Email from Access 2013 Web App

Author: Jim Brown, Applications Plus
www.ApplicationsPlus.com

One of the things missing from Access 2013 web app's is a way to send an email. Until the MS Team adds such a feature, here is a way you can make it happen.

Requirements

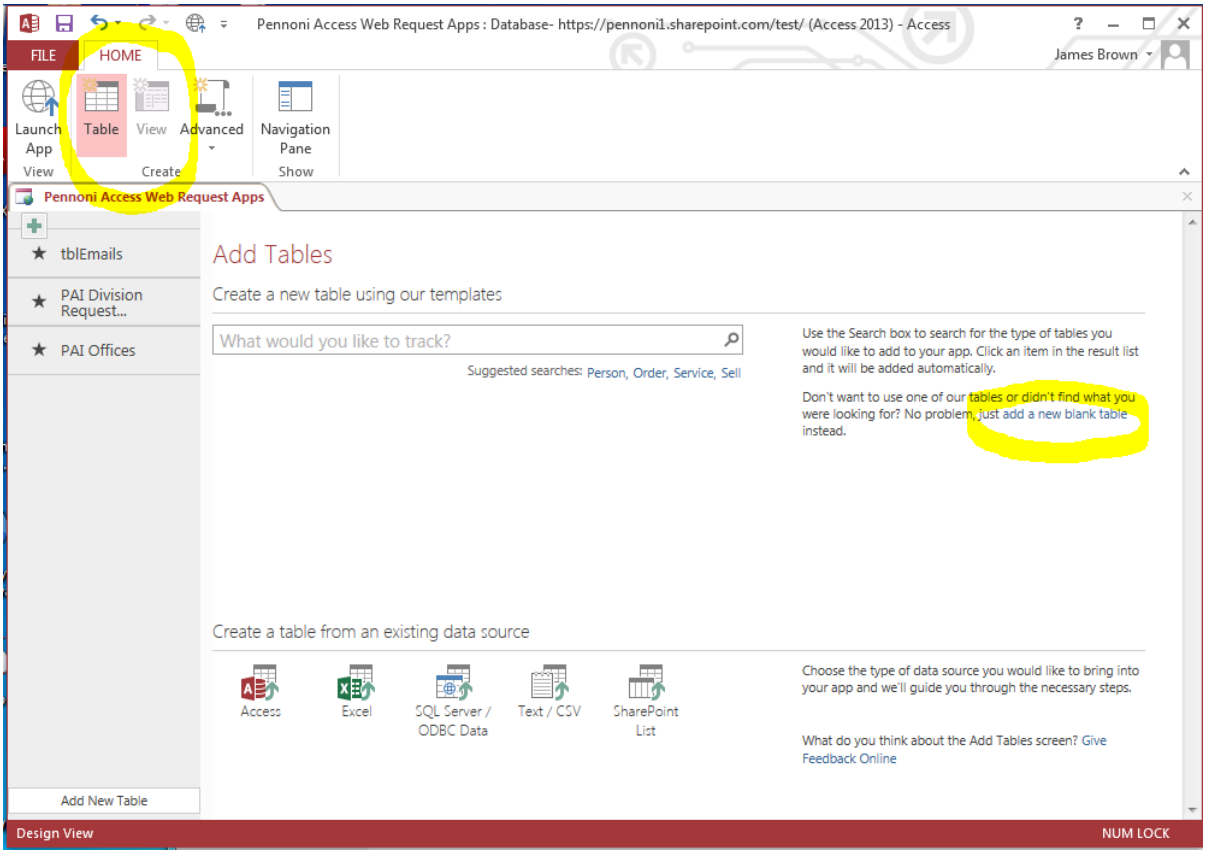
- A SharePoint in-house server or online account that is setup to host your Access web app.
- A created Access Web App that you can edit.
- A Microsoft SQL 2012 in-house server. You will need to setup the Database Mail section of the Management tree, and know the 'Profile' name that can be used. Ask your DBA or Google how to set this up.
- You will also need access to or create a SQL database to place the Stored Procedure we will write.

Overview of the email system we will build

- In the Web App we will add a table to store emails we wish to send. We will populate this table manually or with data macros as needed.
- We will setup an Access 'Connection' so that applications external of the web app can access its data.
- In the SQL server we will use the Server Objects \ Linked Server feature to connect to the Access data.
- We will write a TSQL Stored Procedure to access our Access email table, process outstanding emails, and use the standard msdb.dbo.sp_send_dbmail procedure to send our mail.
- Create a scheduled SQL Job to run the stored procedure.

Access Web App work

OK, let's get started. Open your Access web app for development in the desktop client. On the Home tab Create section click Table and then the 'just add a new blank' table link as shown below:

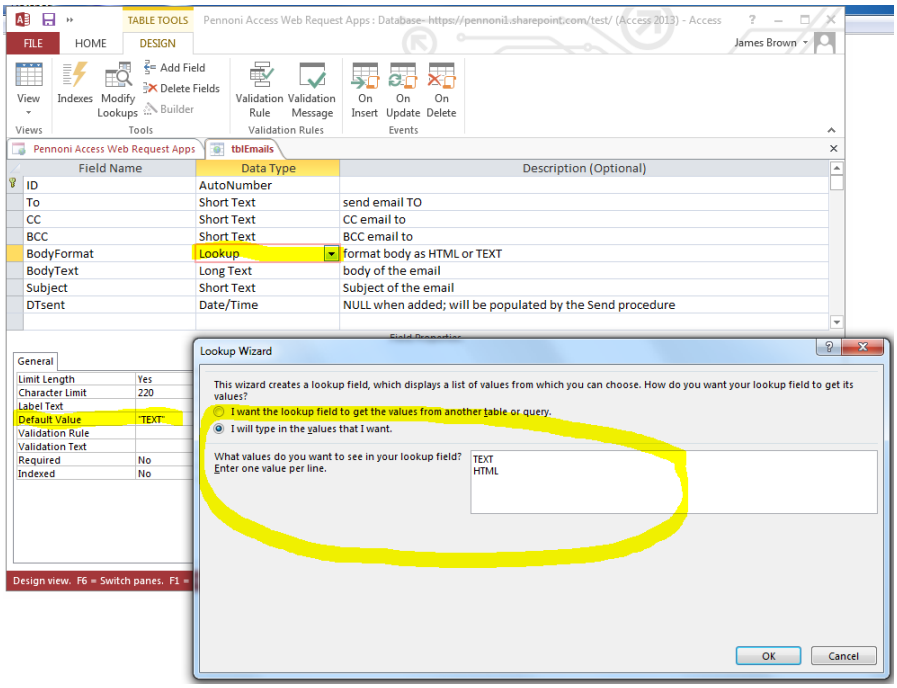


Create the following table and name it tblEmails.

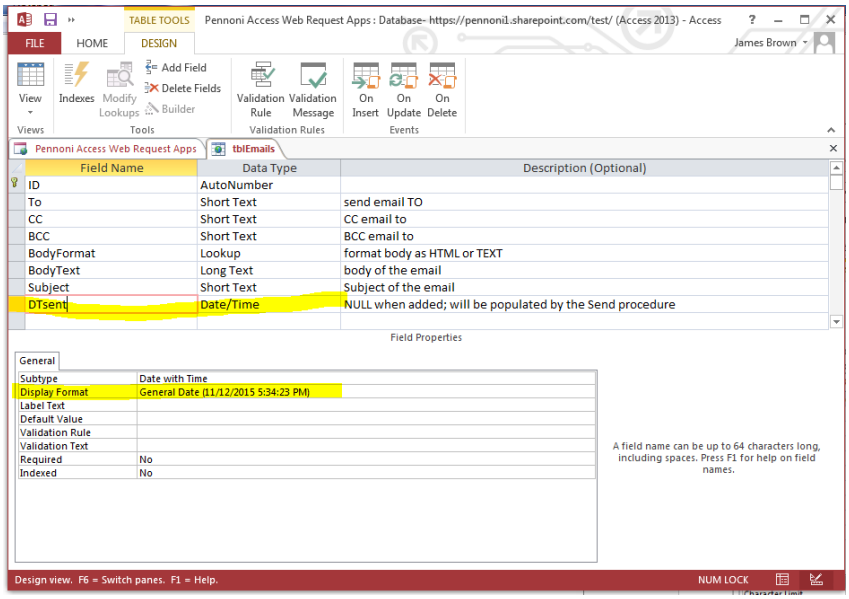
Pennoni Access Web Request Apps | tblEmails

Field Name	Data Type	Description (Optional)
ID	AutoNumber	
To	Short Text	send email TO
CC	Short Text	CC email to
BCC	Short Text	BCC email to
BodyFormat	Lookup	format body as HTML or TEXT
BodyText	Long Text	body of the email
Subject	Short Text	Subject of the email
DTsent	Date/Time	NULL when added; will be populated by the Send procedure

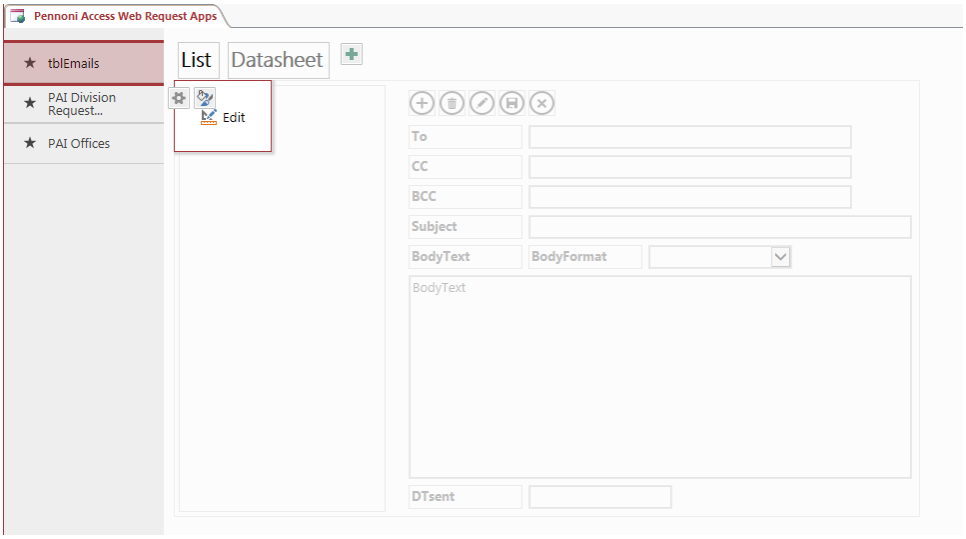
For the BodyFormat field set the **default value** to 'TEXT' and the Lookup values 'TEXT' and 'HTML' which are valid values for the sp_send_dbmail procedure.



For the DTsent field, set the subtype to **Date with Time**.

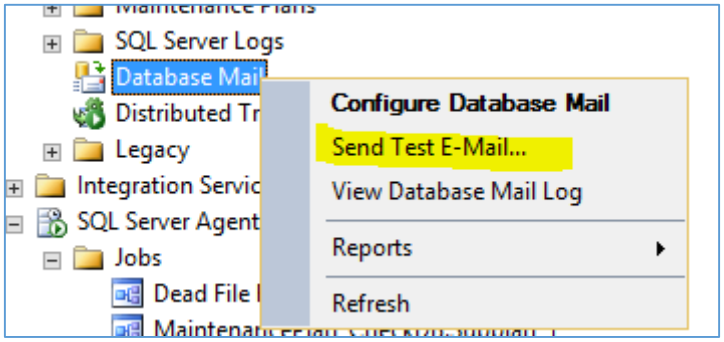


Access will create a List and Datasheet for you (Note: I've edited the List form below to make it more usable). Later you can populate this table using a data macro.

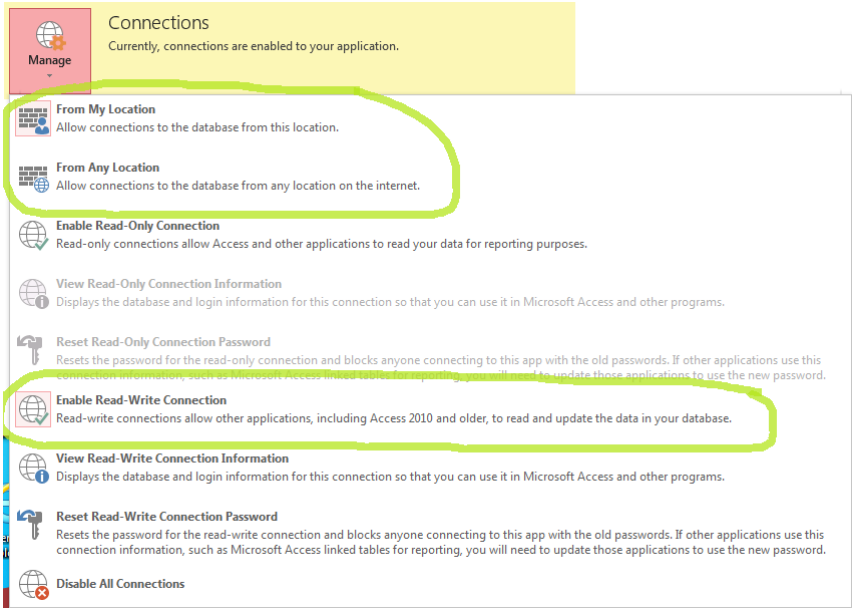


At this point, launch your application and add a few test records. In these records set at least the To, Subject, BodyFormat, and BodyText fields.

Be aware that your SQL servers Database mail setup may restrict the email addresses you can send mail to. Check with your DBA or IT administrator. You can also use Database Mail's **Send Test E-mail** dialog to try a address.



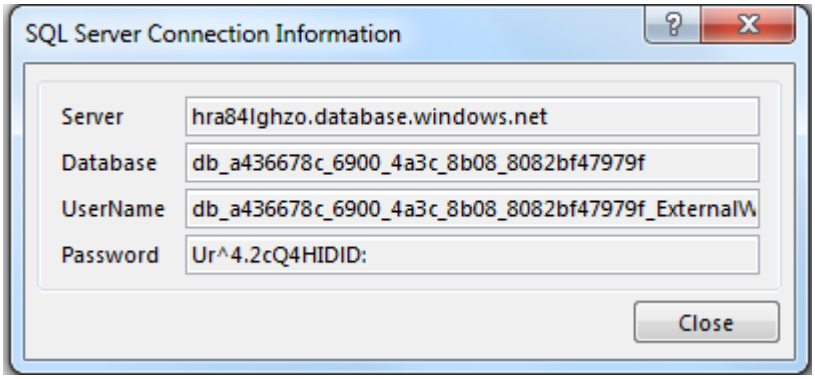
Now go back to the desktop client. We will enable Connections to the database. Click on the File tab to get to the Info page. Then click on the Connections button.



On the Connections dialog click either **From My Location** or **From Any Location**. Then click the **Enable Read-Write Connection**. The icons will highlight to show your selection.

Now click the **View Read-Write Connection** link. The dialog shown below will appear. You will want to Copy each of the four values and Paste them to a NotePad or other text document. As you can see you don't want to be trying to type in this info.

You will use this in the next section to link to the Access data.

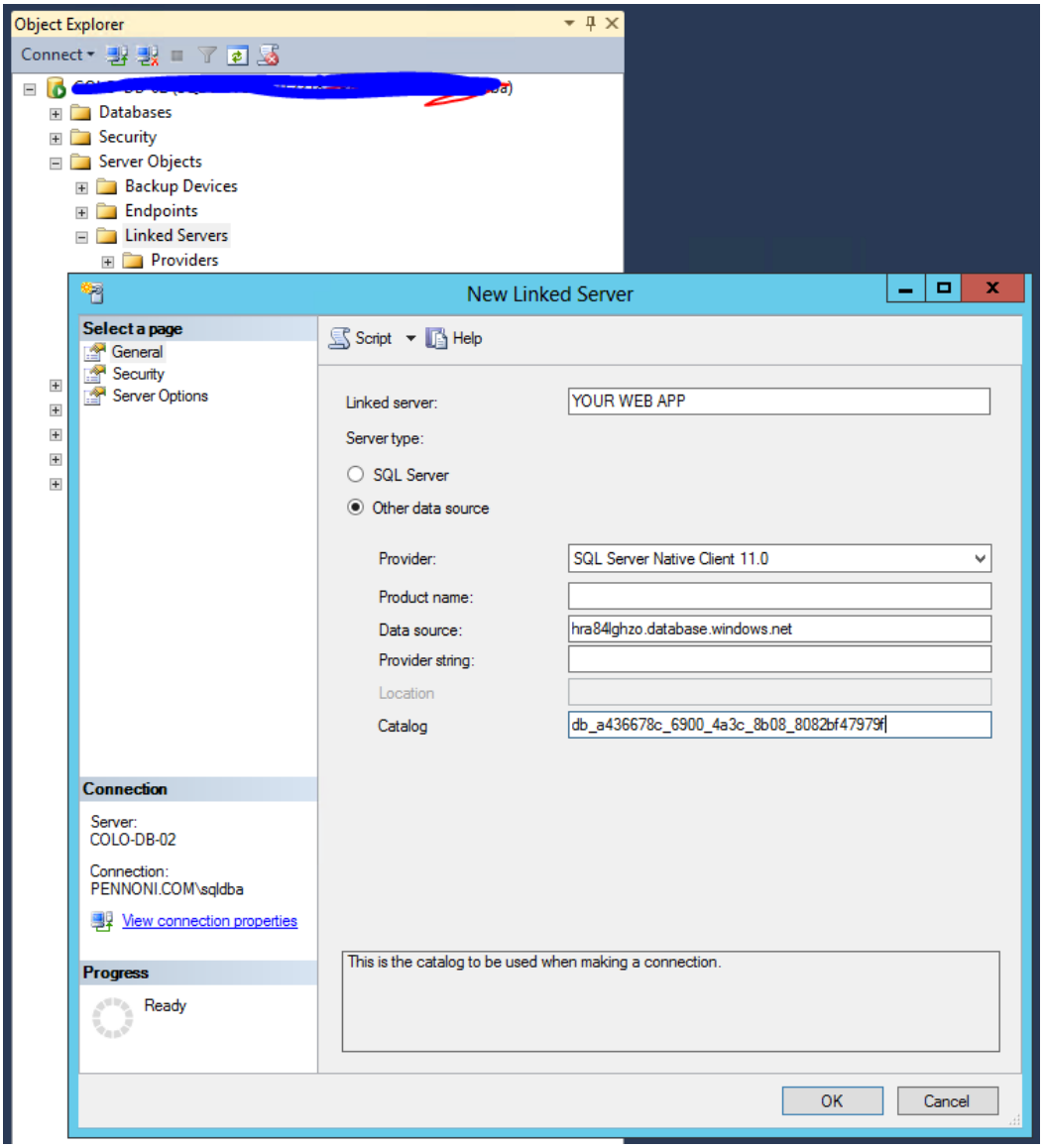


SQL 2012 Server link to the Access data

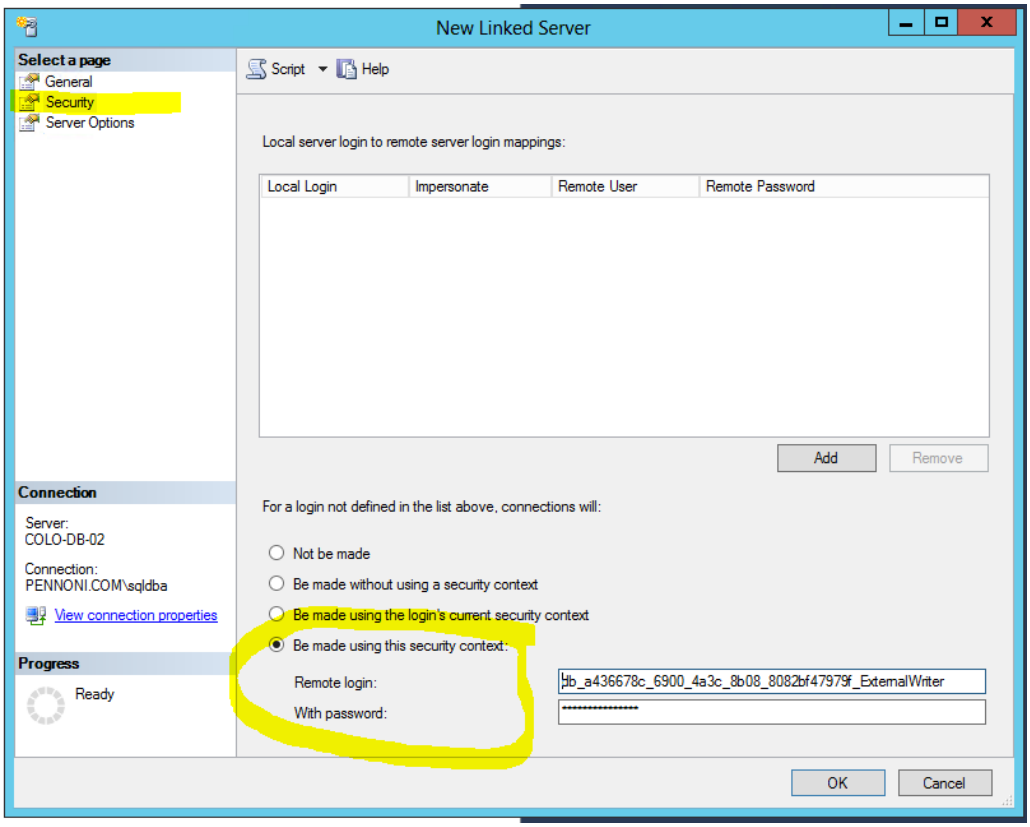
Now we will link your in-house SQL Server to the Access Web App's data. Open your server in SSMS. Select Server Objects \ Linked Servers. Right click and create a new linked server. Set these values:

On the General page	
Linked Server	create a name that describes your web app
Server Type	Check off **Other data source** then set the following
Provider	SQL Server Native Client 11.X. This is the SQL 2012 version provider
Data Source	Set this to the Server name you got from Access
Catalog	Set this to the Database name you got from Access
On the Security page	
Check off **Be made using this security context** then set the following	
Remote Logon	Set this to the Username you got from Access
With Password	Set this to the Password you got from Access

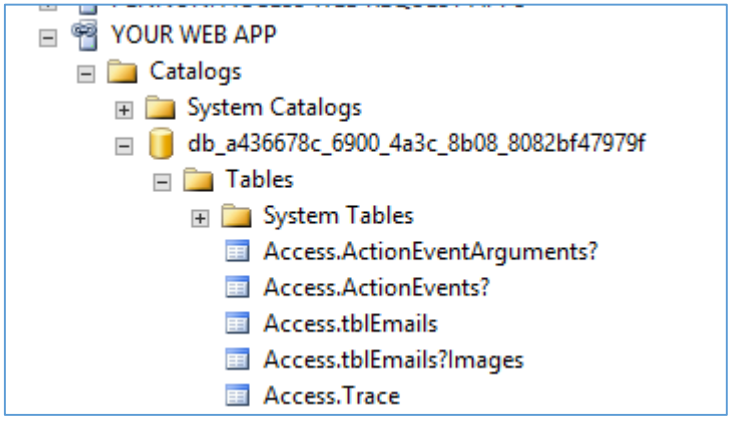
General page



Security page



If you have successfully linked to your Access data you should be able to expand the Catalogs \ database \ and tables to see your Access table names.



TSQL stored procedure to process tblEmail

Now we will write a stored procedure that will process tblEmail for outstanding emails. In SSMS select the database you want to create the procedure in. With it selected click the New Query button and enter or copy/paste the following. This is also available as a TSQL script file on my web site.

Note the sections with ***Edit this***.

```
--**** start of copy/paste ****

CREATE PROCEDURE [dbo].[spSendAccessWebAppEmail]
AS
BEGIN

DECLARE @To as varchar(200), @CC as varchar(200), @BCC as varchar(200), @BodyFormat as varchar(4), @DTsent as datetime
DECLARE @BodyText as varchar(8000), @Subject as varchar(200)

DECLARE Emails CURSOR FOR
        SELECT [To],[CC],[BCC],[BodyFormat],[BodyText],[Subject],[DTsent]

        --***Edit this***
        -- This needs to be set to [Web app server link].[database name].[Access].[tblEmails]
        -- [Web app server link] is as you have set your Linked Server to the Access data
        -- [database name] is assigned in the Access Web app's Connection properties
        FROM [YOUR WEB APP].[db_a436678c_6900_4a3c_8b08_8082bf47979f].[Access].[tblEmails]
        WHERE [DTsent] IS NULL

OPEN Emails
FETCH NEXT FROM Emails INTO
        @To,@CC,@BCC,@BodyFormat,@BodyText,@Subject,@DTsent

-- loop thru the requests and generate the emails
WHILE @@FETCH_STATUS = 0
BEGIN
        --print @To
        --print @bodytext
        --print ''

        SET @Subject = ISNULL(@Subject,'Message from Access Web App')
        SET @BodyFormat = ISNULL(@BodyFormat,'TEXT')

        EXEC msdb.dbo.sp_send_dbmail
--***Edit this***  set your yours servers Database Mail 'profile'
    @profile_name = 'BAS',
    @recipients = @To,
    @copy_recipients = @CC,
    @blind_copy_recipients = @BCC,
    @subject = @subject,
    @body_format = @BodyFormat,
    @body=@bodyText;


FETCH NEXT FROM Emails INTO
        @To,@CC,@BCC,@BodyFormat,@BodyText,@Subject,@DTsent
END
CLOSE Emails
DEALLOCATE Emails

--***Edit this***
-- This needs to be set to [Web app server link].[database name].[Access].[tblEmails]
-- [Web app server link] is as you have set your Linked Server to the Access data
-- [database name] is assigned in the Access Web app's Connection properties
UPDATE [YOUR WEB APP].[db_a436678c_6900_4a3c_8b08_8082bf47979f].[Access].[tblEmails]
SET [DTsent] = GETDATE()
WHERE [DTsent] IS NULL

END
--**** end of copy/paste ****
```

Now to test your procedure, use the following TSQL script.

```
DECLARE @RC int
```

```
EXECUTE @RC = [dbo].[spSendAccessWebAppEmail]
```

Once tested you will want to use this same script in a scheduled SQL Job that wakes up every XX min's and tests for outstanding emails to be sent.

If you want to pass HTML markup in your Body Text, try pasting in something like the following. Be sure to set the BodyFormat field in tblEmails to 'HTML'.

```
<table border="1" style="width:500px">
<col style="width:100px">
<col style="width:400px">
<tr><td>Description</td><td>Value</td></tr>
<tr><td>Office:</td><td>my office</td></tr>
<tr><td>Division:</td><td>my division</td></tr>
</table>
```