The Learning House

427 S 4th Street #300

Louisville KY 40202

Stored Procedures

.NET Cohort

Coding Bootcamp

SOFTWARE GUILD

Lesson Goals

- Learn how to declare and use variables in SQL Server
- Learn how to take your SQL Statements, parameterize them, and store them for later execution

An Example

- Let's say we want to select some of the more expensive products from our Northwind database. We can do this:

```
SELECT *
FROM Products
WHERE UnitPrice > 50.00
```

The Need for Variables

However, if we wanted to change this query, we would have to re-write our WHERE clause. This is no big deal for a small query, but when we have more complex queries, it's better to put variables at the top so we can make changes in only one place.

```
-- Get all information about a customer
SELECT *
FROM Customers
WHERE CustomerID = 'BOTTM'

SELECT *
FROM Orders
WHERE CustomerID = 'BOTTM'
```

Declaring a Variable

- SQL uses the DECLARE command and prefixes all variable names with an @ symbol.
- Just like in C#, we must tell SQL Server what type of data the variable will hold.
- Assign data to a variable using the SET keyword.

```
DECLARE @CustomerID nchar(5)
SET @CustomerID = 'BOTTM'
```

Then we can use it!

This is called *parameterizing* the query.

```
DECLARE @CustomerID nchar(5)
SET @CustomerID = 'BOTTM'

-- Get all information about a customer
SELECT *
FROM Customers
WHERE CustomerID = @CustomerID

SELECT *
FROM Orders
WHERE CustomerID = @CustomerID
```

Another Example

```
DECLARE @CustomerID nchar(5)
SET @CustomerID = 'BOTTM'

DECLARE @MinOrderDate datetime
DECLARE @MaxOrderDate datetime

SET @MinOrderDate = '1/1/1997'
SET @MaxOrderDate = '12/31/1997'

-- Get all information about a customer
SELECT *
FROM Customers
WHERE CustomerID = @CustomerID

SELECT *
FROM Orders
WHERE CustomerID = @CustomerID
    AND Orderdate BETWEEN @MinOrderDate AND @MaxOrderDate
```

Lab Exercise

1. Write a parameterized query to select the grant information from SWCCorp's Grant table to show grants between a minimum and maximum value.

Stored Procedures

- A stored procedure is a single or batch of SQL statements that is saved to be executed later.
- Using stored procedures is something most good DBAs try to push.

Why Stored Procedures?

- Security: You can grant access to them separately from tables.
- Performance: If your SQL is in code, the DBA can’t get to it to tune it.
- Maintenance: You can change the internals of a stored procedure without having to redeploy an application.
- Common Interface: It is easier to prevent duplicate SQL code if it isn’t spread throughout your code files.

Why Not Stored Procedures?

- Small applications that are driven by ORM tools
- Applications (usually commercial products) that need to support many databases… stored procedures aren’t cross-database compatible.
- Application does not benefit from added security (full-trust applications)

Regardless…

- Whether you use stored procedures or write SQL in your code, ALWAYS ALWAYS ALWAYS parameterize your queries.

SQL Injection Attacks

- If you don’t parameterize queries (and just concatenate user strings into your query strings), then malicious users can put in escape codes and inject custom SQL into your queries.
- If you application then runs the code, they can do all sorts of nasty stuff like query your users’ data, add new logins, and otherwise steal or corrupt your data.

How Serious is SQL Injection?

- It is the #1 most common attack vector of web applications…
- Ever heard of a company called Sony? Their Playstation network hack that compromised all the accounts and billing information came from a SQL Injection vulnerability.

Example of Injection

Consider this unparameterized query:

```
Console.WriteLine("Enter a user to retrieve: ");
string userInput = Console.ReadLine();
string query = "SELECT * FROM userinfo WHERE name = '" + userInput + "'";
```

What do you suppose happens if the user puts in the following:

‘ OR 1=1;DROP TABLE UserInfo;SELECT ‘a

Or this:

a’; SELECT * FROM sys.Tables;SELECT ‘a

Word to the Wise

- Parameterized queries and stored procedures protect against injection.
- We will learn how to parameterize SQL in code when we do ADO.NET.

Creating a Stored Procedure

```
CREATE PROCEDURE GetCustomerOrdersByDate
(
    @CustomerID nchar(5),
    @MinOrderDate datetime,
    @MaxOrderDate datetime
) AS

-- Get all information about a customer
SELECT *
FROM Customers
WHERE CustomerID = @CustomerID

SELECT *
FROM Orders
WHERE CustomerID = @CustomerID
    AND Orderdate BETWEEN @MinOrderDate AND @MaxOrderDate

GO
```

Name

Parameter List

SQL to execute

Do it!

No Parameters?

- Just remove the parameter list:

```
CREATE PROCEDURE GetAllCustomers AS

SELECT *
FROM Customers

GO
```

Calling a Stored Procedure

- In SQL, we can call a stored procedure by *executing* it using the EXEC keyword:

exec <name> <params>

```
exec GetAllCustomers

exec GetCustomerOrdersByDate 'BOTTM', '1/1/1997', '12/31/1997'
```

We Can Pass Variables to Stored Procedures, too

```
DECLARE @CustomerID nchar(5)
SET @CustomerID = 'BOTTM'

exec GetCustomerOrdersByDate @CustomerID, '1/1/1997', '12/31/1997'
```

Or pass parameters out of order by specifying the parameter name in the exec statement:

```
exec GetCustomerOrdersByDate @MaxOrderDate= '12/31/1997',
    @MinOrderDate='1/1/1997', @CustomerId = 'BOTTM'
```

Changing a Stored Procedure

- Use the ALTER command to change a stored procedure. You can add/remove parameters and modify the SQL.
  - Or you could just drop and recreate.

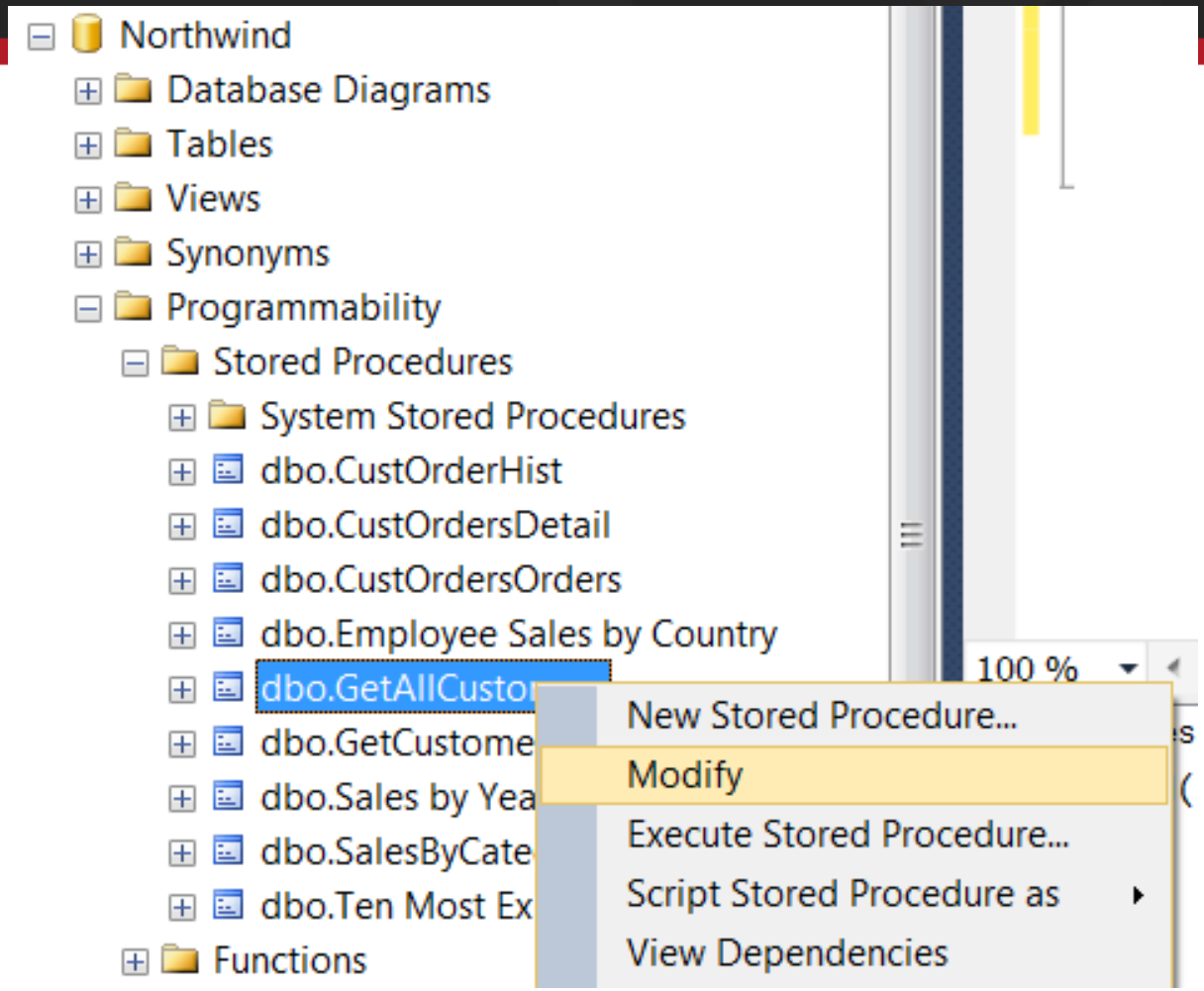
```
ALTER PROCEDURE GetAllCustomers AS

SELECT *
FROM Customers
ORDER BY CompanyName
```

Easy Alter

Right-click the stored procedure name and choose “Modify.”

We can also generate an execute statement from here.

“New Stored Procedure” will create a template for you to fill in.



SOFTWARE GUILD

Lab Exercise!

1. SWCCorp needs a stored procedure called GetProductListByCategory that should take in the category name as a parameter.
2. SWCCorp needs a stored procedure called GetGrantsByEmployee that takes in the LastName as a parameter and returns the EmployeeId, FirstName, LastName, GrantName, and Amount for that employee's grants.
3. Write an update and an insert procedure for the SWCCorp Grant table take takes a grantID as a parameter.

Fin

In most enterprise software, stored procedures will be the de facto storage for your SQL queries.