnull(@includeAll, 0) = 0 and not t.ElementId is null))

and (@includePrivate = 1 or isnull(e.IsPrivate, 0) = 0 )

order by

case isnull(e.GroupName, '')

when '' then 0

when 'General' then 1

when 'Exterior / Body' then 2

when 'Exterior / Dimension' then 3

when 'Exterior / Truck' then 4

when 'Exterior / Trailer' then 5

when 'Exterior / Wheel tire' then 6

when 'Interior' then 7

when 'Interior / Seat' then 8

when 'Mechanical / Transmission' then 9

when 'Mechanical / Drivetrain' then 10

when 'Mechanical / Brake' then 11

when 'Mechanical / Battery' then 12

when 'Mechanical / Battery / Charger' then 13

when 'Engine' then 14

when 'Passive Safety System' then 15

when 'Passive Safety System / Air Bag Location' then 16

when 'Active Safety System' then 17

when 'Internal' then 18

else 99 end

,e.Id

end \*/

IF OBJECT\_ID('tempdb..#DecodingItem') IS NOT NULL

--The following creates a temporary table to select the make and model from the decode table for the query output.

--This will output a maximum of 2 rows, and there will be output even if a row or column is null.

IF @v1 is null or @v1 = '' --handle empty VINs

SET @v1 = 'No VIN' --Output for empty VIN field is "No VIN" to aid processing

-- @v1 is a variable that was added on line 33 to store the original input of the VIN and retain it for output

begin

create table #tempTable (makeModel varchar(250), vin varchar(50), vehicleYear int, errorMessage varchar(250), elementColumn int);

begin

INSERT INTO #tempTable (makeModel, vin, vehicleYear, elementColumn)

SELECT TOP 1 Value,@v1, @modelYear, ElementId

--select only 1 row total, make and model from Value and the VIN @v1.

--this controls for VINs that will generate multiple model listings

FROM #DecodingItem

WHERE ElementId = '26'

UNION SELECT TOP 1 Value, @v1, @modelYear, ElementId --select 1 row

FROM #DecodingItem

WHERE ElementId = '28'

--ElementId 26 is Make, 28 is Model

UNION SELECT TOP 1 Value, @v1, @modelYear, ElementId

FROM #DecodingItem

WHERE ElementId = '24' --Fuel Type - Primary

UNION SELECT TOP 1 Value, @v1, @modelYear, ElementId

FROM #DecodingItem

WHERE ElementId = '126' --Electrification Level

UNION SELECT TOP 1 Value, @v1, @modelYear, ElementId

FROM #DecodingItem

WHERE ElementId = '5' --Body Class

ORDER BY ElementId ASC

--order the output in ascending value

declare @cMake varchar(50)

SELECT TOP 1 @cMake = makeModel FROM #tempTable WHERE elementColumn = '26'

--set column equal to a variable

IF @cMake is null

--if column is empty, generate output (this tests the row where the Make is listed)

INSERT INTO #tempTable (vin)

VALUES (@v1)

declare @cModel varchar(50)

SELECT TOP 1 @cModel = makeModel FROM #tempTable WHERE elementColumn = '28'

--set column equal to variable

declare @cFuelType varchar(50)

SELECT @cFuelType = makeModel FROM #tempTable WHERE elementColumn = '24'

declare @cElectrification varchar(50)

SELECT @cElectrification = makeModel FROM #tempTable WHERE elementColumn = '126'

declare @cbodyStyle varchar(50)

SELECT @cbodyStyle = makeModel FROM #tempTable WHERE elementColumn = '5'

end

SELECT @errorMessages = 'error' WHERE @errorMessages LIKE '%4 -%'

SELECT @errorMessages = 'error' WHERE @errorMessages LIKE '%11 -%'

SELECT TOP 1 @v1, vehicleYear, @cMake, @cModel, @cbodyStyle, @cElectrification, @cFuelType, @errorMessages

FROM #tempTable

--final output table gives the Make, Model, VIN, Vehicle year, Electrification level, Fuel Type (Primary) and the error messages

--associated with the VIN, in 1 row

drop table #DecodingItem

drop table #tempTable

end

end