/\* Create your first stored procedure.

Use the statement CREATE PROCEDURE, give it a name

Code goes inside of the BEGIN END block \*/

CREATE PROCEDURE GetAllEmployees AS

BEGIN

print 'Hello World' /\* code goes here\*/

/\* Execute your stored procedure using EXEC followed by

the name of the procedure\*/

EXEC GetAllEmployees

/\* To change exisiting stored procedure use ALTER followed by

the name of the procedure\*/

ALTER to make modifications

ALTER PROCEDURE GetAllEmployees AS

BEGIN

SELECT \* FROM employee

END;

EXEC GetAllEmployees

/\* To create stored procedure that takes input parameters, you must

specify the name of the input parameters and the data type.

In the example below the parameter name is @dept\_no, and data type is varchar(5)

Note: all parameter names must begin with @ symbol

\*/

ALTER PROCEDURE GetAllEmployees @dept\_no varchar(5) AS

BEGIN

SELECT emp\_fname, emp\_lname, dept\_no

FROM employee

WHERE dept\_no = @dept\_no

END;

/\* To execute a stored procedure that takes input parameters, you must

specify the name EXEC ProcedureName ‘input value(s)’

Note: All character strings must be defined in single quote. See below

\*/

EXEC GetAllEmployees 'D1'

/\* Examples above show procedure that output a result set. The example below show procedure that takes input parameters and makes changes to the data.

\*/

ALTER PROCEDURE [dbo].[ChangeEmployeeDept] @emp\_no INT, @new\_depar\_no varchar(5)

AS

BEGIN

UPDATE employee set dept\_no =@new\_depar\_no

WHERE emp\_no = @emp\_no

END

/\* Often times variables, as in many programming languages, need to be defined.

In Procedure global variables are defined before the BEGIN END block. Local variables can also be created within local block scope.

\*/

CREATE PROCEDURE GetLastEmployee AS

DECLARE @MaxEmpNo int

BEGIN

SELECT @MaxEmpNo = MAX(emp\_no) FROM employee;

SELECT emp\_fname, emp\_lname, dept\_no

FROM employee

WHERE emp\_no =@MaxEmpNo

END;

exEC GetLastEmployee

/\* User Define Functions must take input parameters and must return a result or results back.

The syntax is CREATE FUNCTION FunctionName (ParamNames dataTypes) RETURNS datatype

See an example below\*/

CREATE FUNCTION AmazingSum

(

@a int, @b int

)

RETURNS int

AS

BEGIN

RETURN @a + @b

END

/\* Unlike procedure, functions are called from a SELECT query

\*/

select dbo.AmazingSum(1,2)

/\* Example below shows a function that does a lookup using values stored in a table

\*/

alter FUNCTION GetEmployeeName

(

@emp\_no int

)

RETURNS varchar(100)

AS

BEGIN

DEclare @emp\_name varchar(100)

SELECT @emp\_name = e.emp\_fname + ' ' + e.emp\_lname from employee e WHERE e.emp\_no = @emp\_no

RETURN @emp\_name

END

GO

/\* Functions are executed within a query as shown below

\*/

select \*,

dbo.GetEmployeeName(emp\_no) as FullName

from employee