Paper 086-2010

Data in Your Inbox? Use it!

Douglas Liming, SAS Institute Inc., Cary, NC

ABSTRACT

Using the SAS/ACCESS® Interface to ODBC, you can pull valuable information from your Microsoft Outlook Inbox. A wealth of information can be mined from your e-mail, so why not use it? Based on e-mail, you can answer questions such as: What projects are generating traffic for you? Are certain people and groups increasingly asking for more information? How quickly are responses being handled? What are some frequently asked questions and are there any you had not realized? Find out how to first use SAS/ACCESS® to pull data from your Inbox, and then use some simple SAS® coding to pull the information together to see what gems you find.

INTRODUCTION

E-mail consumes a large portion of our time and attention, and it has unarguably become an integral part of our lives. The information within our Inboxes ranges from the mundane to the critical. What other data is there? What if you had an easy way to pull information from your Inbox into SAS 9,2? Then you could easily leverage the power of SAS for all levels of reporting and discovery.

With the SAS/ACCESS® 9.2 Interface to ODBC, Microsoft Access 2007, and a Microsoft Outlook 2007 Inbox, we will configure a Windows XP workstation to create a SAS data set containing records of information pertaining to the Sent Items and the Deleted Items from Microsoft Outlook. To begin the journey, we will configure a linked table within Microsoft Access, and then configure an ODBC connection to that linked table. After that we will connect with SAS, create the SAS data sets, and do a few sample summaries.

STEP 1: CONFIGURE MICROSOFT ACCESS

Microsoft Access 2007 has the ability to create a link to a data source by creating a linked table. A linked table is a table that will maintain a link to the data source. This will enable us to have real-time data pulls from an Inbox, without having to import the data every time we want to use SAS 9.2 to data mine the Inbox. Also, this will allow us to perform the configuration phase only one time and to solely use SAS in the future.

To begin, we will open Microsoft Access and start with a new database, which we will call `exchange.mdb`. Next, we will click the **External Data** tab and select the **More** drop-down list.

Figure 1. External Data Tab and More Drop-down List.

On the **More** drop-down list, select **Outlook Folders**. This will display the Get External Data screen (Figure 2). Three choices are presented with radio buttons. Select **Link to data source by creating a linked table**, as selected in Figure 2.

The next screen (Figure 3) will display the Link Exchange/Outlook Wizard. It displays the current Outlook Exchange folder setup. The first time through, we are going to select **Sent Items**. Later, to create a link to **Deleted Items**, we will simply repeat this process.

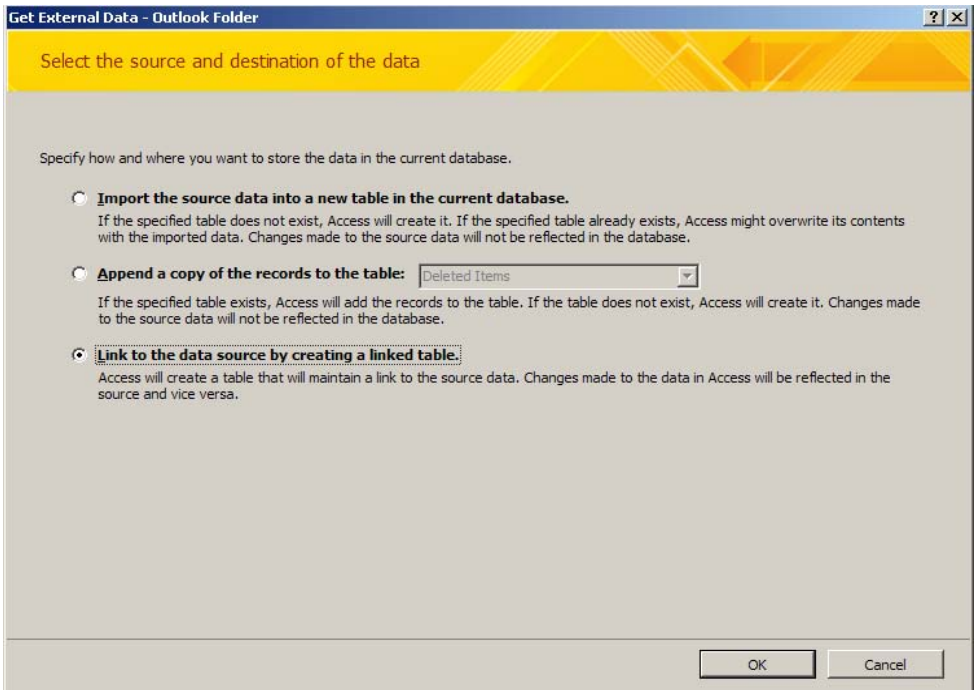

Figure 2: Get External Data

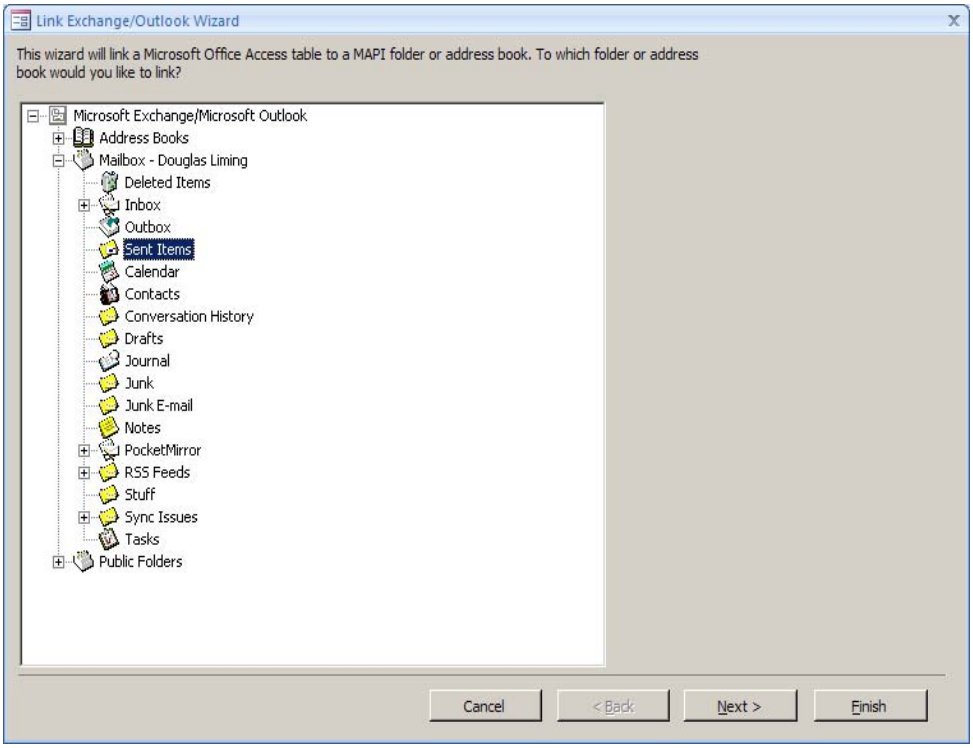

Figure 3: Link Exchange/Outlook Wizard

Click **Next**, and then you will be prompted to name the table (Figure 4). For some of the examples below, we will use the default name of **Sent Items**. This completes defining the linked table. You can test this by double-clicking the table name, **Sent Items**, in the column to the left. In addition, for some of the examples below we will be using the **Deleted Items**. To add that data source, we can simply repeat the process starting with Figure 1, with the **More** drop-down list, creating another linked table within the current database.

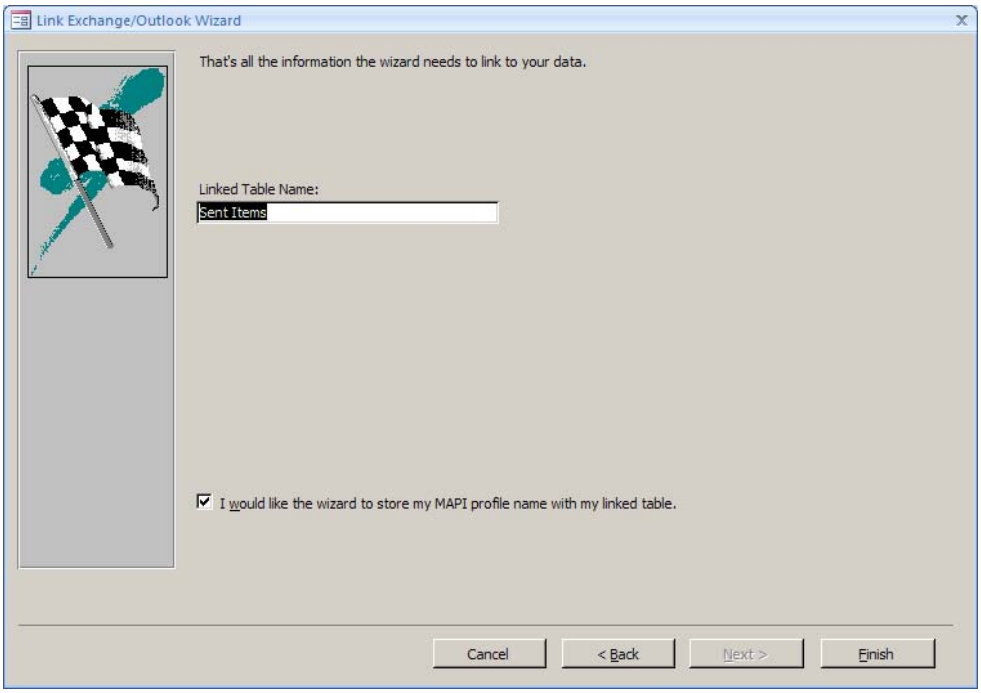

Figure 4: Name the Linked Table

Now that we have completed the Microsoft Access portion, we can move on to defining the ODBC connection.

STEP 2: CONFIGURE A SYSTEM ODBC CONNECTION

SAS/ACCESS will use ODBC to talk to the Microsoft Access database that we just created above. This step begins by defining our ODBC data source using the ODBC Data Source Administrator. It is invoked either by selecting **Start → Run** and then typing **odbcad32** or by selecting **Start → Control Panel → Administrative Tools → Data Sources (ODBC)**.

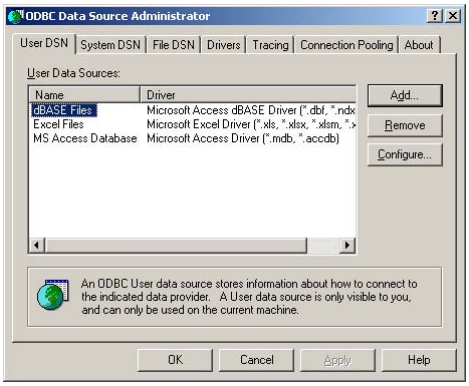

Figure 5: ODBC Administrator

Either method will start the ODBC Data Source Administrator (Figure 5). Continue by clicking the **Add** button. Next, select **Microsoft Access Driver (*.mbd, *.accdb)** (Figure 6). This will take you to the ODBC Microsoft Access Setup screen (Figure 7).

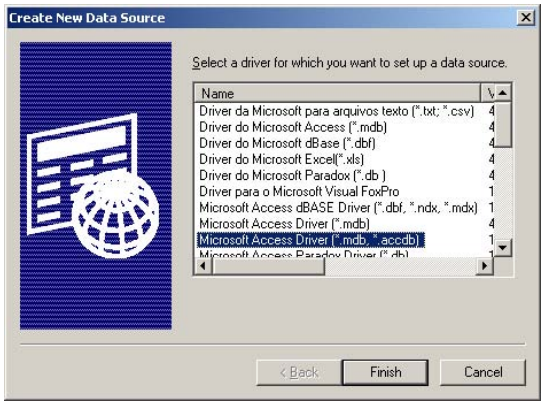

Figure 6: Create New Data Source

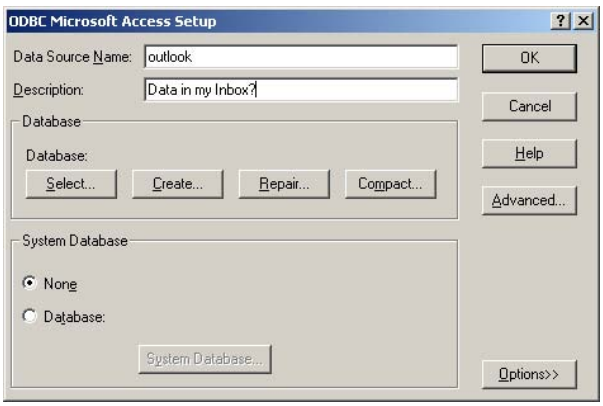

Figure 7: ODBC Microsoft Access Setup

Before clicking **OK**, be sure to click **Select** in the **Database** box, and browse to the Microsoft Access database location and file (`exchange.mdb`) that was created in Step 1.

This basically completes the configuration phase. These steps will not have to be repeated. Now we can move on to SAS.

STEP 3: PULL THE EXCHANGE/OUTLOOK DATA INTO A SAS DATA SET

Fire up SAS 9.2 on the same Windows XP workstation that the configurations work was done on, and submit a simple LIBNAME statement:

```
/* Looking at the month of January */
libname myJan odbc dsn=exchange;
```

If you set up your DSN connection following the steps above, you will now have two SAS data sets called Sent Items and Delete Items. If you look via the SAS Explorer, you will notice several other data sets, which are Microsoft Access internal tables. Let us look at what comes across from Microsoft Outlook:

```
proc contents data=myJan.'Sent Items'n; run;
```

Alphabetic List of Variables and Attributes

#	Variable	Type	Len	Format	Informat	Label
8	CC	Char	255	$255.	$255.	CC
19	Content Unread	Num	8	11.	11.	Content Unread
12	Contents	Char	1024	$1024.	$1024.	Contents
13	Created	Num	8	DATETIME20.	DATETIME20.	Created
16	Has Attachments	Num	8	1.	1.	Has Attachments
2	Icon	Char	255	$255.	$255.	Icon
1	Importance	Num	8	11.	11.	Importance
6	Message CC to Me	Num	8	1.	1.	Message CC to Me
11	Message Size	Num	8	11.	11.	Message Size
5	Message To Me	Num	8	1.	1.	Message To Me
14	Modified	Num	8	DATETIME20.	DATETIME20.	Modified
17	Normalized Subject	Char	255	$255.	$255.	Normalized Subject
18	Object Type	Num	8	11.	11.	Object Type
3	Priority	Num	8	11.	11.	Priority
7	Sender Name	Char	255	$255.	$255.	Sender Name
10	Sent	Num	8	DATETIME20.	DATETIME20.	Sent
4	Subject	Char	255	$255.	$255.	Subject
15	Subject Prefix	Char	255	$255.	$255.	Subject Prefix
9	To	Char	255	$255.	$255.	To

Figure 7: Output from PROC CONTENTS of Sent Items

Now that you have your Inbox in a SAS data set, you can leverage the full potential of SAS to data mine and explore the information and data that you receive every day. Though the SAS data sets are the primary objective of this paper, here are a couple of simple uses of the data sets that I frequently use.

STEP 4: SIMPLE SAMPLE SUMMARIES

Just to get a feel for the data, the following are a couple quick coding examples. First, move the SAS data sets into a WORK library, and keep only the columns that we will be working with.

```
data work.SentItems (keep=To Sent Subject cc contents 'Message Size'n 'Normalized
Subject'n datesent);
    set myJan.'Sent Items'n;
    where datepart(sent)  >= '01JAN10'd and datepart(sent) <= '30JAN10'd;
    datesent = datepart(sent);
    run;


data work.DeletedItems (keep=Subject From 'Sender Name'n cc To Received contents
'Normalized Subject'n );
    set myJan.'Deleted Items'n;
    where datepart(Received)  >= '01JAN10'd and datepart(Received) <='30JAN10'd;
    run;
```

HOW MANY E-MAILS AM I SENDING PER DAY?

```
/* How many e-mails a day am I sending? */
proc sql;
create table work.TotalsEmailsDay as
    select distinct datepart(Sent) format=mmddyy8. as SentDate,
    weekday(datepart(Sent)) format=downame. as DayWeek, count(datepart(Sent)) as
    TotalEmails from work.SentItems group by SentDate;
select * from work.TotalsEmailsDay;
quit;
```

TOP USERS

```
/* Who are my top users? */

/* First let us build a data set with all contacts and counts. */
proc sql;
    create table work.TopUsers as
    select distinct 'Sender Name'n as FullName, count('Sender Name'n) as
    TotalContacts from work.DeletedItems
           group by FullName
           order by TotalContacts descending;
    quit;

/* Load up a list of friends to filter out from the real "work". */
/* In email.addresses, just edit a flat file and list the e-mail
    addresses of friends and family. */
data work.friends;
infile 'C:\sgf2010\email.addresses' length=linelen;
input emails $varying500. linelen;
run;
```

```
/* Now print just the top 10. */
proc sql;
    create table work.workTopUsers as
    select FullName, TotalContacts from work.TopUsers a
    where not exists (select 1 from work.friends b where a.FullName = b.emails);
    quit;

proc sql;
    select FullName as 'Top Senders'n, TotalContacts as 'Number of Emails
    Received'n from work.workTopUsers
    (obs=10);
    quit;
```

NUMBER OF HITS FOR CERTAIN PROJECTS/PRODUCTS

```
/* What project/products are being mentioned the most in my e-mails?
    Being a DBA, I like to see which of the databases are getting hits. */

proc sql;
    select count(*) as 'Database Hits'n, 'Oracle' from DeletedItems where
    UPCASE(contents) like '%ORACLE%'
    union all
    select count(*), 'DB2' from DeletedItems where UPCASE(contents) like '%DB2%'
    union all
    select count(*), 'Teradata' from DeletedItems where UPCASE(contents) like
    '%TERADATA%'
    union all
    select count(*), 'ODBC' from DeletedItems where UPCASE(contents) like '%ODBC%'
    union all
    select count(*), 'MySQL' from DeletedItems where UPCASE(contents) like
    '%MYSQL%'
    union all
    select count(*), 'AsterData' from DeletedItems where UPCASE(contents) like
    '%ASTER%'
    union all
    select count(*), 'Netezza' from DeletedItems where UPCASE(contents) like
    '%NETEZZA%'
    union all
    select count(*), 'Neoview' from DeletedItems where UPCASE(contents) like
    '%NEOVIEW%'
    union all
    select count(*), 'Greenplum' from DeletedItems where UPCASE(contents) like
    '%GREENPLUM%';
quit;
```

CONCLUSION

This paper provides an interesting way to pull e-mail Inbox data into SAS 9.2 using the SAS/ACCESS 9.2 Interface to ODBC.

REFERENCES

Chapter 22, “SAS/ACCESS Interface to ODBC,” in *SAS/ACCESS 9.2 for Relational Databases: Reference*, Second Edition

ACKNOWLEDGMENTS

I would like to thank Brennan Liming for her unwavering support. Also, I would like to thank members of the SAS/ACCESS team: Howard Plemmons, Keith Handlon, Phil Mohr, and Mauro Cazzari.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Douglas B. Liming
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27513
douglas.liming@sas.com
http://www.sas.com