```
'************************
Author: M Samiruzzaman
'************************
' USE: This is working version of code. Use directly or convert to any other language
' or change as you wish to fit with your own needs
' It is free to use.


'*******************************
'** VB Script function can be used directly or by converting to preferred language
Function SQLSafe(byVal S)
'*******************************

Dim cleanstr : cleanstr = ""

SQLSafe = ""
S = "" & S

If Len(S) = 0 Then
        Exit Function
Else
        If InStr(UCase(S), "') OR '") > 0 Then Exit Function
        If InStr(UCase(S), "') AND '") > 0 Then Exit Function
End If

cleanstr = Replace(S, "'", "''", 1, -1, 1)
cleanstr = Replace(cleanstr, "--", "-", 1, -1, 1)
cleanstr = Replace(cleanstr, "xp_cmdshell", "", 1, -1, 1)
cleanstr = Replace(cleanstr, "xp_log70.dll", "", 1, -1, 1)
cleanstr = Replace(cleanstr, "sp_makewebtask", "", 1, -1, 1)
cleanstr = Replace(cleanstr, "openrowset", "", 1, -1, 1)
cleanstr = Replace(cleanstr, "d99_tmp", "", 1, -1, 1)

cleanstr = Trim(cleanstr)

CleanSQL = cleanstr

End Function




'*************************************************************************************
*************
'** VB Script function can be used directly or by converting to preferred language
Function IsValidStringMinMax(ByRef varTemp, ByVal intMinLength, ByVal intMaxLength)
'*************************************************************************************
*************

'** If intMaxLength is zero then it is optional

'** Assume that the inbound parameter is in the correct format
IsValidStringMinMax = True

'** Test if the parameter is a valid string subtype
On Error Resume Next
varTemp = CStr(varTemp)
If Err.Number <> 0 Then
        IsValidStringMinMax = False
        On Error GoTo 0
        Exit Function
End If

'** No error should occur any more
```

```
On Error GoTo 0

'** Check maxlength is optional
If 0 = intMaxLength Then intMaxLength = Len(varTemp)

'** Test the minimum and maximum field lengths
If (Len(varTemp) < intMinLength Or Len(varTemp) > intMaxLength) Then
        IsValidStringMinMax = False
        Exit Function
End If

End Function




// ---------------------------------------------------------------------------
//JavaSCript function can be used directly from client side
function IsValidText(field, iMinLen, iMaxLen, regexp)
{
        var strCheck, len;
        field.value = strCheck = field.value.trim();
        len = strCheck.length;
        if ((len >= iMinLen && len <= iMaxLen) && (!len || regexp.test(strCheck)))
                return true;
        return false;
}

// ---------------------------------------------------------------------------
//JavaSCript function can be used directly from client side
String.prototype.trim = function ()
{
        return this.replace(/^\s+|\s+$/g, '');
}



//-----------------------------------------------------------------------------
//JavaSCript function can be used directly from client side
function CrossScriptingTestJS(){
        var strQsFrmData =
window.location.search.substring(1,window.location.search.length);
        var keywords =
[/script/i,/sysdatabase/i,/sysobject/i,/iframe/i,/frame/i,/alert/i,/video/i];

        if(strQsFrmData.length > 0){
                strQsFrmData = strQsFrmData.replace(/%20/g," ");
                strQsFrmData = strQsFrmData.replace(/%27/g,"'");
                strQsFrmData = strQsFrmData.replace(/%29/g,")");
                strQsFrmData = strQsFrmData.toLowerCase();

                if(MatchChecker()) return true;
        }

        var arrForms = document.forms;
        var objForm;
        var objElement;
        if(arrForms){
                for(var i=0;i<arrForms.length;i++){
                        objForm = arrForms[i];
                        var arrElements = objForm.elements;
                        if(arrElements){
                                for(var j=0;j<arrElements.length;j++){
                                        objElement = arrElements[j];
                                        if(objElement.type == "text" || objElement.type ==
```

```
"textarea"){
                                                strQsFrmData = objElement.value;

                                                if(strQsFrmData.length > 0){
                                                        strQsFrmData =
strQsFrmData.replace(/%20/g," ");
                                                        strQsFrmData =
strQsFrmData.replace(/%27/g,"'");
                                                        strQsFrmData =
strQsFrmData.replace(/%29/g,")");
                                                        strQsFrmData =
strQsFrmData.toLowerCase();

                                                        if(MatchChecker()) return true;
                                                }
                                        }
                                }
                        }
                }
        }

        function MatchChecker(){
                for(var i=0;i<keywords.length;i++){
                        if(keywords[i].test(/select/i)){
                                if(strQsFrmData.search(keywords[i]) > -1 &&
strQsFrmData.search(keywords[i+1]) == -1 && strQsFrmData.search(keywords[i+2]) == -1)
                                        return true;
                                else i += 2;
                        }
                        else{
                                if(strQsFrmData.search(keywords[i]) > -1)
                                        return true;
                        }
                }
                return false;
        }

        return false;
}

//JavaSCript function can be used directly from client side
function HackingAttempt(){
        if (CrossScriptingTestJS()){
                booSuccessfullValidation = false
        }
        else{
                booSuccessfullValidation = true
        }
}


'****************************
'** VB Script function can be used directly or by converting to preferred language
Function CrossScriptingTest()
'****************************

Dim strQsFrmData                        ' Data from query or form

CrossScriptingTest = False

If Len(Request.QueryString()) > 0 Then

        strQsFrmData = "" & Request.QueryString()
        strQsFrmData = Replace(strQsFrmData, "%20", " ")
        strQsFrmData = Replace(strQsFrmData, "%27", "'")
```

```
        strQsFrmData = Replace(strQsFrmData, "%29", ")")
        strQsFrmData = LCase(strQsFrmData)

    If InStr(strQsFrmData, "script") > 0 Then
                Response.AppendToLog(strQsFrmData)
        CrossScriptingTest = True
                Exit Function
    End If
    If InStr(strQsFrmData, "sysdatabase") > 0 Then
                Response.AppendToLog(strQsFrmData)
        CrossScriptingTest = True
                Exit Function
    End If
    If InStr(strQsFrmData, "sysobject") > 0 Then
                Response.AppendToLog(strQsFrmData)
        CrossScriptingTest = True
                Exit Function
    End If
    If InStr(strQsFrmData, "char(") > 0 Then
                Response.AppendToLog(strQsFrmData)
        CrossScriptingTest = True
                Exit Function
    End If
    If InStr(strQsFrmData, "varchar") > 0 Then
                Response.AppendToLog(strQsFrmData)
        CrossScriptingTest = True
                Exit Function
    End If
    If InStr(strQsFrmData, "select") > 0 Then
                Response.AppendToLog(strQsFrmData)
        CrossScriptingTest = True
                Exit Function
    End If
    If InStr(strQsFrmData, "alert") > 0 Then
                Response.AppendToLog(strQsFrmData)
        CrossScriptingTest = True
                Exit Function
    End If
    If InStr(strQsFrmData, "\x00") > 0 Then
                Response.AppendToLog(strQsFrmData)
        CrossScriptingTest = True
                Exit Function
    End If
    If InStr(strQsFrmData, "\x20") > 0 Then
                Response.AppendToLog(strQsFrmData)
        CrossScriptingTest = True
                Exit Function
    End If
        If InStr(strQsFrmData, "') or '") > 0 Then
                Response.AppendToLog(strQsFrmData)
        CrossScriptingTest = True
                Exit Function
    End If
        If InStr(strQsFrmData, "') and '") > 0 Then
                Response.AppendToLog(strQsFrmData)
        CrossScriptingTest = True
                Exit Function
    End If
End If

If Request.ServerVariables("REQUEST_METHOD") = "POST" And UCase(TypeName(Request.Form()))
<> UCase("IRequestDictionary") Then
    If Len(Request.Form()) > 0 Then

                strQsFrmData = "" & Request.Form()
```

```
                strQsFrmData = Replace(strQsFrmData, "%20", " ")
                strQsFrmData = Replace(strQsFrmData, "%27", "'")
                strQsFrmData = Replace(strQsFrmData, "%29", ")")
                strQsFrmData = LCase(strQsFrmData)

        If InStr(strQsFrmData, "script") > 0 Then
                        Response.AppendToLog(strQsFrmData)
            CrossScriptingTest = True
            Exit Function
        End If
        If InStr(strQsFrmData, "sysdatabase") > 0 Then
                        Response.AppendToLog(strQsFrmData)
            CrossScriptingTest = True
            Exit Function
        End If
        If InStr(strQsFrmData, "sysobject") > 0 Then
                        Response.AppendToLog(strQsFrmData)
            CrossScriptingTest = True
            Exit Function
        End If
        If InStr(strQsFrmData, "char(") > 0 Then
                        Response.AppendToLog(strQsFrmData)
            CrossScriptingTest = True
            Exit Function
        End If
        If InStr(strQsFrmData, "varchar") > 0 Then
                        Response.AppendToLog(strQsFrmData)
            CrossScriptingTest = True
            Exit Function
        End If
        If InStr(strQsFrmData, "select") > 0 Then
                        Response.AppendToLog(strQsFrmData)
            CrossScriptingTest = True
            Exit Function
        End If
        If InStr(strQsFrmData, "alert") > 0 Then
                        Response.AppendToLog(strQsFrmData)
            CrossScriptingTest = True
            Exit Function
        End If
        If InStr(strQsFrmData, "\x00") > 0 Then
                        Response.AppendToLog(strQsFrmData)
            CrossScriptingTest = True
            Exit Function
        End If
        If InStr(strQsFrmData, "\x20") > 0 Then
                        Response.AppendToLog(strQsFrmData)
            CrossScriptingTest = True
            Exit Function
        End If
                If InStr(strQsFrmData, "') or '") > 0 Then
                        Response.AppendToLog(strQsFrmData)
                        CrossScriptingTest = True
                        Exit Function
                End If
                If InStr(strQsFrmData, "') and '") > 0 Then
                        Response.AppendToLog(strQsFrmData)
                        CrossScriptingTest = True
                        Exit Function
                End If
    End If
End If

End Function
```

```
'********************
'** VB Script Sub routine can be used directly or by converting to preferred language
Sub HTTPRedirection()
'********************

'** Be wary that this function truncates querystrings
If Request.ServerVariables("SERVER_PORT") = 80 And Session("IsLiveSite") Then
        If InStr(LCase(Request.ServerVariables("HTTP_REFERER")), "http:") > 0 Then
                Response.Redirect(Replace(Request.ServerVariables("HTTP_REFERER"), "http:",
"https:"))
        End If
End If

End Sub

'*************************************
'** VB Script Sub routine can be used directly or by converting to preferred language
Function IsValidBoolean(ByRef varTemp)
'*************************************

'** Assume that the inbound parameter is not in the correct format
IsValidBoolean = False

'** Test if the parameter is a valid boolean subtype
On Error Resume Next
varTemp = CBool(varTemp)
If Err.Number = 0 And Len(varTemp) > 0 Then IsValidBoolean = True
On Error GoTo 0

End Function

'*************************************
Function IsValidByte(ByRef varTemp)
'*************************************

'** Assume that the inbound parameter is not in the correct format
IsValidByte = False

'** Test if the parameter is a valid byte subtype
On Error Resume Next
varTemp = CByte(varTemp)
If Err.Number = 0 Then IsValidByte = True
On Error GoTo 0

End Function

'*************************************
'** VB Script function routine can be used directly or by converting to preferred language
Function IsValidCurrency(ByRef varTemp)
'*************************************

'** Assume that the inbound parameter is not in the correct format
IsValidCurrency = False

'** Test if the parameter is a valid currency subtype
On Error Resume Next
varTemp = CCur(varTemp)
If Err.Number = 0 Then IsValidCurrency = True
On Error GoTo 0

End Function

'*************************************
'** VB Script function routine can be used directly or by converting to preferred language
Function IsValidDate(ByRef varTemp)
```

```
'**************************************

'** Assume that the inbound parameter is not in the correct format
IsValidDate = False

'** Test if the parameter is a valid date subtype
On Error Resume Next
varTemp = CDate(varTemp)
If Err.Number = 0 Then IsValidDate = True
On Error GoTo 0

End Function

'**************************************
'** VB Script function routine can be used directly or by converting to preferred language
Function IsValidDouble(ByRef varTemp)
'**************************************

'** Assume that the inbound parameter is not in the correct format
IsValidDouble = False

'** Test if the parameter is a valid double subtype
On Error Resume Next
varTemp = CDbl(varTemp)
If Err.Number = 0 Then IsValidDouble = True
On Error GoTo 0

End Function

'**************************************
'** VB Script function routine can be used directly or by converting to preferred language
Function IsValidSmallInteger(ByRef varTemp)
'**************************************

'** Assume that the inbound parameter is not in the correct format
IsValidSmallInteger = False

'** Test if the parameter is a valid small integer subtype
On Error Resume Next
varTemp = CInt(varTemp)
If Err.Number = 0 Then IsValidSmallInteger = True
On Error GoTo 0

End Function

'**************************************
'** VB Script function routine can be used directly or by converting to preferred language
Function IsValidLongInteger(ByRef varTemp)
'**************************************

'** Assume that the inbound parameter is not in the correct format
IsValidLongInteger = False

'** Test if the parameter is a valid long integer subtype
On Error Resume Next
varTemp = CLng(varTemp)
If Err.Number = 0 Then IsValidLongInteger = True
On Error GoTo 0

End Function

'**************************************
'** VB Script function routine can be used directly or by converting to preferred language
Function IsValidSingle(ByRef varTemp)
'**************************************
```

```
'** Assume that the inbound parameter is not in the correct format
IsValidSingle = False

'** Test if the parameter is a valid single subtype
On Error Resume Next
varTemp = CSng(varTemp)
If Err.Number = 0 Then IsValidSingle = True
On Error GoTo 0

End Function

'**************************************
'** VB Script function routine can be used directly or by converting to preferred language
Function IsValidString(ByRef varTemp)
'**************************************

'** Assume that the inbound parameter is not in the correct format
IsValidString = False

'** Test if the parameter is a valid string subtype
On Error Resume Next
varTemp = CStr(varTemp)
If Err.Number = 0 Then IsValidString = True
On Error GoTo 0

End Function

'****************************************************************************
'** VB Script function routine can be used directly or by converting to preferred language
Function IsValidStringMinMax(ByRef varTemp, ByVal intMinLength, ByVal intMaxLength)
'****************************************************************************

'** If intMaxLength is zero then it is optional

'** Assume that the inbound parameter is in the correct format
IsValidStringMinMax = True

'** Test if the parameter is a valid string subtype
On Error Resume Next
varTemp = CStr(varTemp)
If Err.Number <> 0 Then
        IsValidStringMinMax = False
        On Error GoTo 0
        Exit Function
End If

'** No error should occur any more
On Error GoTo 0

'** Check maxlength is optional
If 0 = intMaxLength Then intMaxLength = Len(varTemp)

'** Test the minimum and maximum field lengths
If (Len(varTemp) < intMinLength Or Len(varTemp) > intMaxLength) Then
        IsValidStringMinMax = False
        Exit Function
End If

End Function


'******************************
'** VB Script function routine can be used directly or by converting to preferred language
Sub CheckLoginAttempt()
```

```
'*******************************
Dim booLogin                            ' Does login or not

booLogin = False
If Len(Session("AgtLoginAttempt")) = 0 Then Session("AgtLoginAttempt") = 0

If IsNumeric(Session("PatronID")) Then
        If Session("PatronID") > 0 Then
                Session("AgtLoginAttempt") = 0
                Exit Sub
        End If
End If

If Not booLogin Then Session("AgtLoginAttempt") = Session("AgtLoginAttempt") + 1

Call CheckAccoutStatus()

End Sub

'*******************************
'** VB Script function routine can be used directly or by converting to preferred language
Sub CheckAccoutStatus()
'*******************************

If Len(Session("AgtLoginAttempt")) = 0 Then Session("AgtLoginAttempt") = 0

If Session("AgtLoginAttempt") > 10 Then
        Session("ErrorType") = "Excessive Login Attempt"
        Response.Redirect "helppage.asp"
End If

End Sub



'*****************************************************************************
'** VB Script function routine can be used directly or by converting to preferred language
Function SanitiseInput(ByVal strInput, ByVal strAllowedCharacters)
'*****************************************************************************
'** This function is for cleaning sensitive characters that likely to occur XSS attack

Dim arrOfInvalidCharacters                                              ' Array of
invalid characters likely to happen XSS attacks
Dim intLoopCounter                                                              '
Loop Counter
Dim strThisCharacter                                                    ' Current
character
Dim intNumberOfInvalidCharacters                                ' Total number of
invalid characters

strInput = Trim(strInput)
If  Len(strInput) = 0 Then
        SanitiseInput = strInput
        Exit Function
End If

arrOfInvalidCharacters = Array("<",">","(",")","'", """" , "/","\","*",";",":","=","
{","}","`","%","+","^","!","-","\x00","\x20")
intNumberOfInvalidCharacters = UBound(arrOfInvalidCharacters)

For intLoopCounter = 0 To intNumberOfInvalidCharacters

        strThisCharacter = arrOfInvalidCharacters( intLoopCounter)
        If InStr(strAllowedCharacters, strThisCharacter) = 0 Then strInput =
Replace(strInput, strThisCharacter, " ")
```

```
Next

SanitiseInput = strInput

End Function


'** VB.NET function as an example of stored procedure calling from middle tier function
'** Length check and data type check are done explicitly from middle tier to call SP
instead of querying the DB directly
Public Sub Customer_INSERT(ByRef AgtID As String, ByVal AgtName As String, ByVal
AgtNameSoundex As String, _
        ByVal AgtNameCaps As String, ByVal AgtTitle As String, ByVal AgtFirstName As
String, _
        ByVal AgtFirstNameCaps As String, ByVal AgtOtherInitials As String, ByVal
AgtHonours As String, _
        ByVal AgtGender As String, ByVal AgtType As Integer, ByVal AgtStatus As Integer, _
        ByVal AgtInvoicePeriod As String, ByVal AgtMaritalStatus As Integer, ByVal
AgtDateOfBirth As Date, _
        ByVal AgtVATCode As String, ByVal AgtAddressForm As String, ByVal AgtDearForm As
String, ByVal AgtBodyForm As String, _
        ByVal PatCode As String, ByVal PatSubCode As String, ByVal PscID As String, ByVal
AgtCreditLimit As Decimal, _
        ByVal AgtWarningLevel As Byte, ByVal AgtOldPassword As String, ByVal AgtNewPassword
As String, _
        ByVal AgtDataProtectFlag As Boolean, ByVal AggID As Int32, ByVal AgtJobTitle As
String, _
        ByVal AgtSignOffForm As String, ByVal AgtOffCode As Integer, ByVal SlaCreditTerms
As Integer, _
        ByVal SlaInvoiceContact As String, ByVal SlaStatementContact As String, ByVal
WebAccountStatus As Byte, _
        ByVal AgtCanGiftAid As Boolean, ByVal AemEmailAddress As String, ByVal
SlaChargeAccount As Byte, _
        ByVal AgtAccountLevel As Byte, ByVal SlaInvoiceType As Byte, ByVal SlaBookingUnpaid
As Byte, _
        ByVal SlaInvoiceNonAccount As Byte, ByVal DplID As Int32, ByVal SlaOnMaturityType
As Byte, _
        ByVal SlaOnMaturityOffSet As Short, ByVal AgtBulkMailing As Integer, ByVal
SlaInvoiceLevel As Byte, _
        ByVal AgtCommVatCode As String)

        Dim objSQLCommand As New SqlCommand
        Dim objSQLParameter As SqlParameter

        Try
            objSQLCommand.Connection = m_objSQLConnection
                        'Comment:  m_objSQLConnection is the connection that user/developer
can use of their own one
            objSQLCommand.CommandType = CommandType.StoredProcedure
            objSQLCommand.CommandText = "Customer_INSERT"
            objSQLParameter.Direction = ParameterDirection.InputOutput
            objSQLParameter = objSQLCommand.Parameters.Add("@AgtID", SqlDbType.VarChar, 20)
            objSQLParameter.Value = AgtID
            objSQLParameter.Direction = ParameterDirection.InputOutput
            objSQLCommand.Parameters.Add("@AgtName", SqlDbType.VarChar, 50).Value = AgtName
            objSQLCommand.Parameters.Add("@AgtNameSoundex", SqlDbType.VarChar, 10).Value =
AgtNameSoundex
            objSQLCommand.Parameters.Add("@AgtNameCaps", SqlDbType.VarChar, 50).Value =
AgtNameCaps
            objSQLCommand.Parameters.Add("@AgtTitle", SqlDbType.VarChar, 30).Value =
AgtTitle
            objSQLCommand.Parameters.Add("@AgtFirstName", SqlDbType.VarChar, 30).Value =
AgtFirstName
            objSQLCommand.Parameters.Add("@AgtFirstNameCaps", SqlDbType.VarChar, 30).Value
```

```
= AgtFirstNameCaps
            objSQLCommand.Parameters.Add("@AgtOtherInitials", SqlDbType.VarChar, 5).Value =
AgtOtherInitials
            objSQLCommand.Parameters.Add("@AgtHonours", SqlDbType.VarChar, 30).Value =
AgtHonours
            objSQLCommand.Parameters.Add("@AgtGender", SqlDbType.VarChar, 1).Value =
AgtGender
            objSQLCommand.Parameters.Add("@AgtType", SqlDbType.SmallInt).Value = AgtType
            objSQLCommand.Parameters.Add("@AgtStatus", SqlDbType.SmallInt).Value =
AgtStatus
            objSQLCommand.Parameters.Add("@AgtInvoicePeriod", SqlDbType.VarChar, 1).Value =
AgtInvoicePeriod
            objSQLCommand.Parameters.Add("@AgtMaritalStatus", SqlDbType.SmallInt).Value =
AgtMaritalStatus
            objSQLCommand.Parameters.Add("@AgtDateOfBirth", SqlDbType.DateTime).Value =
AgtDateOfBirth
            objSQLCommand.Parameters.Add("@AgtVATCode", SqlDbType.VarChar, 20).Value =
AgtVATCode
            objSQLCommand.Parameters.Add("@AgtAddressForm", SqlDbType.VarChar, 80).Value =
AgtAddressForm
            objSQLCommand.Parameters.Add("@AgtDearForm", SqlDbType.VarChar, 50).Value =
AgtDearForm
            objSQLCommand.Parameters.Add("@AgtBodyForm", SqlDbType.VarChar, 50).Value =
AgtBodyForm
            objSQLCommand.Parameters.Add("@PatCode", SqlDbType.VarChar, 3).Value = PatCode
            objSQLCommand.Parameters.Add("@PatSubCode", SqlDbType.VarChar, 3).Value =
PatSubCode
            objSQLCommand.Parameters.Add("@PscID", SqlDbType.VarChar, 3).Value = PscID
            objSQLCommand.Parameters.Add("@AgtCreditLimit", SqlDbType.Money).Value =
AgtCreditLimit
            objSQLCommand.Parameters.Add("@AgtWarningLevel", SqlDbType.TinyInt).Value =
AgtWarningLevel
            objSQLCommand.Parameters.Add("@AgtOldPassword", SqlDbType.VarChar, 50).Value =
AgtOldPassword
            objSQLCommand.Parameters.Add("@AgtNewPassword", SqlDbType.VarChar, 50).Value =
AgtNewPassword
            objSQLCommand.Parameters.Add("@AgtDataProtectFlag", SqlDbType.Bit).Value =
AgtDataProtectFlag
            objSQLCommand.Parameters.Add("@AggID", SqlDbType.Int).Value = AggID
            objSQLCommand.Parameters.Add("@AgtJobTitle", SqlDbType.VarChar, 50).Value =
AgtJobTitle
            objSQLCommand.Parameters.Add("@AgtSignOffForm", SqlDbType.VarChar, 50).Value =
AgtSignOffForm
            objSQLCommand.Parameters.Add("@AgtOffCode", SqlDbType.SmallInt).Value =
AgtOffCode
            objSQLCommand.Parameters.Add("@SlaCreditTerms", SqlDbType.SmallInt).Value =
SlaCreditTerms
            objSQLCommand.Parameters.Add("@SlaInvoiceContact", SqlDbType.VarChar, 30).Value
= SlaInvoiceContact
            objSQLCommand.Parameters.Add("@SlaStatementContact", SqlDbType.VarChar,
30).Value = SlaStatementContact
            objSQLCommand.Parameters.Add("@WebAccountStatus", SqlDbType.TinyInt).Value =
WebAccountStatus
            objSQLCommand.Parameters.Add("@AgtCanGiftAid", SqlDbType.Bit).Value =
AgtCanGiftAid
            objSQLCommand.Parameters.Add("@AemEmailAddress", SqlDbType.VarChar, 100).Value
= AemEmailAddress
            objSQLCommand.Parameters.Add("@SlaChargeAccount", SqlDbType.TinyInt).Value =
SlaChargeAccount
            objSQLCommand.Parameters.Add("@AgtAccountLevel", SqlDbType.TinyInt).Value =
AgtAccountLevel
            objSQLCommand.Parameters.Add("@SlaInvoiceType", SqlDbType.TinyInt).Value =
SlaInvoiceType
            objSQLCommand.Parameters.Add("@SlaBookingUnpaid", SqlDbType.TinyInt).Value =
SlaBookingUnpaid
```

```
            objSQLCommand.Parameters.Add("@SlaInvoiceNonAccount", SqlDbType.TinyInt).Value
= SlaInvoiceNonAccount
            objSQLCommand.Parameters.Add("@DplID", SqlDbType.Int).Value = DplID
            objSQLCommand.Parameters.Add("@SlaOnMaturityType", SqlDbType.TinyInt).Value =
SlaOnMaturityType
            objSQLCommand.Parameters.Add("@SlaOnMaturityOffSet", SqlDbType.SmallInt).Value
= SlaOnMaturityOffSet
            objSQLCommand.Parameters.Add("@AgtBulkMailing", SqlDbType.SmallInt).Value =
AgtBulkMailing
            objSQLCommand.Parameters.Add("@SlaInvoiceLevel", SqlDbType.TinyInt).Value =
SlaInvoiceLevel
            objSQLCommand.Parameters.Add("@AgtCommVatCode", SqlDbType.VarChar, 3).Value =
AgtCommVatCode

            objSQLCommand.ExecuteNonQuery()
            AgtRefNo = Convert.ToInt32(objSQLCommand.Parameters(2).Value)
            AgtID = (objSQLCommand.Parameters(3).Value).ToString
            objSQLCommand = Nothing
        Catch ex As Exception
            Close()
            Throw New WebException("File name", "Unable to run Customer_INSERT in file
name", ex)
        Finally
            objSQLCommand = Nothing
        End Try
End Sub
```