Prepare the application (Foundation)

The Foundation is an MS Access database used as the basis for all new MS Access applications. It is also to be retro-fitted to pre-existing applications built with earlier versions of the Foundation. Retro-fitting is done as those applications are enhanced or repaired. The latest version of the Foundation database can be found in the application directory (**APP ID #123).**

In this chapter there are two sets of instructions for preparing an application:

A. **NEW APPLICATION**: Instructions for a **new** application are provided entirely in the short section "For a new application" below. You can skip the rest of the chapter after that.

B. **PRE-EXISTING APPLICATION**: The other set of instructions is for modifying a **pre-existing** MS Access application, for which most of this chapter is dedicated.

For a new application

If you are building a new application from scratch, you must use as your starting point an MS Access database called "the Foundation" (**APP ID #123**). It contains the sole permissible means of doing various things in your new application and will save you a lot of development time. To ensure that you are using the latest version of the Foundation for all your projects, always download the latest version from application directory.

Some of the "means of doing various things" and their corresponding Foundation objects are:

- Navigating between features (form **Switchboard**).
- Selecting a file for import (form **frmFileSelect**).
- Creating automated screen scrapes of OMICRON (module **modOMICRON**).
- Logging user activity (module **modLogActivity**).
- Measuring, gathering, and delivering metrics (all the **"…metrics**" modules, tables, and forms).

By using the Foundation and its components in your new builds, you will be in compliance and will be saved a lot of research and coding time. You also will not need to do the piecemeal updates --- and so can skip the remainder of this chapter.

For a pre-existing application

If you are modifying a pre-existing application, there is a strong likelihood it was built with an earlier version of the Foundation. Later versions of the Foundation addressed bugs, performance issues, best practices, and Operational Metrics. Your first step in working on a pre-existing application is therefore to selectively implement the components of the latest version of the Foundation. This requires you to be extremely careful as you do so because of possible (or probable) custom code or data in the objects to be replaced or updated.

The rest of this chapter will help you through that re-Foundationing process.

Objects to delete in your app

TABLE: tblZeta
FORM: frmZeta

Objects to import from Foundation

Note that if you modifying a pre-existing application, you need to examine the original object for any custom code or, if a table, custom data: if present, you must copy that custom code/data to your new, imported Foundation object or, alternately, to its own module (if code).

Shown below is a summary checklist of the tasks in this chapter. It does not replace this chapter, but should be used as a checklist once you become proficient with the tasks in this chapter. **Read this entire chapter** before attempting to act on this table.

Object Type	**Object**	**Replace if present**	**Custom code/data present** (if so move or copy into new object)	**Comments**
TABLE	tblMetricLogTime	Do NOT	None	This table only contains data temporarily, so any records you find in your original version can be ignored.
	tblReference	Do NOT	**YES**	Add new text field **MetricReviewUserID**.
	tblMetricLogTimeHistory	Do NOT	**YES**	If table structure the same as latest, keep current version.
	tblMetricReview	Do NOT	**YES**	If table structure the same as latest, keep current version rather than copying all the data to the new version.
	tblVersionInfo	Replace	**YES;** copy into new version.	Copy info from old version's fields **CurrVer** and **CurrLoc**.
	Switchboard Items	Replace	**YES;** copy into new version.	Copy all info from old table to new. In new table, mark all buttons as Visible = True.

QUERY	qpasDateLookup	Replace	None	
FORM	frmMetricRevDetails_sub	Replace	None	
	frmMetricSentLog_sub	Replace	None	
	frmMetricReview	Replace	POSSIBLE; move any into module modMetricCalculations.	Developers were not to edit frmMetricReview, but sometimes did.
	frmMetricsAlertPopUp	Replace	None	
	Switchboard	Replace	POSSIBLE; move any into new version of form.	Custom code might include calls to functions **StartTimer** and **EndTimer**.
	frmUtility	Replace	POSSIBLE; move any into new version of form.	
MODULE	modMetricsConfig	Replace	None	
	modStartUp	Replace	None	
	modMetricCalculations	Do NOT	**YES**	If present, you will need to manually edit the code in modMetricCalculations.
	modConnect (replace if already present)	Replace	None	
	modShutDown (replace if already present)	Replace	None	
	modDateCalculations (replace if already present)	Replace	None	
	modConstants	Do NOT	**YES**	You only need to delete a few things and add a few more.
	modMaintenance	Replace	None	
	modQAChecks	Do NOT	**YES**	Copy Foundation's new code (differences) to current version.

After you bring in the components (as needed) in the table above, there are some other preparatory steps to do. **Read the rest of this chapter** before you act on the table above.

Objects to import with no revisions needed

Remove pre-existing versions if they exist.

TABLE	**tblMetricLogTime**
QUERY (pass-through)	**qpasDate_Lookup**
FORM	**frmMetricsAlertPopUp**
	frmMetricRevDetails_sub
	frmMetricSentLog_sub
	frmCloseNow
	frmProcess
MODULE	**modStartUp**
	modMetricsConfig
	modConnections
	modShutDown
	modDateCalculations
	modMaintenance

Import & prepare table tblVersionInfo

Import this table, then copy the following information from the old version of this table to the new (and record today's date):

- **CurrVer** (current version of the *app*, not the Foundation).
- **CurrLoc** (the current app's location).

Leave the fields **DateUpdate** and **CurrFoundationVersion** as they were when you imported the table. **DateUpdate** will be updated automatically by the application database when in production. **CurrFoundationVersion** is *always* what was in the table you imported from the Foundation.

Import & prepare form Switchboard & table Switchboard Items

After importing this form and table:

1. The form: In design mode, make the Switchboard's clock graphic **imgMetrics** be visible. If you will be implementing automatic delivery of metrics rather than making user use the Metric Review form (frmMetricReview), make the "Auto-Delivery" graphic **imgMetricsAutoDelivery** be visible, also. If in doubt, make "Auto-Delivery" be visible.

2. The form: In the app's original (old) **Switchboard form module**, look for custom code specific to this application. It is usually identifiable by the original developer's recording the project number, their name, and an explanation for the customization. Copy any customized code to the new version of the form Switchboard.

3. Delete or deprecate the original **form Switchboard**.

4. In the new version of table **Switchboard Items**, delete all records, then append the records from the original version of this table into the new version. When you have done that, delete or deprecate the original table **Switchboard Items**.

5. In the new version of table Switchboard Items, add a new entry for the Staging form:

   SwitchboardID: **1**
   ItemNumber: [just before EXIT]
   Item Text: "**Deliver Metrics**"
   Command: **3**
   Argument: "**frmMetricReview**"
   Visible: **True**
   Business Description: "**Open form to gather metrics/deliver to Metrics Central Database.**"
   Technical Description: [same as Business Description]

6. Make all items in the new version of table Switchboard Items be visible (Visible=True).

NOTE: In the Switchboard's module is a call to the procedure **SwitchboardTimer**. You may get a compile error that this is missing. That is OK for now. We will add the missing function when we get to preparing the module modMetricCalculations.

Edit standard modules

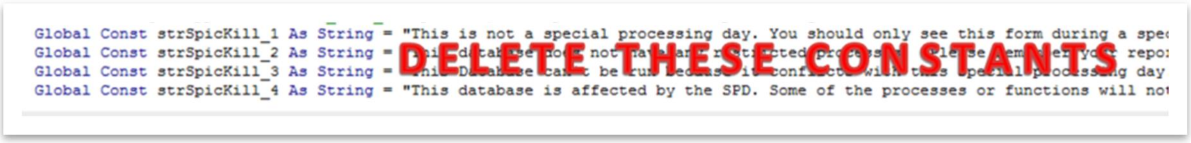
A. Module modQAChecks

If this module does not exist in your application, simply import it. If this module already exists in your app, it is likely to have custom code. If so, compare the current module with the Foundation's and update the current version accordingly.

B. Module modConstants

Never replace the existing version of this module as it will always contain customized code. We will simply update it selectively.

1. Delete all four global constants that begin with "**Global Const strSpicKill**…" See below.

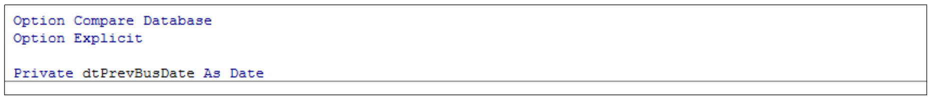


```
Global Const strSpicKill_1 As String = "This is not a special processing day. You should only see this form during a spe
Global Const strSpicKill_2 As String = "This database does not have any restricted processes. Please remember your repo
Global Const strSpicKill_3 As String = "This Database can't be run because it conflicts with this special processing day
Global Const strSpicKill_4 As String = "This database is affected by the SPD. Some of the processes or functions will no
```

C. Module modMetricCalculations

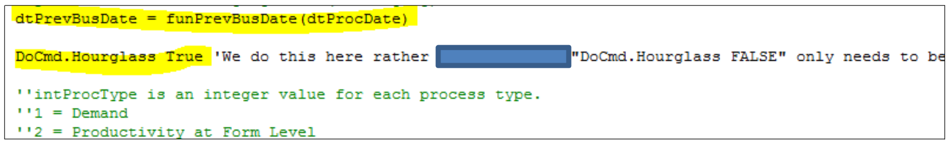
If your pre-existing application did not have this module, simply import it. Otherwise, you need to update the current version of modMetricCalculations as follows:

1. At the top of the module, add the module-level variable, **dtPrevBusDate**.



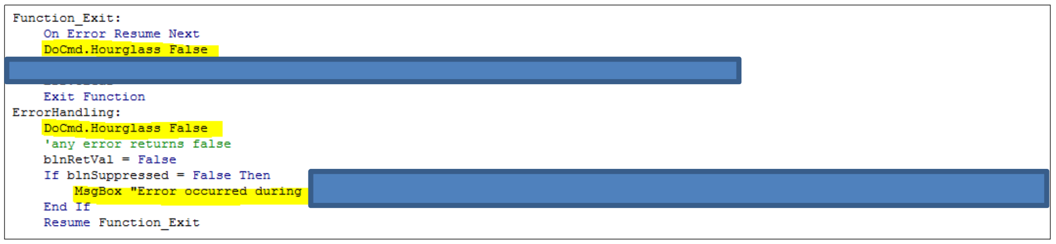
```
Option Compare Database
Option Explicit

Private dtPrevBusDate As Date
```

2. In function GatherMetrics, add the highlighted code to the top of the function…



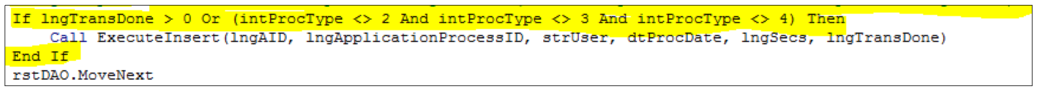
```
dtPrevBusDate = funPrevBusDate(dtProcDate)

DoCmd.Hourglass True 'We do this here rather [redacted]"DoCmd.Hourglass FALSE" only needs to be

''intProcType is an integer value for each process type.
''1 = Demand
''2 = Productivity at Form Level
```

…and this highlighted code to the end of the function.



```
Function_Exit:
    On Error Resume Next
    DoCmd.Hourglass False
[redacted]
    Exit Function
ErrorHandling:
    DoCmd.Hourglass False
    'any error returns false
    blnRetVal = False
    If blnSuppressed = False Then
        MsgBox "Error occurred during [redacted]
    End If
    Resume Function_Exit
```

3. In function ExecuteGatherMetrics, add the following If…Then statement around the “Call ExecuteInsert.”



```
If lngTransDone > 0 Or (intProcType <> 2 And intProcType <> 3 And intProcType <> 4) Then
    Call ExecuteInsert(lngAID, lngApplicationProcessID, strUser, dtProcDate, lngSecs, lngTransDone)
End If
rstDAO.MoveNext
```

4. Add the following procedures/functions in their entirety:
    - StartTimer
    - AutomaticMetricsDelivery
    - funDatePreviousTrans

5. To every error handler in modMetricCalculations, add “DoCmd.Hourglass False” as the first line to execute if there is an error.

Prepare table tblReference (hidden table)

1. Unhide and then add a new field to the hidden table tblReference:

    NAME: **MetricReviewerID**
    TYPE: **Text (255)**

    Leave all other fields and default properties as they are.

2. Hide the table again when you are done.

See Appendix B for instructions on how to unhide and hide a table.

Prepare tables tblMetricLogTimeHistory & tblMetricReview

If these tables are not present in your app, simply import them. If these tables are already present in your app, be aware that the information stored in tblMetricLogTimeHistory and tblMetricReview must be maintained. Compare the structure of your current versions to the Foundation’s: if they differ, simply edit the structure of your app’s current version.

Forms frmMetricReview

If missing, simply import this form. You may replace it if already present, but ensure that any custom code and custom *formatting of the form itself* is addressed:

- The most common customizations of these form's design are the disabling of the date-selection combo boxes (forcing use of today's date) and the customization of comments on the "About Metrics" tab. You should incorporate these customizations into the new version that you import.
- The most common custom VBA in **frmMetricReview**'s module is the calling of procedures that generate or update a table of metrics. Any custom VBA will usually be prefaced with a comment giving the date, developer's name, and a project number. If you encounter custom code in frmMetricReview, you should move it to modMetricCalculations if possible (which will probably require you to re-write the custom code).

ENTIRE APP: Date calculations

One of the frequent calculations performed for both business processes and operational metrics is determining **business day**. In the past, there were various ways these calculations were supported, both by formulas and by centralized data tables in SQL Server. The most common method in the past was to link to a table called "tblNonBusDays." This and other methods are **no longer supported and must be replaced**.

The methods to use for Business Day (and all other date calculations) are found in:

- Module **modDateCalculations**
- Query (pass-through) **qpasDate_Lookup**

Therefore, search your app's pre-existing code for all occurrences of dates in the code ("date", "dt*", "*holiday*", "*PrevBus*","Biz", etc.) and replace date calculations as necessary with those in module **modDateCalculations** as applicable.

ENTIRE APP: Other edits to make in pre-existing applications

The latest version of the Foundation's VB Project has some important revisions that may not carry over if you selectively copy-and-paste changes from the Foundation into the app. You will need to review the pre-existing code and **make the following changes manually**:

- "**Option Explicit**" at the top of every module: form, standard, or class.
- **Error handling** with name of function and module included in error message.
- **Naming convention** compliance for variables (see chapter "Naming conventions for custom objects and variables", below).