Build Dynamic SQL in a Stored Procedure - Essential SQL

After reading this article you will understand the basics of dynamic SQL; how to build statements based on variable values, and how to execute those constructed statements using sp_executesql and EXECUTE() from within a stored procedure.

All the examples for this lesson are based on Microsoft SQL Server Management Studio and the sample databases AdventureWorks and WideWorldImporters. You can get started using these free tools with my Guide *Getting Started Using SQL Server*.

Most SQL we write is written directly into the stored procedure. It is what is called **static SQL**. It is called this because it doesn't change. Once it is written, it's meaning is set, it's hammered into stone.

Below is an example of static SQL:

```
SELECT    JobTitle, Count(BusinessEntityID)
FROM      HumanResources.Employee
WHERE     Year(BirthDate) = 1970
GROUP BY  JobTitle

SELECT    JobTitle, Count(BusinessEntityID)
FROM      HumanResources.Employee
WHERE     Year(BirthDate) = 1971
GROUP BY JobTitle
```

Notice there are two statements, each returning a summary of JobTitles for a specific employee birth year. If we want to add more birth years, we need to add more statements. What if we only had to write the statement once and be able to change the year on-the-fly?

The code in these variables is then executed. Continuing with our example, here is the same code using dynamic SQL:

```
DECLARE @birthYear int = 1970
DECLARE @statement NVARCHAR(4000)

WHILE @birthYear <= 1971
BEGIN
   SET @statement = '
```

Build Dynamic SQL in a Stored Procedure - Essential SQL

```
   E)
   SE
END
```

The dynan
the SQL is
we'll expla

After reading this article you will understand the basics of dynamic SQL; how to build statements based on variable values, and how to execute those constructed statements using sp_executesql and EXECUTE() from within a stored procedure.

Introc

You can us
form is

All the examples for this lesson are based on Microsoft SQL Server Management Studio and the sample databases AdventureWorks and WideWorldImporters. You can get started using these free tools with my Guide *Getting Started Using SQL Server*.

```
EXECU
```

Most SQL we write is written directly into the stored procedure. It is what is called **static SQL**. It is called this because it doesn't change. Once it is written, it's meaning is set, it's hammered into stone.

In case you
procedure

Here is a s

Below is an example of static SQL:

```
DECLA
SET @
EXECU
```

```
SELECT    JobTitle, Count(BusinessEntityID)
FROM      HumanResources.Employee
WHERE     Year(BirthDate) = 1970
GROUP BY  JobTitle

SELECT    JobTitle, Count(BusinessEntityID)
FROM      HumanResources.Employee
WHERE     Year(BirthDate) = 1971
GROUP BY JobTitle
```

If you run

2018-01-2

Now that y
been aske
LineTotal b

Your boss
accept one
the sum.

Notice there are two statements, each returning a summary of JobTitles for a specific employee birth year. If we want to add more birth years, we need to add more statements. What if we only had to write the statement once and be able to change the year on-the-fly?

Of course, you could write this as two separate queries as shown in the following stored proc but that wouldn't be much fun, as it would be too much typing and prone to errors!

```
CREATE PROCEDURE uspCalcuateSalesSummaryStatic
@returnAverage bit
AS
```

```
IF (@returnAverage = 1)
BEGIN
   SELECT   SOD.ProductID,
            AVG(SOD.LineTotal) as ResultAvg
   FROM     Sales.SalesOrderDetail SOD
            INNER JOIN Sales.SalesOrderHEader SOH
                     ON SOH.SalesOrderID = SOD.SalesOrderID
   WH
   GR
END
ELSE
BEGIN
   SE

   FR


   WH
   GR
END
```

Build Dynamic SQL in a Stored Procedure - Essential SQL

After reading this article you will understand the basics of dynamic SQL; how to build statements based on variable values, and how to execute those constructed statements using sp_executesql and EXECUTE() from within a stored procedure.

The bad pa
much unic

With all thi
go for it!

All the examples for this lesson are based on Microsoft SQL Server Management Studio and the sample databases AdventureWorks and WideWorldImporters. You can get started using these free tools with my Guide *Getting Started Using SQL Server*.

```
CREAT

AS
DECLA
@func
IF (@
ELSE

SET @


EXECU
```

Most SQL we write is written directly into the stored procedure. It is what is called **static SQL**. It is called this because it doesn't change. Once it is written, it's meaning is set, it's hammered into stone.

Below is an example of static SQL:

```
SELECT    JobTitle, Count(BusinessEntityID)
FROM      HumanResources.Employee
WHERE     Year(BirthDate) = 1970
GROUP BY  JobTitle

SELECT    JobTitle, Count(BusinessEntityID)
FROM      HumanResources.Employee
WHERE     Year(BirthDate) = 1971
GROUP BY JobTitle
```

Here, inste
we build th

The SQL is
parameter
otherwise,

Notice there are two statements, each returning a summary of JobTitles for a specific employee birth year. If we want to add more birth years, we need to add more statements. What if we only had to write the statement once and be able to change the year on-the-fly?

You can see where the SQL is then built to create statement. Notice the color coding. It should correspond similar portions within the static version; this should help you do a comparison.

Debugging Dynamic SQL

You may be wonder what the SQL looks like at run time. You can easily inspect the code using the debugger:

Run the stored procedure using the debugger's run command, and then Step Into the code

Continue t

Build Dynamic SQL in a Stored Procedure - Essential SQL

After reading this article you will understand the basics of dynamic SQL; how to build statements based on variable values, and how to execute those constructed statements using sp_executesql and EXECUTE() from within a stored procedure.

All the examples for this lesson are based on Microsoft SQL Server Management Studio and the sample databases AdventureWorks and WideWorldImporters. You can get started using these free tools with my Guide *Getting Started Using SQL Server*.

Once you

displayed,

Most SQL we write is written directly into the stored procedure. It is what is called **static SQL**. It is called this because it doesn't change. Once it is written, it's meaning is set, it's hammered into stone.

The debug

more abou

Below is an example of static SQL:

Using

You can us

makes you

can be cor

The staten

```
EXECU
@parm
```

```
SELECT    JobTitle, Count(BusinessEntityID)
FROM      HumanResources.Employee
WHERE     Year(BirthDate) = 1970
GROUP BY  JobTitle

SELECT    JobTitle, Count(BusinessEntityID)
FROM      HumanResources.Employee
WHERE     Year(BirthDate) = 1971
GROUP BY JobTitle
```

So let's exp

Notice there are two statements, each returning a summary of JobTitles for a specific employee birth year. If we want to add more birth years, we need to add more statements. What if we only had to write the statement once and be able to change the year on-the-fly?

- @st
- @p

in @

type are separated by a space. Multiple parameters are separated by a comma.

Next we set the parameter values, by specifying the parameters and desired value. The parameters are listed in order defined within the @parameterDefinition string.

- @parm1 is the first parameter defined within the @parameterDefinition string. Value is the value, you wish to set it to.
- @parm2, is the second parameters, if defines, as declared in @parameterDefinition.
- and so on...

Here is a s

```
DECLA
DECLA

SET (
SET (

EXECU
```

The variou

- @st
  are
- @pa
- Par

To wrap up parameter

These para sp_execute

Exam

Let's take c query as w to works w

To make th query. We parameter

The updat parameter

```
CREAT


AS
DECLA
@para
@function NVARCHAR(10)

IF (@returnAverage = 1) SET @function = 'Avg'
ELSE SET @function = 'Sum'

SET @parameterDefinition = '@shipDateYear int'
SET @statement =
```

Build Dynamic SQL in a Stored Procedure - Essential SQL

After reading this article you will understand the basics of dynamic SQL; how to build statements based on variable values, and how to execute those constructed statements using sp_executesql and EXECUTE() from within a stored procedure.

All the examples for this lesson are based on Microsoft SQL Server Management Studio and the sample databases AdventureWorks and WideWorldImporters. You can get started using these free tools with my Guide *Getting Started Using SQL Server*.

Most SQL we write is written directly into the stored procedure. It is what is called **static SQL**. It is called this because it doesn't change. Once it is written, it's meaning is set, it's hammered into stone.

Below is an example of static SQL:

```
SELECT    JobTitle, Count(BusinessEntityID)
FROM      HumanResources.Employee
WHERE     Year(BirthDate) = 1970
GROUP BY  JobTitle

SELECT    JobTitle, Count(BusinessEntityID)
FROM      HumanResources.Employee
WHERE     Year(BirthDate) = 1971
GROUP BY JobTitle
```

Notice there are two statements, each returning a summary of JobTitles for a specific employee birth year. If we want to add more birth years, we need to add more statements. What if we only had to write the statement once and be able to change the year on-the-fly?

```
'SELECT   SOD.ProductID,' +
          @function +  + '(SOD.LineTotal) as Result' + @function + '
 FROM     Sales.SalesOrderDetail SOD
          INNER JOIN Sales.SalesOrderHEader SOH
                    ON SOH.SalesOrderID = SOD.SalesOrderID
 WHERE    YEAR(SOH.ShipDate) = @shipDateYear
 GROUP BY SOD.ProductID'
```

EXECU
@shi

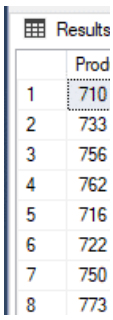
Build Dynamic SQL in a Stored Procedure - Essential SQL

To run this
using the f

EXECU

After reading this article you will understand the basics of dynamic SQL; how to build statements based on variable values, and how to execute those constructed statements using sp_executesql and EXECUTE() from within a stored procedure.

If you do s



All the examples for this lesson are based on Microsoft SQL Server Management Studio and the sample databases AdventureWorks and WideWorldImporters. You can get started using these free tools with my Guide *Getting Started Using SQL Server*.

Most SQL we write is written directly into the stored procedure. It is what is called **static SQL**. It is called this because it doesn't change. Once it is written, it's meaning is set, it's hammered into stone.

Let me sho
one param
and read:

Below is an example of static SQL:

```
SELECT    JobTitle, Count(BusinessEntityID)
FROM      HumanResources.Employee
WHERE     Year(BirthDate) = 1970
GROUP BY  JobTitle

SELECT    JobTitle, Count(BusinessEntityID)
FROM      HumanResources.Employee
WHERE     Year(BirthDate) = 1971
GROUP BY JobTitle
```

CREAT

AS
DECLA

IF ((
ELSE

SET (
SET (

Notice there are two statements, each returning a summary of JobTitles for a specific employee birth year. If we want to add more birth years, we need to add more statements. What if we only had to write the statement once and be able to change the year on-the-fly?

```
GROUP BY SOD.ProductID'

EXECUTE sp_executesql @statement, @parameterDefinition, @shipDate
```

Notice that the EXECUTE statement is much simpler, there is no need to assign the SQL statement parameter @shipDateYear to the store procedure parameter @shipDate's value.

This makes the statement more compact and easier to read. The flow seems to read better, as you don't have to mentally make connections between the stored procedure parameters and SQL parameters

Run [

You can al
command

```
EXECU
```

Here is a s

```
DECLA
SET @
EXECU
```

It is import
takes @sta
name of a

```
Msg 2
Could
```

Of course,
stored pro

sp_ex

You may b
between tl

Here are s
SQL:

- Witl
  This
  exis
- By u
  SQL
- It is

text which incorporates them.

- Parameterized queries are less prone to SQL injection attacks.

Build Dynamic SQL in a Stored Procedure - Essential SQL

After reading this article you will understand the basics of dynamic SQL; how to build statements based on variable values, and how to execute those constructed statements using sp_executesql and EXECUTE() from within a stored procedure.

All the examples for this lesson are based on Microsoft SQL Server Management Studio and the sample databases AdventureWorks and WideWorldImporters. You can get started using these free tools with my Guide *Getting Started Using SQL Server*.

Most SQL we write is written directly into the stored procedure. It is what is called **static SQL**. It is called this because it doesn't change. Once it is written, it's meaning is set, it's hammered into stone.

Below is an example of static SQL:

```
SELECT    JobTitle, Count(BusinessEntityID)
FROM      HumanResources.Employee
WHERE     Year(BirthDate) = 1970
GROUP BY  JobTitle

SELECT    JobTitle, Count(BusinessEntityID)
FROM      HumanResources.Employee
WHERE     Year(BirthDate) = 1971
GROUP BY JobTitle
```

Notice there are two statements, each returning a summary of JobTitles for a specific employee birth year. If we want to add more birth years, we need to add more statements. What if we only had to write the statement once and be able to change the year on-the-fly?

Kris Wenzel

Build Dynamic SQL in a Stored Procedure - Essential SQL

After reading this article you will understand the basics of dynamic SQL; how to build statements based on variable values, and how to execute those constructed statements using sp_executesql and EXECUTE() from within a stored procedure.

All the examples for this lesson are based on Microsoft SQL Server Management Studio and the sample databases AdventureWorks and WideWorldImporters. You can get started using these free tools with my Guide *Getting Started Using SQL Server*.

Most SQL we write is written directly into the stored procedure. It is what is called **static SQL**. It is called this because it doesn't change. Once it is written, it's meaning is set, it's hammered into stone.

Below is an example of static SQL:

```
SELECT    JobTitle, Count(BusinessEntityID)
FROM      HumanResources.Employee
WHERE     Year(BirthDate) = 1970
GROUP BY  JobTitle

SELECT    JobTitle, Count(BusinessEntityID)
FROM      HumanResources.Employee
WHERE     Year(BirthDate) = 1971
GROUP BY JobTitle
```

Notice there are two statements, each returning a summary of JobTitles for a specific employee birth year. If we want to add more birth years, we need to add more statements. What if we only had to write the statement once and be able to change the year on-the-fly?