How can I drop all indexes in a SQL database with one command?

18

5

So, how can I drop all indexes in a SQL database with one command? I have this command that will get me all the 20 or so drop statements, but how can I run all of those drop statements from this "result set"?

```
select * from vw_drop_idnex;
```

Another variation that gives me the same list is:

```
SELECT  'DROP INDEX ' + ix.Name + ' ON ' + OBJECT_NAME(ID)  AS QUERYLIST
FROM  sysindexes ix
WHERE   ix.Name IS NOT null and ix.Name like '%pre_%'
```

I tried to do "exec(select cmd from vw_drop_idnex)" and it didn't work. I am looking for something that works like a for loop and runs the queries one by one.

With Rob Farleys help, final draft of the script is:

```
declare @ltr nvarchar(1024);
SELECT @ltr = ( select 'alter table '+o.name+' drop constraint '+i.name+';'
  from sys.indexes i join sys.objects o on  i.object_id=o.object_id
  where o.type<>'S' and is_primary_key=1
  FOR xml path('') );
exec sp_executesql @ltr;

declare @qry nvarchar(1024);
select @qry = (select 'drop index '+o.name+'.'+i.name+';'
  from sys.indexes i join sys.objects o on  i.object_id=o.object_id
  where o.type<>'S' and is_primary_key<>1 and index_id>0
for xml path(''));
exec sp_executesql @qry
```

sql sql-server tsql

edited Sep 3 '09 at 18:18

asked Aug 28 '09 at 0:14

djangofan
16.3k 48 150 240

I think it's interesting that you added " AS QUERYLIST", when actually the answer you need shouldn't use that - as FOR XML PATH('') will concatenate strings in unnamed columns better (ie, without tags). – Rob Farley Aug 28 '09 at 0:23

Cool - you've got it sorted. I was basing my stuff on your sysindexes query, since most people don't have a copy of your 'vw_drop_idnex'. – Rob Farley Aug 29 '09 at 2:16

5 Answers

25 ✓

You're very close.

```
declare @qry nvarchar(max);
select @qry =
(SELECT  'DROP INDEX ' + ix.name + ' ON ' + OBJECT_NAME(ID) + '; '
FROM  sysindexes ix
WHERE   ix.Name IS NOT null and ix.Name like '%prefix_%'
for xml path(''));
exec sp_executesql @qry
```

answered Aug 28 '09 at 0:21

Rob Farley
13.8k 4 38 52

thanks a lot! i knew about that method but i was hoping for something that didnt require declaring a variable. i can make it work though. – djangofan Aug 28 '09 at 0:24

Also worth mentioning... you should probably consider the quotename function around your names, in case they have spaces in them. – Rob Farley Aug 28 '09 at 0:24

wow, that XML thing is cool. ;-) – djangofan Aug 28 '09 at 0:32

its not quite working. i get a message that says: Msg 102, Level 15, State 1, Line 1 Incorrect syntax near '<'. – djangofan Aug 28 '09 at 0:34

4 Given this is ~3 years later, i used this and fixed it by changing the select like to read: `SELECT 'DROP INDEX [' + ix.name + '] ON ' +`

6

this worked for me we skip sys indexes and for constraints

```
declare @qry nvarchar(max);
select @qry = (

    select  'IF EXISTS(SELECT * FROM sys.indexes WHERE name='''+ i.name +''' AND 
object_id = OBJECT_ID(''['+s.name+'].['+o.name+']''))      drop index ['+i.name+'] ON 
['+s.name+'].['+o.name+'];  '
    from sys.indexes i
        inner join sys.objects o on  i.object_id=o.object_id
        inner join sys.schemas s on o.schema_id = s.schema_id
    where o.type<>'S' and is_primary_key<>1 and index_id>0
    and s.name!='sys' and s.name!='sys' and is_unique_constraint=0
for xml path(''));

exec sp_executesql @qry
```

edited Apr 5 at 0:23
Graymatter
5,560 2 22 43

answered Dec 31 '15 at 9:25
Anatoli Klamer
1,161 1 9 16

0

None of the answers quite suited my needs.

I needed one that will also drop indexes that backup unique or primary constraints (except if these can't be dropped as they back up a foreign key)

```
DECLARE @SqlScript NVARCHAR(MAX);


SELECT @SqlScript = 
(
SELECT
'
BEGIN TRY
'+ CASE WHEN 1 IN (i.is_primary_key, i.is_unique_constraint) THEN
 '
 ALTER TABLE ' + QUOTENAME(OBJECT_SCHEMA_NAME(i.object_id)) + '.' + QUOTENAME(t.name) +
 '
 DROP INDEX ' + QUOTENAME(i.name)  + ' ON ' + QUOTENAME(OBJECT_SCHEMA_NAME(i.object_id))
+ '.' + QUOTENAME(t.name)
 END+'

END TRY
BEGIN CATCH
RAISERROR(''Could not drop %s on table %s'', 0,1, ' + QUOTENAME(i.name, '''') + ', ' +
QUOTENAME(t.name, '''') + ')
END CATCH
'
FROM sys.indexes i
JOIN sys.tables t ON i.object_id = t.object_id
WHERE i.type_desc IN ('CLUSTERED', 'NONCLUSTERED' )
ORDER BY t.object_id, i.index_id DESC
FOR XML PATH(''), TYPE
).value('.', 'NVARCHAR(MAX)');

--Return script that will be run
SELECT @SqlScript AS [processing-instruction(x)]
FOR XML PATH('');

EXEC (@SqlScript);
```

answered Jan 13 '18 at 19:12

Martin Smith

358k 61 603 712

1

Minor improvements to the accepted answer that I had to make in my own case, mostly to account for schemas:

```
declare @qry nvarchar(4000);
select @qry = (select 'drop index ['+s.name+'].['+o.name+'].['+i.name+'];'
  from sys.indexes i join sys.objects o on i.object_id=o.object_id join sys.schemas s on
o.schema_id=s.schema_id
  where o.type<>'S' and is_primary_key<>1 and index_id>0 and s.name<>'sys'
for xml path(''));
exec sp_executesql @qry
```

Also: In my case it couldn't run in one go because the script becomes longer than 4000 characters. The only way I could think of to deal with that was to put a "top 20" on the inner select and execute it multiple times

John
3,331 2 31 66

To get round the character count limitation, use `nvarchar(max)` as the datatype for `@qry` if you are on SQL 2005 or later. – Ed Harper Nov 3 '12 at 18:22

@EdHarper I tried that and it told me that type could be used as a variable. I'm on 2008R2 (Express). – John Nov 3 '12 at 23:35

5

From: Stephen Hill's Bloggie

```
DECLARE @indexName VARCHAR(128)
DECLARE @tableName VARCHAR(128)

DECLARE [indexes] CURSOR FOR

        SELECT          [sysindexes].[name] AS [Index],
                        [sysobjects].[name] AS [Table]

        FROM            [sysindexes]

        INNER JOIN      [sysobjects]
        ON              [sysindexes].[id] = [sysobjects].[id]

        WHERE           [sysindexes].[name] IS NOT NULL
        AND             [sysobjects].[type] = 'U'
        --AND               [sysindexes].[indid] > 1

OPEN [indexes]

FETCH NEXT FROM [indexes] INTO @indexName, @tableName

WHILE @@FETCH_STATUS = 0
BEGIN
        --PRINT 'DROP INDEX [' + @indexName + '] ON [' + @tableName + ']'
        Exec ('DROP INDEX [' + @indexName + '] ON [' + @tableName + ']')

        FETCH NEXT FROM [indexes] INTO @indexName, @tableName
END

CLOSE           [indexes]
DEALLOCATE      [indexes]
```

answered Sep 1 '11 at 19:10

Tawani

6,313 19 72 101

Excellent! It even includes a commented out print line for testing :) – Wouter Nov 15 '16 at 16:53

This will fail if there is one constraint on the table since those can only be dropoped using `DROP CONSTRAINT` , not `DROP INDEX` – tigrou Jan 4 '18 at 10:34