Generate SQL Create Scripts for existing tables with Query

Asked 10 years, 9 months ago Active 3 months ago Viewed 138k times

63

I'm trying to get the CREATE scripts for existing tables within SQL Server 2008. I assume I can do this by querying the sys.tables somehow, however this isn't returning me the CREATE script data.

sql sql-server tsql

31

edited Mar 9 '19 at 19:55

asked Apr 1 '09 at 17:51

cweston

9,511 15 71 102

Just `CREATE TABLE` or all of the other baggage, e.g indexes, relations, check constraints, triggers *(With the firing order preserved!)*, ...? As soon as you start adding relations the order of creation becomes critical. – HABO Jun 1 '13 at 16:44

13 Answers

134

+50

Possible this be helpful for you. This script generate indexes, FK's, PK and common structure for any table.

For example -

DDL:

```
CREATE TABLE [dbo].[WorkOut](
    [WorkOutID] [bigint] IDENTITY(1,1) NOT NULL,
    [TimeSheetDate] [datetime] NOT NULL,
    [DateOut] [datetime] NOT NULL,
    [EmployeeID] [int] NOT NULL,
    [IsMainWorkPlace] [bit] NOT NULL,
    [DepartmentUID] [uniqueidentifier] NOT NULL,
    [WorkPlaceUID] [uniqueidentifier] NULL,
    [TeamUID] [uniqueidentifier] NULL,
    [WorkShiftCD] [nvarchar](10) NULL,
    [WorkHours] [real] NULL,
    [AbsenceCode] [varchar](25) NULL,
    [PaymentType] [char](2) NULL,
    [CategoryID] [int] NULL,
    [Year]  AS (datepart(year,[TimeSheetDate])),
 CONSTRAINT [PK_WorkOut] PRIMARY KEY CLUSTERED
(
    [WorkOutID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
```

```
ALTER TABLE [dbo].[WorkOut]  WITH CHECK ADD  CONSTRAINT [FK_WorkOut_Employee_EmployeeID]
FOREIGN KEY([EmployeeID])
REFERENCES [dbo].[Employee] ([EmployeeID])

ALTER TABLE [dbo].[WorkOut] CHECK CONSTRAINT [FK_WorkOut_Employee_EmployeeID]
```

Query:

```
DECLARE @table_name SYSNAME
SELECT @table_name = 'dbo.WorkOut'

DECLARE
      @object_name SYSNAME
    , @object_id INT

SELECT
      @object_name = '[' + s.name + '].[' + o.name + ']'
    , @object_id = o.[object_id]
FROM sys.objects o WITH (NOWAIT)
JOIN sys.schemas s WITH (NOWAIT) ON o.[schema_id] = s.[schema_id]
WHERE s.name + '.' + o.name = @table_name
    AND o.[type] = 'U'
    AND o.is_ms_shipped = 0

DECLARE @SQL NVARCHAR(MAX) = ''

;WITH index_column AS
(
    SELECT
          ic.[object_id]
        , ic.index_id
        , ic.is_descending_key
        , ic.is_included_column
        , c.name
    FROM sys.index_columns ic WITH (NOWAIT)
    JOIN sys.columns c WITH (NOWAIT) ON ic.[object_id] = c.[object_id] AND ic.column_id
= c.column_id
    WHERE ic.[object_id] = @object_id
),
fk_columns AS
(
     SELECT
          k.constraint_object_id
        , cname = c.name
        , rcname = rc.name
    FROM sys.foreign_key_columns k WITH (NOWAIT)
    JOIN sys.columns rc WITH (NOWAIT) ON rc.[object_id] = k.referenced_object_id AND
rc.column_id = k.referenced_column_id
    JOIN sys.columns c WITH (NOWAIT) ON c.[object_id] = k.parent_object_id AND
c.column_id = k.parent_column_id
    WHERE k.parent_object_id = @object_id
)
SELECT @SQL = 'CREATE TABLE ' + @object_name + CHAR(13) + '(' + CHAR(13) + STUFF((
    SELECT CHAR(9) + ', [' + c.name + '] ' +
        CASE WHEN c.is_computed = 1
            THEN 'AS ' + cc.[definition]
            ELSE UPPER(tp.name) +
                CASE WHEN tp.name IN ('varchar', 'char', 'varbinary', 'binary', 'text')
                       THEN '(' + CASE WHEN c.max_length = -1 THEN 'MAX' ELSE
CAST(c.max_length AS VARCHAR(5)) END + ')'
                     WHEN tp.name IN ('nvarchar', 'nchar', 'ntext')
                       THEN '(' + CASE WHEN c.max_length = -1 THEN 'MAX' ELSE
CAST(c.max_length / 2 AS VARCHAR(5)) END + ')'
                     WHEN tp.name IN ('datetime2', 'time2', 'datetimeoffset')
```

```
                    ELSE ''
                END +
                CASE WHEN c.collation_name IS NOT NULL THEN ' COLLATE ' +
c.collation_name ELSE '' END +
                CASE WHEN c.is_nullable = 1 THEN ' NULL' ELSE ' NOT NULL' END +
                CASE WHEN dc.[definition] IS NOT NULL THEN ' DEFAULT' + dc.[definition]
ELSE '' END +
                CASE WHEN ic.is_identity = 1 THEN ' IDENTITY(' +
CAST(ISNULL(ic.seed_value, '0') AS CHAR(1)) + ',' + CAST(ISNULL(ic.increment_value, '1')
AS CHAR(1)) + ')' ELSE '' END
        END + CHAR(13)
    FROM sys.columns c WITH (NOWAIT)
    JOIN sys.types tp WITH (NOWAIT) ON c.user_type_id = tp.user_type_id
    LEFT JOIN sys.computed_columns cc WITH (NOWAIT) ON c.[object_id] = cc.[object_id]
AND c.column_id = cc.column_id
    LEFT JOIN sys.default_constraints dc WITH (NOWAIT) ON c.default_object_id != 0 AND
c.[object_id] = dc.parent_object_id AND c.column_id = dc.parent_column_id
    LEFT JOIN sys.identity_columns ic WITH (NOWAIT) ON c.is_identity = 1 AND c.
[object_id] = ic.[object_id] AND c.column_id = ic.column_id
    WHERE c.[object_id] = @object_id
    ORDER BY c.column_id
    FOR XML PATH(''), TYPE).value('.', 'NVARCHAR(MAX)'), 1, 2, CHAR(9) + ' ')
    + ISNULL((SELECT CHAR(9) + ', CONSTRAINT [' + k.name + '] PRIMARY KEY (' +
                    (SELECT STUFF((
                         SELECT ', [' + c.name + '] ' + CASE WHEN ic.is_descending_key =
1 THEN 'DESC' ELSE 'ASC' END
                         FROM sys.index_columns ic WITH (NOWAIT)
                         JOIN sys.columns c WITH (NOWAIT) ON c.[object_id] = ic.
[object_id] AND c.column_id = ic.column_id
                         WHERE ic.is_included_column = 0
                             AND ic.[object_id] = k.parent_object_id
                             AND ic.index_id = k.unique_index_id
                         FOR XML PATH(N''), TYPE).value('.', 'NVARCHAR(MAX)'), 1, 2,
''))
            + ')' + CHAR(13)
            FROM sys.key_constraints k WITH (NOWAIT)
            WHERE k.parent_object_id = @object_id
                AND k.[type] = 'PK'), '') + ')'  + CHAR(13)
    + ISNULL((SELECT (
        SELECT CHAR(13) +
             'ALTER TABLE ' + @object_name + ' WITH'
            + CASE WHEN fk.is_not_trusted = 1
                THEN ' NOCHECK'
                ELSE ' CHECK'
              END +
              ' ADD CONSTRAINT [' + fk.name  + '] FOREIGN KEY('
              + STUFF((
                SELECT ', [' + k.cname + ']'
                FROM fk_columns k
                WHERE k.constraint_object_id = fk.[object_id]
                FOR XML PATH(''), TYPE).value('.', 'NVARCHAR(MAX)'), 1, 2, '')
               + ')' +
              ' REFERENCES [' + SCHEMA_NAME(ro.[schema_id]) + '].[' + ro.name + '] ('
              + STUFF((
                SELECT ', [' + k.rcname + ']'
                FROM fk_columns k
                WHERE k.constraint_object_id = fk.[object_id]
                FOR XML PATH(''), TYPE).value('.', 'NVARCHAR(MAX)'), 1, 2, '')
               + ')'
            + CASE
                WHEN fk.delete_referential_action = 1 THEN ' ON DELETE CASCADE'
                WHEN fk.delete_referential_action = 2 THEN ' ON DELETE SET NULL'
                WHEN fk.delete_referential_action = 3 THEN ' ON DELETE SET DEFAULT'
                ELSE ''
              END
            + CASE
```

```
                ELSE ''
              END
            + CHAR(13) + 'ALTER TABLE ' + @object_name + ' CHECK CONSTRAINT [' + fk.name
+ ']' + CHAR(13)
        FROM sys.foreign_keys fk WITH (NOWAIT)
        JOIN sys.objects ro WITH (NOWAIT) ON ro.[object_id] = fk.referenced_object_id
        WHERE fk.parent_object_id = @object_id
        FOR XML PATH(N''), TYPE).value('.', 'NVARCHAR(MAX)')), '')
    + ISNULL(((SELECT
         CHAR(13) + 'CREATE' + CASE WHEN i.is_unique = 1 THEN ' UNIQUE' ELSE '' END
                + ' NONCLUSTERED INDEX [' + i.name + '] ON ' + @object_name + ' (' +
                STUFF((
                SELECT ', [' + c.name + ']' + CASE WHEN c.is_descending_key = 1 THEN '
DESC' ELSE ' ASC' END
                FROM index_column c
                WHERE c.is_included_column = 0
                    AND c.index_id = i.index_id
                FOR XML PATH(''), TYPE).value('.', 'NVARCHAR(MAX)'), 1, 2, '') + ')'
                + ISNULL(CHAR(13) + 'INCLUDE (' +
                    STUFF((
                    SELECT ', [' + c.name + ']'
                    FROM index_column c
                    WHERE c.is_included_column = 1
                        AND c.index_id = i.index_id
                    FOR XML PATH(''), TYPE).value('.', 'NVARCHAR(MAX)'), 1, 2, '') +
')', '')  + CHAR(13)
        FROM sys.indexes i WITH (NOWAIT)
        WHERE i.[object_id] = @object_id
            AND i.is_primary_key = 0
            AND i.[type] = 2
        FOR XML PATH(''), TYPE).value('.', 'NVARCHAR(MAX)')
    ), '')

PRINT @SQL
--EXEC sys.sp_executesql @SQL
```

Output:

```
CREATE TABLE [dbo].[WorkOut]
(
      [WorkOutID] BIGINT NOT NULL IDENTITY(1,1)
    , [TimeSheetDate] DATETIME NOT NULL
    , [DateOut] DATETIME NOT NULL
    , [EmployeeID] INT NOT NULL
    , [IsMainWorkPlace] BIT NOT NULL DEFAULT((1))
    , [DepartmentUID] UNIQUEIDENTIFIER NOT NULL
    , [WorkPlaceUID] UNIQUEIDENTIFIER NULL
    , [TeamUID] UNIQUEIDENTIFIER NULL
    , [WorkShiftCD] NVARCHAR(10) COLLATE Cyrillic_General_CI_AS NULL
    , [WorkHours] REAL NULL
    , [AbsenceCode] VARCHAR(25) COLLATE Cyrillic_General_CI_AS NULL
    , [PaymentType] CHAR(2) COLLATE Cyrillic_General_CI_AS NULL
    , [CategoryID] INT NULL
    , [Year] AS (datepart(year,[TimeSheetDate]))
    , CONSTRAINT [PK_WorkOut] PRIMARY KEY ([WorkOutID] ASC)
)

ALTER TABLE [dbo].[WorkOut] WITH CHECK ADD CONSTRAINT [FK_WorkOut_Employee_EmployeeID]
FOREIGN KEY([EmployeeID]) REFERENCES [dbo].[Employee] ([EmployeeID])
ALTER TABLE [dbo].[WorkOut] CHECK CONSTRAINT [FK_WorkOut_Employee_EmployeeID]

CREATE NONCLUSTERED INDEX [IX_WorkOut_WorkShiftCD_AbsenceCode] ON [dbo].[WorkOut]
([WorkShiftCD] ASC, [AbsenceCode] ASC)
INCLUDE ([WorkOutID], [WorkHours])
```

[How to Generate a CREATE TABLE Script For an Existing Table: Part 1]

edited Dec 30 '13 at 13:23

answered May 28 '13 at 16:53

Devart

104k 20 149 164

4 No problem :) I add support for computed columns tomorrow. – Devart May 29 '13 at 12:55

1 Also, I noticed that the columns are reordered alphabetically, not keeping their original order. – enb081 May 29 '13 at 13:23

4 Please see the updated answer (added support for computed columns, fix order for columns and speed up query). – Devart May 30 '13 at 5:55

1 @Devart I have updated my query, please see updated question for ddl : stackoverflow.com/questions/38998330/… – Neelam Sharma Aug 19 '16 at 10:08

1 You should use `quotename()` instead of `'[' + thing + ']'` – DJL Dec 13 '18 at 13:23

16

do you mean you wish to create a TSQL script which generates a CREATE script, or use the Management tools in SQL SERVER Management Studio to generate a Create script?

If it's the latter, it's a simply matter of right-clicking a table, and selecting Script Table As -> Create To -> New Query Window.

If you want the whole database scripted, then right click the database and select Tasks--> Generate Scripts... and then follow the wizard

otherwise it's a matter of selecting all sorts of fun things out of the various system tables.

answered Apr 1 '09 at 18:04

Stephen Wrighton

29.9k 6 60 84

I was actually looking for a way to pro grammatically get these, as this will be ran from a .NET app to generate a master script file. – cweston Apr 7 '09 at 23:35

Thank you! I knew there was a way to have all the create to scripts in one place, your answer pointed me to find out where. – Tschallacka Aug 23 '17 at 10:02

5

I realize this question is old, but it recently popped up in a search I just ran, so I thought I'd post an alternative to the above answer.

If you are looking to generate `create` scripts programmatically in .Net, I would highly recommend looking into Server Management Objects (SMO) or Distributed Management Objects (DMO) -- depending on which version of SQL Server you are using (the former is 2005+, the latter 2000). Using these libraries, scripting a table is as easy as:

```
Server server      = new Server(".");
Database northwind = server.Databases["Northwind"];
Table categories   = northwind.Tables["Categories"];
```

```
script.CopyTo(scriptArray, 0);
```

Here is a blog post with more information.

edited Mar 10 '13 at 3:48

Drew
1,397 3 20 43

answered May 9 '12 at 13:27

Bobby D
2,050 13 21

3

Try sp_helptext Equivalent for Tables?

answered Apr 1 '09 at 18:48

John MacIntyre
12.5k 10 61 100

3

"Easiest way is to use the built-in feature of SQL Management Studio" but... I have resolved it with a function and a couple of procedures. For example, to obtain the create table for a table named 'table_name', you have to execute just the procedure called sp_ppinScriptTabla:

```
Exec sp_ppinScriptTabla 'table_name'
```

Here is the tsql script code:

```
Use Master
GO

Create Function sp_ppinTipoLongitud
(
    @xtype int,
    @length int,
    @isnullable int
)
Returns Varchar(512)
As
Begin
    -- Función que a partir de un tipo de datos y una logitud, devuelve el texto del
tipo.
    -- Por ejemplo: para xtype=varchar y length=10 devolverá "varchar(10)"
    Declare @ret varchar(512)
    Set @ret = ''

    Select @ret = t.name +
    Case When name in ('varchar', 'nvarchar', 'char', 'nchar') Then '(' +
Convert(varchar, @length) + ')' Else '' End + ' ' +
    Case @isnullable When 1 Then 'NULL' Else 'NOT NULL' End
    From systypes t
    Where t.xtype = @xtype

    Return @ret
End
GO

Create Procedure sp_ppinScriptLlavesForaneas
(
    @vchTabla sysname
```

```
Begin

    DECLARE @tmpFK table(
        TablaF sysname,
        TablaR sysname,
        ColF sysname,
        ColR sysname,
        FKName sysname)

    -- obtengo las llaves foraneas en @vchForeign
    Declare @vchForeign varchar(8000), @FKName sysname, @vchColumnasF varchar(4000),
@vchColumnasR varchar(4000), @ColF sysname, @ColR sysname
    Declare @vchTemp varchar(1000), @TablaR sysname

    Insert into @tmpFK
    Select TablaF.name AS TablaF, TablaR.name AS TablaR, ColF.name AS ColF, ColR.name AS
ColR, ofk.name AS FKName
    From sysforeignkeys fk, sysobjects ofk, sysobjects TablaF, sysobjects TablaR,
    syscolumns ColF, syscolumns ColR
    Where TablaF.name = @vchTabla
    And ofk.id = fk.constid
    And TablaF.id = fk.fkeyid
    And TablaR.id = fk.rkeyid
    And ColF.id = TablaF.id And ColF.colid = fk.fkey
    And ColR.id = TablaR.id And ColR.colid = fk.rkey
    order by FKName

    Set @vchForeign = ''
    While Exists ( Select * From @tmpFK )
    Begin
        Select Top 1 @FKName = FKName From @tmpFK
        Set @vchColumnasF = ''
        Set @vchColumnasR = ''
        While Exists ( Select * From @tmpFK Where FKName = @FKName )
        Begin
            Select Top 1 @ColF = ColF, @ColR = ColR, @TablaR = TablaR From @tmpFK Where
FKName = @FKName
            Delete From @tmpFK Where ColF = @ColF And ColR = @ColR And TablaR = @TablaR
And FKName = @FKName
            Set @vchColumnasF = @vchColumnasF + @ColF + ', '
            Set @vchColumnasR = @vchColumnasR + @ColR + ', '
        End

        Set @vchColumnasF = LEFT(@vchColumnasF, LEN(@vchColumnasF) - 1)
        Set @vchColumnasR = LEFT(@vchColumnasR, LEN(@vchColumnasR) - 1)
        Set @vchTemp = 'Constraint ' + @FKName + ' Foreign Key (' + @vchColumnasF + ') '
        Set @vchTemp = @vchTemp + 'References ' + @TablaR + ' (' + @vchColumnasR + ')'
        Set @vchForeign = @vchForeign + char(9) + @vchTemp + ',' + char(13)
    End

    Select @vchResultado = Case When Len(@vchForeign) >=2 Then Left(@vchForeign,
Len(@vchForeign) - 2) Else @vchForeign End
End
GO

Create Procedure sp_ppinScriptTabla
(
    @vchTabla sysname
)
AS

Set nocount on

-- Obtengo las foreign keys
Declare @foreign varchar(8000)
Exec sp_ppinScriptLlavesForaneas @vchTabla, @foreign output
```

```
Case o.xtype When 'U' Then 'Table' When 'P' Then 'Procedure' Else '??' End + ' ' +
@vchTabla + char(13) + '('
From sysobjects o
Where o.name = @vchTabla
Union all
-- Campos + identitys + DEFAULTS
select char(9) + c.name + ' ' +                                    -- Nombre
dbo.sp_ppinTipoLongitud(t.xtype, c.length, c.isnullable) +          -- Tipo(longitud)
Case When c.colstat & 1 = 1                                        -- Identity (si aplica)
    Then ' Identity(' + convert(varchar, ident_seed(@vchTabla)) + ',' + Convert(varchar,
ident_incr(@vchTabla)) + ')'
    Else ''
End +
Case When not od.name is null                                      -- Defaults (si aplica)
    Then ' Constraint ' + od.name + ' Default ' + replace(replace(cd.text, '((', '('),
'))', ')')
    Else ''
End + ', '
from sysobjects o, syscolumns c
LEFT OUTER JOIN sysobjects od On od.id = c.cdefault LEFT OUTER join syscomments cd On
cd.id = od.id,
systypes t
where o.id = object_id(@vchTabla)
and o.id = c.id
and c.xtype = t.xtype
Union all
-- Primary Keys y Unique keys
select char(9) + 'Constraint ' + o.name + ' ' +
Case o.xtype When 'PK' Then 'Primary Key' Else 'Unique' End + ' ' +
dbo.sp_ppinCamposIndice (db_name(), @vchTabla, i.indid) + ', '
from sysobjects o, sysindexes i
where o.parent_obj = object_id(@vchTabla)
and o.xtype in ('PK','UQ')
and i.id = o.parent_obj
and o.name = i.name
Union all
-- Check constraints
select char(9) + 'Constraint ' + o.name + ' Check ' + c.text + ', '
from sysobjects o, syscomments c
where o.parent_obj = object_id(@vchTabla)
and o.xtype in ('C')
and o.id = c.id
Union all
-- Foreign keys
Select @foreign
Union all
Select ')'

Set nocount off
GO
```

answered Apr 8 '09 at 19:56

community wiki
Federico Colombo

2

You forgot to include the stored procedure or function script for `sp_ppinCamposIndice`

edited Nov 23 '11 at 8:52

Hans Olsson
49.4k 13 83 107

answered Jan 19 '10 at 22:06

Aaron Thomason
29 1

1

Try this (using "Results to text"):

```
SELECT
ISNULL(smsp.definition, ssmsp.definition) AS [Definition]
FROM
sys.all_objects AS sp
LEFT OUTER JOIN sys.sql_modules AS smsp ON smsp.object_id = sp.object_id
LEFT OUTER JOIN sys.system_sql_modules AS ssmsp ON ssmsp.object_id = sp.object_id
WHERE
(sp.type = N'V' OR sp.type = N'P' OR sp.type = N'RF' OR
sp.type=N'PC')and(sp.name=N'YourObjectName' and SCHEMA_NAME(sp.schema_id)=N'dbo')
```

- C: Check constraint
- D: Default constraint
- F: Foreign Key constraint
- L: Log
- P: Stored procedure
- PK: Primary Key constraint
- RF: Replication Filter stored procedure
- S: System table
- TR: Trigger
- U: User table
- UQ: Unique constraint
- V: View
- X: Extended stored procedure

Cheers,

answered Jun 5 '15 at 17:33

Sp4
49 3

Doesn't work for tables (and it was the main point of question). – Deadsheep39 Nov 22 '18 at 9:22

1

Here's a slight variation on @Devart 's answer so you can get the CREATE script for a temp table.

Please note that since the @SQL variable is an `NVARCHAR(MAX)` data type you might not be able to copy it from the result using just only SSMS. Please see this question to see how to get the full value of a **MAX** field.

```
DECLARE @temptable_objectid INT = OBJECT_ID('tempdb.db.#Temp');

DECLARE
      @object_name SYSNAME
```

```
      @object_name = '[' + s.name + '].[' + o.name + ']'
    , @object_id = o.[object_id]
FROM tempdb.sys.objects o WITH (NOWAIT)
JOIN tempdb.sys.schemas s WITH (NOWAIT) ON o.[schema_id] = s.[schema_id]
WHERE object_id = @temptable_objectid

DECLARE @SQL NVARCHAR(MAX) = ''

;WITH index_column AS
(
    SELECT
          ic.[object_id]
        , ic.index_id
        , ic.is_descending_key
        , ic.is_included_column
        , c.name
    FROM tempdb.sys.index_columns ic WITH (NOWAIT)
    JOIN tempdb.sys.columns c WITH (NOWAIT) ON ic.[object_id] = c.[object_id] AND
ic.column_id = c.column_id
    WHERE ic.[object_id] = @object_id
),
fk_columns AS
(
     SELECT
          k.constraint_object_id
        , cname = c.name
        , rcname = rc.name
    FROM tempdb.sys.foreign_key_columns k WITH (NOWAIT)
    JOIN tempdb.sys.columns rc WITH (NOWAIT) ON rc.[object_id] = k.referenced_object_id
AND rc.column_id = k.referenced_column_id
    JOIN tempdb.sys.columns c WITH (NOWAIT) ON c.[object_id] = k.parent_object_id AND
c.column_id = k.parent_column_id
    WHERE k.parent_object_id = @object_id
)
SELECT @SQL = 'CREATE TABLE ' + @object_name + CHAR(13) + '(' + CHAR(13) + STUFF((
    SELECT CHAR(9) + ', [' + c.name + '] ' +
        CASE WHEN c.is_computed = 1
            THEN 'AS ' + cc.[definition]
            ELSE UPPER(tp.name) +
                CASE WHEN tp.name IN ('varchar', 'char', 'varbinary', 'binary', 'text')
                       THEN '(' + CASE WHEN c.max_length = -1 THEN 'MAX' ELSE
CAST(c.max_length AS VARCHAR(5)) END + ')'
                     WHEN tp.name IN ('nvarchar', 'nchar', 'ntext')
                       THEN '(' + CASE WHEN c.max_length = -1 THEN 'MAX' ELSE
CAST(c.max_length / 2 AS VARCHAR(5)) END + ')'
                     WHEN tp.name IN ('datetime2', 'time2', 'datetimeoffset')
                       THEN '(' + CAST(c.scale AS VARCHAR(5)) + ')'
                     WHEN tp.name = 'decimal'
                       THEN '(' + CAST(c.[precision] AS VARCHAR(5)) + ',' + CAST(c.scale
AS VARCHAR(5)) + ')'
                     ELSE ''
                END +
                CASE WHEN c.collation_name IS NOT NULL THEN ' COLLATE ' +
c.collation_name ELSE '' END +
                CASE WHEN c.is_nullable = 1 THEN ' NULL' ELSE ' NOT NULL' END +
                CASE WHEN dc.[definition] IS NOT NULL THEN ' DEFAULT' + dc.[definition]
ELSE '' END +
                CASE WHEN ic.is_identity = 1 THEN ' IDENTITY(' +
CAST(ISNULL(ic.seed_value, '0') AS CHAR(1)) + ',' + CAST(ISNULL(ic.increment_value, '1')
AS CHAR(1)) + ')' ELSE '' END
        END + CHAR(13)
    FROM tempdb.sys.columns c WITH (NOWAIT)
    JOIN tempdb.sys.types tp WITH (NOWAIT) ON c.user_type_id = tp.user_type_id
    LEFT JOIN tempdb.sys.computed_columns cc WITH (NOWAIT) ON c.[object_id] = cc.
[object_id] AND c.column_id = cc.column_id
    LEFT JOIN tempdb.sys.default_constraints dc WITH (NOWAIT) ON c.default_object_id !=
```

```
    WHERE c.[object_id] = @object_id
    ORDER BY c.column_id
    FOR XML PATH(''), TYPE).value('.', 'NVARCHAR(MAX)'), 1, 2, CHAR(9) + ' ')
    + ISNULL((SELECT CHAR(9) + ', CONSTRAINT [' + k.name + '] PRIMARY KEY (' +
                    (SELECT STUFF((
                         SELECT ', [' + c.name + '] ' + CASE WHEN ic.is_descending_key =
1 THEN 'DESC' ELSE 'ASC' END
                         FROM tempdb.sys.index_columns ic WITH (NOWAIT)
                         JOIN tempdb.sys.columns c WITH (NOWAIT) ON c.[object_id] = ic.
[object_id] AND c.column_id = ic.column_id
                         WHERE ic.is_included_column = 0
                             AND ic.[object_id] = k.parent_object_id
                             AND ic.index_id = k.unique_index_id
                         FOR XML PATH(N''), TYPE).value('.', 'NVARCHAR(MAX)'), 1, 2,
''))
            + ')' + CHAR(13)
            FROM tempdb.sys.key_constraints k WITH (NOWAIT)
            WHERE k.parent_object_id = @object_id
                AND k.[type] = 'PK'), '') + ')'  + CHAR(13)
    + ISNULL((SELECT (
        SELECT CHAR(13) +
             'ALTER TABLE ' + @object_name + ' WITH'
            + CASE WHEN fk.is_not_trusted = 1
                THEN ' NOCHECK'
                ELSE ' CHECK'
              END +
              ' ADD CONSTRAINT [' + fk.name  + '] FOREIGN KEY('
              + STUFF((
                SELECT ', [' + k.cname + ']'
                FROM fk_columns k
                WHERE k.constraint_object_id = fk.[object_id]
                FOR XML PATH(''), TYPE).value('.', 'NVARCHAR(MAX)'), 1, 2, '')
               + ')' +
              ' REFERENCES [' + SCHEMA_NAME(ro.[schema_id]) + '].[' + ro.name + '] ('
              + STUFF((
                SELECT ', [' + k.rcname + ']'
                FROM fk_columns k
                WHERE k.constraint_object_id = fk.[object_id]
                FOR XML PATH(''), TYPE).value('.', 'NVARCHAR(MAX)'), 1, 2, '')
               + ')'
            + CASE
                WHEN fk.delete_referential_action = 1 THEN ' ON DELETE CASCADE'
                WHEN fk.delete_referential_action = 2 THEN ' ON DELETE SET NULL'
                WHEN fk.delete_referential_action = 3 THEN ' ON DELETE SET DEFAULT'
                ELSE ''
              END
            + CASE
                WHEN fk.update_referential_action = 1 THEN ' ON UPDATE CASCADE'
                WHEN fk.update_referential_action = 2 THEN ' ON UPDATE SET NULL'
                WHEN fk.update_referential_action = 3 THEN ' ON UPDATE SET DEFAULT'
                ELSE ''
              END
            + CHAR(13) + 'ALTER TABLE ' + @object_name + ' CHECK CONSTRAINT [' + fk.name
+ ']' + CHAR(13)
        FROM tempdb.sys.foreign_keys fk WITH (NOWAIT)
        JOIN tempdb.sys.objects ro WITH (NOWAIT) ON ro.[object_id] =
fk.referenced_object_id
        WHERE fk.parent_object_id = @object_id
        FOR XML PATH(N''), TYPE).value('.', 'NVARCHAR(MAX)')), '')
    + ISNULL(((SELECT
         CHAR(13) + 'CREATE' + CASE WHEN i.is_unique = 1 THEN ' UNIQUE' ELSE '' END
                + ' NONCLUSTERED INDEX [' + i.name + '] ON ' + @object_name + ' (' +
                STUFF((
                SELECT ', [' + c.name + ']' + CASE WHEN c.is_descending_key = 1 THEN '
DESC' ELSE ' ASC' END
                FROM index_column c
```

```
                    + ISNULL(CHAR(13) + 'INCLUDE (' +
                        STUFF((
                        SELECT ', [' + c.name + ']'
                        FROM index_column c
                        WHERE c.is_included_column = 1
                            AND c.index_id = i.index_id
                        FOR XML PATH(''), TYPE).value('.', 'NVARCHAR(MAX)'), 1, 2, '') +
')', '')  + CHAR(13)
            FROM tempdb.sys.indexes i WITH (NOWAIT)
            WHERE i.[object_id] = @object_id
                AND i.is_primary_key = 0
                AND i.[type] = 2
            FOR XML PATH(''), TYPE).value('.', 'NVARCHAR(MAX)')
        ), '')

SELECT @SQL
```

answered Aug 23 '17 at 14:09

Sal
2,679 3 15 38

0

Easiest way is to use the built-in feature of SQL Management Studio.

Right-click the database, go to tasks, Generate Scripts, and walk through the wizard. You can choose what objects to script, and it'll make it all for you.

Now if you are trying to make your OWN script to do the same thing, you're probably up for a lot of work...

answered Apr 1 '09 at 18:02

BradC
35.5k 12 63 90

This worked for me and saved me a TON of time and potential errors. It was unclear on what type of script the wizard would generate. It does indeed generate a CREATE TABLE script if you select the table. This answer was downvoted earlier. Any particular reason why? – jward01 Nov 23 '16 at 22:05

1 @jward01 *Shrug*. No idea, perhaps it would have received more votes with a detailed walk-through of the SSMS wizard, showing where you can choose to include options like indexes and foreign keys. – BradC Nov 28 '16 at 15:10

I can confirm in SSMS v18.0 preview (possibly earlier versions), you can open Object Explorer, expand your database, expand Programmability, expand Types, expand User-Defined Table Types, right-click your type and the "Script User-Defined Table Type as " option is there to choose the "CREATE To" script. – David Alan Condit Jan 11 '19 at 16:32

This is *not* the easiest way if you're trying to automate the process you're designing. – Chad Nov 15 '19 at 5:26

0

Since we're offering alternatives to what you asked..

If you're in .Net, you should look at the Database Publishing Wizard in Visual Studio. Easy way to script your tables/data to a text file.

madcolor
7,552 8 44 72

0

Use the SSMS, easiest way You can configure options for it as well (eg collation, syntax, drop...create)

Otherwise, SSMS Tools Pack, or DbFriend on CodePlex can help you generate scripts

answered Apr 1 '09 at 18:33

jerryhung
923 5 10

0

See my answer in this Question : How to generate create script of table using SQL query in SQL Server

Use this query :

```
DROP FUNCTION [dbo].[Get_Table_Script]
Go

Create Function Get_Table_Script
(
    @vsTableName varchar(50)
)

Returns
    VarChar(Max)
With ENCRYPTION

Begin

Declare @ScriptCommand varchar(Max)

Select @ScriptCommand =
    ' Create Table [' + SO.name + '] (' + o.list + ')'
    +
    (
        Case
        When TC.Constraint_Name IS NULL
            Then ''
        Else 'ALTER TABLE ' + SO.Name + ' ADD CONSTRAINT ' +
            TC.Constraint_Name  + ' PRIMARY KEY ' + ' (' + LEFT(j.List, Len(j.List)-1) +
')'
        End
    )
From sysobjects As SO
    Cross Apply

    (
        Select
            '  [' + column_name + '] ' +
             data_type +
             (
                Case data_type
                    When 'sql_variant'
                        Then ''
                    When 'text'
```

```
                    Else Coalesce( '(' +
                                    Case
                                        When character_maximum_length = -1
                                            Then 'MAX'
                                        Else Cast( character_maximum_length As
VarChar )
                                    End + ')' , ''
                                )
                End
            )
            + ' ' +
            (
                Case
                    When Exists (
                                    Select id
                                    From syscolumns
                                    Where
                                        ( object_name(id) = SO.name )
                                        And
                                        ( name = column_name )
                                        And
                                        ( columnproperty(id,name,'IsIdentity') = 1 )
                                )
                        Then 'IDENTITY(' +
                                Cast( ident_seed(SO.name) As varchar ) + ',' +
                                Cast( ident_incr(SO.name) As varchar ) + ')'

                    Else ''

                End
            ) + ' ' +

            (
                Case
                    When IS_NULLABLE = 'No'
                        Then 'NOT '
                    Else ''
                End
            ) + 'NULL ' +
            (
                Case
                    When information_schema.columns.COLUMN_DEFAULT IS NOT NULL
                        Then 'DEFAULT ' + information_schema.columns.COLUMN_DEFAULT
                    ELse ''
                End
            ) + ', '
        From information_schema.columns
        Where
            ( table_name = SO.name )
        Order by ordinal_position
        FOR XML PATH('')) o (list)

        Inner Join information_schema.table_constraints As TC On (
                                                                    ( TC.Table_name =
SO.Name )
                                                                    AND
                                                                    ( TC.Constraint_Type
= 'PRIMARY KEY' )
                                                                    And
                                                                    ( TC.TABLE_NAME =
@vsTableName )
                                                                )
        Cross Apply
            (
                Select '[' + Column_Name + '], '
                From  information_schema.key_column_usage As kcu
```

```
                FOR XML PATH('')
            ) As j (list)
Where
    ( xtype = 'U' )
    AND
    ( Name NOT IN ('dtproperties') )

Return @ScriptCommand

End
```

And you can fire this `Function` like this :

```
Select [dbo].Get_Table_Script '<Your_Table_Name>'
```

edited May 23 '17 at 12:02 — Community ♦ — 1 1

answered Sep 25 '13 at 13:47 — Ardalan Shahgholi — 8,322 11 84 109

0

First of all I love the script written by devart and I wanted to use it, but I found some limit, so I decided to improve it:

- I fixed the bug that limits the script at 4000 chars (it's still possible that some crazy table still exceeds the limits)
- I fixed the bug/limitation in case the table uses a nonclustered primary key
- I replaced '[' with quotename
- I added the name of the default constraints
- I changed the logic to identify the source table
- I added the possibility to drop and recreate the table and its FKs
- I added the possibility to generate specific attributes
- I replaced '' with N''

I didn't have time to test it properly and I tested it only on SQL Server 2012/4

Consider that the final print is still limited to 4000 chars, but the variable contains the full script.

Any comment will be appreciated.

This is my version of the code of DevArt:

```
DECLARE @object_id                      int;
DECLARE @SQL                            NVARCHAR(MAX) = N''
DECLARE @GenerateFKs                    bit = 1;
DECLARE @UseSourceCollation             bit = 1;
DECLARE @GenerateIdentity               bit = 1;
DECLARE @GenerateIndexes                bit = 1;
DECLARE @GenerateConstraints            bit = 1;
DECLARE @GenerateKeyConstraints         bit = 1;
DECLARE @AssignConstraintNameOfDefaults bit = 1;
DECLARE @AddDropIfItExists              bit = 1;

----------------------- PLEASE SET the table name here -----------------
```

```
;WITH index_column AS
(
    SELECT
          ic.[object_id]
        , ic.index_id
        , ic.is_descending_key
        , ic.is_included_column
        , c.name
    FROM sys.index_columns ic WITH (NOWAIT)
    JOIN sys.columns c WITH (NOWAIT) ON ic.[object_id] = c.[object_id] AND ic.column_id
= c.column_id
    WHERE ic.[object_id] = @object_id
),
fk_columns AS
(
     SELECT
          k.constraint_object_id
        , cname = c.name
        , rcname = rc.name
    FROM sys.foreign_key_columns k WITH (NOWAIT)
    JOIN sys.columns rc WITH (NOWAIT) ON rc.[object_id] = k.referenced_object_id AND
rc.column_id = k.referenced_column_id
    JOIN sys.columns c WITH (NOWAIT) ON c.[object_id] = k.parent_object_id AND
c.column_id = k.parent_column_id
    WHERE k.parent_object_id = @object_id and @GenerateFKs = 1
)
SELECT @SQL =
    -------------------  DROP IS Exists --------------------------------------------
-------------------------------------------------
        CASE WHEN @AddDropIfItExists = 1
        THEN
            --Drop table if exists
            CAST(
                N'IF OBJECT_ID(''' + quotename(OBJECT_schema_name(@object_id)) + N'.' +
quotename(OBJECT_NAME(@object_id)) + N''') IS NOT NULL DROP TABLE ' +
quotename(OBJECT_schema_name(@object_id)) + N'.' + quotename(OBJECT_NAME(@object_id)) +
N';' + NCHAR(13)
            as nvarchar(max))
            +
            --Drop foreign keys
            ISNULL(((
                SELECT
                    CAST(
                        N'ALTER TABLE ' + quotename(s.name) + N'.' + quotename(t.name) +
N' DROP CONSTRAINT ' + RTRIM(f.name) + N';' + NCHAR(13)
                    as nvarchar(max))
                FROM sys.tables t
                INNER JOIN sys.foreign_keys f ON f.parent_object_id = t.object_id
                INNER JOIN sys.schemas      s ON s.schema_id = f.schema_id
                WHERE f.referenced_object_id = @object_id
                FOR XML PATH(N''), TYPE).value(N'.', N'NVARCHAR(MAX)'))
            ,N'') + NCHAR(13)
        ELSE N'' END
    +
    -------------------- CREATE TABLE ----------------------------------------------
-------------------------------------------------------------
    CAST(
            N'CREATE TABLE ' + quotename(OBJECT_schema_name(@object_id)) + N'.' +
quotename(OBJECT_NAME(@object_id)) + NCHAR(13) + N'(' + NCHAR(13) + STUFF((
            SELECT
                CAST(
                    NCHAR(9) + N',' + quotename(c.name) + N' ' +
                    CASE WHEN c.is_computed = 1
                        THEN N' AS ' + cc.[definition]
                        ELSE UPPER(tp.name) +
                            CASE WHEN tp.name IN (N'varchar', N'char', N'varbinary',
```

```
                                    WHEN tp.name IN (N'nvarchar', N'nchar', N'ntext')
                                    THEN N'(' + CASE WHEN c.max_length = -1 THEN N'MAX'
ELSE CAST(c.max_length / 2 AS NVARCHAR(5)) END + N')'
                                    WHEN tp.name IN (N'datetime2', N'time2',
N'datetimeoffset')
                                    THEN N'(' + CAST(c.scale AS NVARCHAR(5)) + N')'
                                    WHEN tp.name = N'decimal'
                                    THEN N'(' + CAST(c.[precision] AS NVARCHAR(5)) +
N',' + CAST(c.scale AS NVARCHAR(5)) + N')'
                                ELSE N''
                            END +
                            CASE WHEN c.collation_name IS NOT NULL and
@UseSourceCollation = 1 THEN N' COLLATE ' + c.collation_name ELSE N'' END +
                            CASE WHEN c.is_nullable = 1 THEN N' NULL' ELSE N' NOT NULL'
END +
                            CASE WHEN dc.[definition] IS NOT NULL THEN CASE WHEN
@AssignConstraintNameOfDefaults = 1 THEN N' CONSTRAINT ' + quotename(dc.name) ELSE N''
END + N' DEFAULT' + dc.[definition] ELSE N'' END +
                            CASE WHEN ic.is_identity = 1 and @GenerateIdentity = 1 THEN
N' IDENTITY(' + CAST(ISNULL(ic.seed_value, N'0') AS NCHAR(1)) + N',' +
CAST(ISNULL(ic.increment_value, N'1') AS NCHAR(1)) + N')' ELSE N'' END
                        END + NCHAR(13)
                    AS nvarchar(Max))
            FROM sys.columns c WITH (NOWAIT)
                INNER JOIN sys.types tp WITH (NOWAIT) ON c.user_type_id =
tp.user_type_id
                LEFT JOIN sys.computed_columns cc WITH (NOWAIT) ON c.[object_id] = cc.
[object_id] AND c.column_id = cc.column_id
                LEFT JOIN sys.default_constraints dc WITH (NOWAIT) ON
c.default_object_id != 0 AND c.[object_id] = dc.parent_object_id AND c.column_id =
dc.parent_column_id
                LEFT JOIN sys.identity_columns ic WITH (NOWAIT) ON c.is_identity = 1 AND
c.[object_id] = ic.[object_id] AND c.column_id = ic.column_id
            WHERE c.[object_id] = @object_id
            ORDER BY c.column_id
            FOR XML PATH(N''), TYPE).value(N'.', N'NVARCHAR(MAX)'), 1, 2, NCHAR(9) + N'
')
    as nvarchar(max))
    +
    ---------------------- Key Constraints ----------------------------------------
------------------
    CAST(
        case when @GenerateKeyConstraints <> 1 THEN N'' ELSE
            ISNULL((SELECT NCHAR(9) + N', CONSTRAINT ' + quotename(k.name) + N' PRIMARY
KEY ' + ISNULL(kidx.type_desc, N'') + N'(' +
                        (SELECT STUFF((
                            SELECT N', ' + quotename(c.name) + N' ' + CASE WHEN
ic.is_descending_key = 1 THEN N'DESC' ELSE N'ASC' END
                            FROM sys.index_columns ic WITH (NOWAIT)
                            JOIN sys.columns c WITH (NOWAIT) ON c.[object_id] = ic.
[object_id] AND c.column_id = ic.column_id
                            WHERE ic.is_included_column = 0
                                AND ic.[object_id] = k.parent_object_id
                                AND ic.index_id = k.unique_index_id
                            FOR XML PATH(N''), TYPE).value(N'.', N'NVARCHAR(MAX)'), 1,
2, N''))
                + N')' + NCHAR(13)
                FROM sys.key_constraints k WITH (NOWAIT) LEFT JOIN sys.indexes kidx ON
                    k.parent_object_id = kidx.object_id and k.unique_index_id =
kidx.index_id
                WHERE k.parent_object_id = @object_id
                    AND k.[type] = N'PK'), N'') + N')'  + NCHAR(13)
        END
    as nvarchar(max))
    +
    -------------------- FOREIGN KEYS ----------------------------------------------
```

```
            SELECT NCHAR(13) +
             N'ALTER TABLE ' + + quotename(OBJECT_schema_name(@object_id)) + N'.' +
quotename(OBJECT_NAME(@object_id)) + + N' WITH'
            + CASE WHEN fk.is_not_trusted = 1
                THEN N' NOCHECK'
                ELSE N' CHECK'
              END +
              N' ADD CONSTRAINT ' + quotename(fk.name)  + N' FOREIGN KEY('
              + STUFF((
                SELECT N', ' + quotename(k.cname) + N''
                FROM fk_columns k
                WHERE k.constraint_object_id = fk.[object_id]
                FOR XML PATH(N''), TYPE).value(N'.', N'NVARCHAR(MAX)'), 1, 2, N'')
               + N')' +
              N' REFERENCES ' + quotename(SCHEMA_NAME(ro.[schema_id])) + N'.' +
quotename(ro.name) + N' ('
              + STUFF((
                SELECT N', ' + quotename(k.rcname) + N''
                FROM fk_columns k
                WHERE k.constraint_object_id = fk.[object_id]
                FOR XML PATH(N''), TYPE).value(N'.', N'NVARCHAR(MAX)'), 1, 2, N'')
               + N')'
            + CASE
                WHEN fk.delete_referential_action = 1 THEN N' ON DELETE CASCADE'
                WHEN fk.delete_referential_action = 2 THEN N' ON DELETE SET NULL'
                WHEN fk.delete_referential_action = 3 THEN N' ON DELETE SET DEFAULT'
                ELSE N''
              END
            + CASE
                WHEN fk.update_referential_action = 1 THEN N' ON UPDATE CASCADE'
                WHEN fk.update_referential_action = 2 THEN N' ON UPDATE SET NULL'
                WHEN fk.update_referential_action = 3 THEN N' ON UPDATE SET DEFAULT'
                ELSE N''
              END
            + NCHAR(13) + N'ALTER TABLE ' + + quotename(OBJECT_schema_name(@object_id))
+ N'.' + quotename(OBJECT_NAME(@object_id)) + + N' CHECK CONSTRAINT ' +
quotename(fk.name)  + N'' + NCHAR(13)
        FROM sys.foreign_keys fk WITH (NOWAIT)
        JOIN sys.objects ro WITH (NOWAIT) ON ro.[object_id] = fk.referenced_object_id
        WHERE fk.parent_object_id = @object_id
        FOR XML PATH(N''), TYPE).value(N'.', N'NVARCHAR(MAX)')), N'')
    as nvarchar(max))
    +
    ---------------------- INDEXES -----------------------------------------------------
----------------------------------------------------
    CAST(
        ISNULL(((SELECT
             NCHAR(13) + N'CREATE' + CASE WHEN i.is_unique = 1 THEN N' UNIQUE ' ELSE N'
' END
                    + i.type_desc + N' INDEX ' + quotename(i.name) + N' ON ' + +
quotename(OBJECT_schema_name(@object_id)) + N'.' + quotename(OBJECT_NAME(@object_id)) +
+ N' (' +
                    STUFF((
                    SELECT N', ' + quotename(c.name) + N'' + CASE WHEN
c.is_descending_key = 1 THEN N' DESC' ELSE N' ASC' END
                    FROM index_column c
                    WHERE c.is_included_column = 0
                        AND c.index_id = i.index_id
                    FOR XML PATH(N''), TYPE).value(N'.', N'NVARCHAR(MAX)'), 1, 2, N'') +
N')'
                    + ISNULL(NCHAR(13) + N'INCLUDE (' +
                        STUFF((
                        SELECT N', ' + quotename(c.name) + N''
                        FROM index_column c
                        WHERE c.is_included_column = 1
                            AND c.index_id = i.index_id
```

```
                WHERE i.[object_id] = @object_id
                    AND i.is_primary_key = 0
                    AND i.[type] in (1,2)
                    and @GenerateIndexes = 1
                FOR XML PATH(N''), TYPE).value(N'.', N'NVARCHAR(MAX)')
            ), N'')
        as nvarchar(max))

PRINT @SQL
SELECT datalength(@SQL), @sql
--EXEC sys.sp_executesql @SQL
```

edited Mar 26 '19 at 11:39

answered Mar 8 '19 at 17:26

ildanny

211 1 10

I'm getting an Invalid object error when I try to run this - `Invalid object name 'sys.Tables'.` Any ideas? – Nick Mar 21 '19 at 17:19

@Nick: sys.tables exists was introduces in SQL Server 2008. What version of SQL Server are you using? Can you execute "SELECT * from sys.tables" ? Have you tried to execute the version of the script written by DevArt? Do you have the same error? – ildanny Mar 24 '19 at 9:08

I am using SQL Server 2014, and no I can not execute `SELECT * FROM SYS.TABLES` I receive the same Invalid object name error as above. – Nick Mar 25 '19 at 16:19

"SELECT * from sys.tables" is different from "SELECT * FROM SYS.TABLES". It was meant to check the database collation. Are you using a CS (case sensitive) database collation? Are you logged in as dbo? Is it possible that the compatibility level of your database is SQL Server 2005 (sys.tables exists starting from 2008)? – ildanny Mar 26 '19 at 11:36

@Nick: I updated my script and changed sys.Tables to sys.table, let me know if you still have the problem. Thanks for your feedback :) – ildanny Mar 26 '19 at 11:40