Overview of Projects Within the Visual Studio Solution

- **CmdpErrorReviewer**
  - Front end portal installed on staff computers
  - Used to review rejections
  - Rejections from both validation steps will appear in the list
- **MigrateValidateDseData**
  - Stand-alone executable used to parse data from XML files that are output by DSE
  - Loads data into our database tables
  - Runs our validation on parsed data
- **CompileForXmlSampling**
  - Stand-alone executable used to create a new XML to be submitted to XML Sampling
  - Compiles data that has passed our validation
- **CheckXmlSamplingRejections**
  - Stand-alone executable used to check for XML Sampling rejections
  - Looks in migration tables in SDWIS database
- **ValidationDataAccess**
  - Class library used by other projects for accessing databases
  - Also defines data model for Entity Framework
- **ValidationUtilities**
  - Class library containing utility functions such as directory and file access, error logging, and email

Environment Setup

- Visual Studio 2015 or later with EntityFramework
- SQL Server 2012 or later
  - Database connections are defined in ValidationDataAccess.Settings.settings
  - Contains connections for test and production instances
  - Connects to two databases
    - SDWIS database
    - A second database containing our own validation tables (Ours is called SDWISAddOn)
      - Validation tables contain sample and result data as well as certain flags and timestamps to track the validation process, including:
      - **DataValidation** – Boolean used to show whether a sample passed our custom validation
      - **DataValidationDate**
      - **StaffValidation** – Boolean used to show whether staff agreed with a rejection or sent a rejected sample back for revalidation
        - Only used if a sample fails DataValidation or XmlSamplingValidation
      - **StaffValidationDate**
      - **StaffValidator** – Records the username of the staff member who reviewed the sample rejection
      - **XmlSamplingValidation** – Boolean used to show if sample was rejected by XML Sampling
        - This field only gets set if a rejection is found, so the only values here will be false or null

                - If a sample has been submitted to XML Sampling already and this field is left blank, it is assumed the sample was migrated
            - **XmlSamplingValidationDate**
            - **Errors** – If sample fails either validation step, the error message will be stored here
            - **XmlParseDate** – Stores the original date the XML file from the DSE was parsed and validated
            - **XmlCompilationDate** – Records the date the sample was output to an XML file to be submitted to XML Sampling
        - More information about the database tables and what different combinations of values means for the validation can be found below
    - Contains connection strings for test and production instances
    - Entity Framework will automatically create the validation database the first time the application is run, provided the application has permission to do so in the database defined in the connection string
    - There is a stored procedure on the SDWIS database that's used by the application, called MIGRATION_ERROR_REPORT
        - The SQL code for this stored procedure is copied below
    - There is an additional table used to define holding times for certain analytes that won't be automatically created by Entity Framework. It has four columns:
        - ID: int, primary key
        - PWS_NAME: nvarchar(40)
        - PWSID: nvarchar(12)
        - HOUR: int
- Environment Settings
    - Defined in ValidationDataAccess.Settings.settings
    - UserDomain
        - The application uses the Windows identity of the person logged in to the computer for database logging purposes
        - The domain is listed in the settings file so that it can be stripped from the identity returned and so the application can store the basic username
    - UseProductionSettings
        - Set to true to use production settings, or false to run in test mode
- Staging folders for XML migration files
    - Defined in ValidationUtilities.Settings.settings
    - Should be on the same machine where executable files are run, or on a network path accessible to them
    - Requires 4 folders
        - DseOutputPath – location where DSE outputs its XML files
        - DseOutputBackup – when files are parsed from DseOutputPath, they are deleted from there and backed up to DseOutputBackup
        - XMLSamplingInputPath – location of XML files to be uploaded to XML Sampling
            - If XML Sampling is set to run automatically, this folder should be set as the input folder in the XML Sampling configuration
        - XMLSamplingInputBackup – all files prepared for XML Sampling are backed up here, since, when run using the scheduler, XML Sampling deletes all files it processes

- Email settings
    - Defined in ValidationUtilities.Settings.settings
    - Email is used for alerting staff of the outcomes of automatic jobs and for sending rejection messages to labs submitting samples
- Executable files
    - Should be installed on a server or computer in a location where they have access to the database and to XML output folders
    - Should be scheduled to run automatically using Windows Task scheduler
    - Schedule should be coordinated with DSE and XML Sampling
- Windows Desktop Application
    - CmdpErrorReviewer is a Microsoft ClickOnce application that is installed on the computers of staff
    - Installation files are kept on an accessible network drive or FTP server, and are used to install the application on individual computers
    - Every time the application runs, it will check the network location for updates
    - Visual Studio publishes the installer to the network drive, as defined in the CmdpErrorReviewer project properties, under the Publish tab

Sequence of Migration and Validation Events

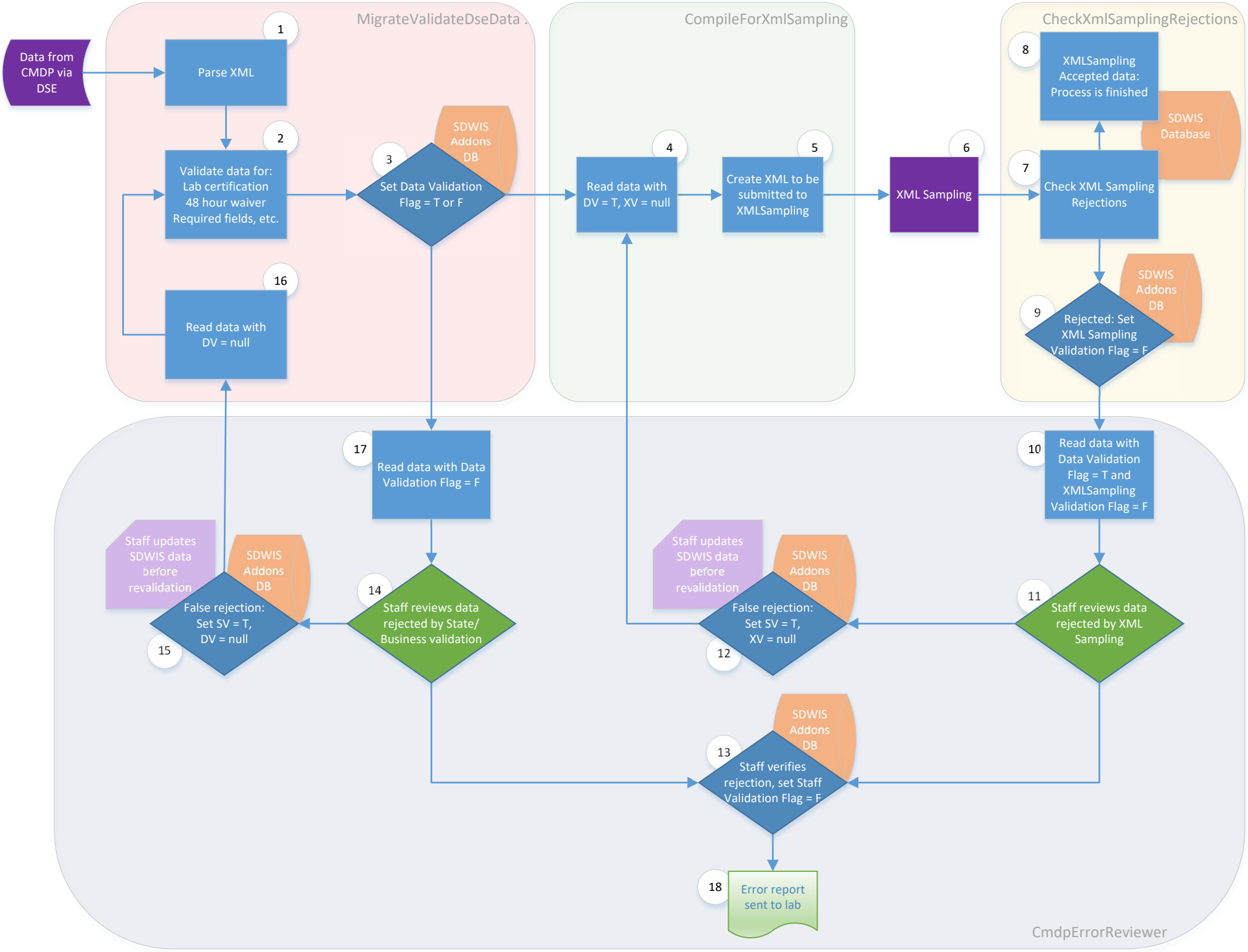
- The following sequence is a series of scheduled tasks that run in a coordinated sequence to validate and migrate sampling data (see also the associated flowchart document)
- **CMDP**
    - Lab submits sampling data
- **DSE**
    - Migrates data from CMDP to an XML file in DseOutputPath
- **MigrateValidateDseData**
    - Parses all files in DseOutputPath folder into C# objects
    - Copies files into DseOutputBackup
    - Deletes files from DseOutputPath
    - Gets any samples from the database that at first failed our validation but then were corrected (e.g. added lab certifications that should have been in SDWIS but weren't) to add to the samples just read in from XML to be validated
    - Validate all samples and results according to our criteria:
        - Collection Date required
        - Collection Time Required
        - Lab Receipt Date Required
        - Sampling Point Required
        - Sampling Location Required
        - Lab sample ID less than 20 characters
        - Analysis start date required
        - Analysis start time required
        - Holding time check
        - Lab analyte method certification check
        - Check for E. coli result if total coliform result is positive
        - And more: look in MigrateValidateDseData.Validator.cs for all validation checks

    - When a result is rejected, the error is marked on the sample
    - Set flags on sample data objects
        - XMLParseDate is set
        - DataValidation is set: 1 = sample or result passed validation, 0 = failed validation
        - DataValidationDate is set
        - If DataValidation = 0, error message is added
    - Update database with parsed data and flag values
    - Send email to admins with:
        - List of files parsed
        - List of sampleIDs for samples that passed
        - List of sampleIDs for samples with errors (including samples that passed, but that have rejected results)
- **CompileForXmlSampling**
    - Creates a new XML file with all samples that passed the data validation
        - If all samples from a given DSE output file are accepted, the newly created XML file may be exactly the same as the DSE output
    - Selects files from database WHERE
        - Data validation flag is set to true AND
        - XML Sampling Validation flag is null AND
        - Either
            - XML Compilation date is null OR
            - XML Compilation date is not null and Staff Validation flag is true
    - Set XML Compilation date for all samples selected and for all results linked to those samples
    - Write XML for selected samples in new file in the XMLSamplingInputPath folder, and creates a backup copy in XMLSamplingInputBackup
- **XMLSampling**
    - Pulls files from DSE_Validated_for_XMLSampling, processes them, and deletes them
- **CheckXmlSamplingRejections**
    - Get error report from SDWIS database using stored procedure MIGRATION_ERROR_REPORT
        - All rejections for a given date are returned by this stored procedure, so if XML Sampling is run more than once during a single day, all rejections for the day will be returned
    - If there are no rejections, the process stops here
    - If there are rejections, Iterate through rejections obtained from stored procedure
        - Copy the rejection message to the record with matching lab sample ID on CmdpSamples
        - Set XML Sampling validation flag to false
        - Set Xml Sampling Validation date

Review Rejections

- **CmdpErrorReviewer**
    - The error reviewer is a Windows application used by staff reviewers and does not work on a schedule like the other executables
    - After the automated tasks are run and rejection messages are generated, the Error Reviewer will show a list of rejected samples
        - Samples shown are those rejected by both the initial data validation and by XML Sampling

- When a sample is selected, there are three options:
- If **Send To Lab** button is selected
    - Email form is displayed with email body prepopulated with error messages
    - Email address for lab is prepopulated if it is set for the lab in SDWIS
    - After email is sent, flags are set on database record
        - StaffValidation = 0
        - StaffValidationDate is set
        - StaffValidator is set
- If **Revalidate** button is selected
    - Flags are set on database record
        - StaffValidation = 1
        - StaffValidationDate is set
        - StaffValidator is set
- If Remove button is selected
    - Flags are set on database record
        - StaffValidation = 0
        - StaffValidationDate is set
        - StaffValidator is set
- Note: these steps occur the same way whether the error we are checking for is for is a data validation error or an xml sampling validation error. If it is a data error, the DataValidation flag will be set to 0 and XmlSamplingValidation flag will be blank. If it is a xml sampling error, the DataValidation flag will be set to 1 and XmlSamplingValidation flag will be set to 0


MigrateValidateDseData
CompileForXmlSampling
CheckXmlSamplingRejections
CmdpErrorReviewer
Data from CMDP via DSE
1
Parse XML
2
Validate data for: Lab certification 48 hour waiver Required fields, etc.
3
Set Data Validation Flag = T or F
SDWIS Addons DB
4
Read data with DV = T, XV = null
5
Create XML to be submitted to XMLSampling
6
XML Sampling
7
Check XML Sampling Rejections
8
XMLSampling Accepted data: Process is finished
SDWIS Database
9
Rejected: Set XML Sampling Validation Flag = F
SDWIS Addons DB
10
Read data with Data Validation Flag = T and XMLSampling Validation Flag = F
11
Staff reviews data rejected by XML Sampling
12
Staff updates SDWIS data before revalidation
False rejection: Set SV = T, XV = null
SDWIS Addons DB
13
Staff verifies rejection, set Staff Validation Flag = F
SDWIS Addons DB
14
Staff reviews data rejected by State/ Business validation
15
Staff updates SDWIS data before revalidation
False rejection: Set SV = T, DV = null
SDWIS Addons DB
16
Read data with DV = null
17
Read data with Data Validation Flag = F
18
Error report sent to lab

Example Migration Scenarios

Numbers used here refer to numbers in flowchart above. These scenarios are not exhaustive.

- Submission passes all validation steps and migrates to SDWIS
    - 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8
- Submission passes state data validation but is legitimately rejected by XML Sampling
    - 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 9 -> 10 -> 11 -> 13 -> 18
- Submission passes state data validation but is rejected by XML Sampling because monitoring period association isn't updated in SDWIS
    - 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 9 -> 10 -> 11 -> 12 -> 4 -> 5 -> 6 -> 7 -> 8
    - In reviewing data in step 12, staff update monitoring period association in SDWIS so the submission will be accepted by XML Sampling on the second try
- Submission legitimately fails state data validation
    - 1 -> 2 -> 3 -> 17 -> 14 -> 13 -> 18
- Submission fails state data validation because lab analyte method associations are not up to date in SDWIS
    - 1 -> 2 -> 3 -> 17 -> 14 -> 15 -> 16 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8
    - In reviewing data in step 15, staff update lab analyte method associations in SDWIS so the submission will pass data validation on the second try
- Submission fails state data validation because lab analyte method associations are not up to date in SDWIS, which is corrected, and then is rejected by XML Sampling because monitoring period association isn't updated in SDWIS, which is also corrected
    - 1 -> 2 -> 3 -> 17 -> 14 -> 15 -> 16 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 9 -> 10 -> 11 -> 12 -> 4 -> 5 -> 6 -> 7 -> 8
    - In reviewing data in step 15, staff update lab analyte method associations in SDWIS so the submission will pass data validation on the second try
    - In reviewing data in step 12, staff update monitoring period association in SDWIS so the submission will be accepted by XML Sampling on the second try

Database Flag Interpretation Key							
Dates shown below are used only to show the timing of one even relative to another							
DV	**DVDate**	**SV**	**SVDate**	**XV**	**XVDate**	**XMLComp**	
T	1/1/2018						Passed our validation but XML Sampling not run yet.
T	1/1/2018					1/1/2018	Passed our validation and submitted to XML Sampling. Presumed migrated unless rejection message is found later
T	1/1/2018			**F**	1/1/2018	1/1/2018	Passed our validation, rejected by XMLSampling
T	1/1/2018	**F**	1/2/2018	**F**	1/1/2018	1/1/2018	Passed our validation, rejected by XMLSampling, rejection verified by staff
T	1/1/2018	**T**	1/2/2018		1/1/2018	1/1/2018	Passed our validation, rejected by XML Sampling, rejection overruled by staff, but not yet recompiled for submission to XML Sampling
T	1/1/2018	**T**	1/2/2018		1/1/2018	1/3/2018	Passed our validation, rejected by XML Sampling, rejection overruled by staff, recompiled for submission to XML Sampling
	1/1/2018	**T**	1/2/2018				Failed our validation, rejection overruled by staff, but not yet revalidated
T	1/3/2018	**T**	1/2/2018				Initially failed our validation, but was revalidated after staff review and passed second validation, but not yet compiled for XML Sampling
T	1/3/2018	**T**	1/2/2018			1/3/2018	Initially failed our validation, but was revalidated after staff review and passed second validation, compiled for XML Sampling and presumed migrated
T	1/3/2018	**T**	1/2/2018	**F**	1/3/2018	1/3/2018	Failed our validation, rejection overruled, then passed our validation on second try, then rejected by XML Sampling
F	1/1/2018						Failed our validation
F	1/1/2018	**F**	1/2/2018				Failed our validation, and rejection verified by staff

```
/****** Object:  StoredProcedure [dbo].[MIGRATION_ERROR_REPORT]    Script Date: 8/15/2018
4:06:40 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

ALTER PROCEDURE [dbo].[MIGRATION_ERROR_REPORT]
(
    @migdate datetime
)
AS
Begin

       select j.tmgjob_is_number as job_number, e.b_record_id as sample_number,
                     e.b_xmldoc_record_id, e.b_lab_sample_num,
                     e.b_pws_number, e.b_lab_id_number, e.b_analyte_cd,
e.b_wsf_st_asgn_id,
                     e.b_sampling_point, e.b_sampling_loc, e.b_mon_per_beg_dat,
                     e.b_mon_per_end_dat, '' as b_result_type, '' as b_data_quality,
                     e.b_record_type as b_stag_tbl_name, err.b_error_number as
b_error_cd,
                     err.b_error_message as b_error_desc, 'SCHEMA' as error_type
       from tmgjob j, tmgjbrec e, tmgsjerr err
       where j.tmgjob_is_number = err.tmgjob_is_number
       and j.tmgjob_is_number = e.tmgjob_is_number
       and e.b_record_id = err.b_record_id
       and j.b_completion_ts > @migdate
       and j.b_completion_ts < @migdate + 1
       union
       select s.*, je.b_stag_tbl_name, e.b_error_cd, e.b_error_desc,
                     'BUSINESS' as error_type
       from tmgjob j, tmgjberr je, tmgerror e,
              (
              select sr.tmgjob_is_number as job_number, sr.tmgsar_is_number as
sample_number,
                     sr.b_xmldoc_record_id, sr.b_lab_sample_num,
                     sr.b_pws_number, sr.b_lab_id_number, sr.b_analyte_cd, '' as
b_wsf_st_asgn_id,
                     '' as b_sampling_point, '' as b_sampling_loc, null as
b_mon_per_beg_dat,
                     null as b_mon_per_end_dat, '' as b_result_type, '' as b_data_quality
              from tmgsar sr
              union
              select s.tmgjob_is_number as job_number, s.tmgsampl_is_number as
sample_number,
                     s.b_xmldoc_record_id, s.b_lab_sample_num,
                     s.b_pws_number, s.b_lab_id_number, '' as b_analyte_cd,
s.b_wsf_st_asgn_id,
                     s.b_sampling_point, s.b_sampling_loc, null as b_mon_per_beg_dat,
                     null as b_mon_per_end_dat, '' as b_result_type, '' as b_data_quality
              from tmgsampl s
              union
              select ssm.tmgjob_is_number as job_number, ssm.tmgsmpsm_is_number as
sample_number,
                     ssm.b_xmldoc_record_id, '' as b_lab_sample_num,
                     ssm.b_pws_number, ssm.b_lab_id_number, ssm.b_analyte_cd,
                     ssm.b_wsf_stat_asgn_id as b_wsf_st_asgn_id, '' as b_sampling_point,
```

```
                    '' as b_sampling_loc, ssm.b_mon_per_beg_dat, ssm.b_mon_per_end_dat,
                    '' as b_result_type, '' as b_data_quality
             from tmgsmpsm ssm
             union
             select ssr.tmgjob_is_number as job_number, ssr.tmgssr_is_number as
sample_number,
                    ssr.b_xmldoc_record_id, '' as b_lab_sample_num,
                    ssr.b_pws_number, '' as b_lab_id_number, ssr.b_analyte_cd,
                    ssr.b_wsf_stat_asgn_id as b_wsf_st_asgn_id, '' as b_sampling_point,
                    '' as b_sampling_loc, ssr.b_mon_per_beg_dat, ssr.b_mon_per_end_dat,
                    ssr.b_result_type, ssr.b_data_quality
             from tmgssr ssr
             ) s
      where s.job_number = j.tmgjob_is_number
      and j.b_completion_ts > @migdate
      and j.b_completion_ts < @migdate + 1
      and je.b_record_id = s.sample_number
      and je.tmgjob_is_number = s.job_number
      and je.b_error_cd = e.b_error_cd
      order by error_type desc, b_lab_id_number, b_lab_sample_num, b_analyte_cd,
sample_number

END
```