SQL Delta Command Line Specification
Version 1.1, Released April 21, 2005

This document refers to SQL Delta version 3.1.

Introduction

SQL Delta can be run without the need for user intervention by passing a structured xml file in the command line. This file must contain a valid set of commands allowing SQL Delta to connect to a database server to perform known tasks.

The command line process does not output any prompts or notices to the user interface and all errors are sent to CommandLineErrors.txt. Feedback can be sent to a user through email which can be defined within the command script.

Quick Start

The quickest and easiest way to get started with the command line process is to open SQL Delta and perform a comparison on the databases required for scripting. Once compared, save the project to a file, for example base.sdp. This sdp file is an XML file containing most of the information needed for the command line script.

Run SQL Delta with Script

Add the command line file name to the end of the executable command.

sqldelta script filename

For example
“C:\program files\sql delta\sqldelta.exe” base.sdp

Base Script Details

Each command line script is an xml style file using the definitions of W3C for XML. Each script must contain a single <PROJECT> element starting with <PROJECT> and ending with </PROJECT>. The xml version and encoding definition is not strictly required by the script processor however should be retained for consistency with the standard. The script processor can process Unicode or normal ASCII and does not refer to the encoding definition to determine the text style.

All commands are case insensitive and empty elements, that is elements with no text between the open and close are ignored. For example <CONNECT></CONNECT> is ignored.

The command elements are processed by SQL Delta in the order required by SQL Delta and not the order presented in the xml script file.

Example Base.sdp
<?xml version="1.0" encoding="UTF-8"?>
<PROJECT> …. </PROJECT>

Command List – Primary Section Elements

The following is a list of primary section elements used to control SQL Delta and these refer to the primary functions of SQL Delta. Each primary section element can exist once within a project. If a primary section element is duplicated then the first element detected will override all subsequent duplicates. It is not currently possible to execute multiple comparisons within one project or have multiple projects within one script.

Primary Section Elements

CONNECTIONS
STRUCTURE
DATA
REPORTS
EMAILS
OPTIONS

The following Optional Primary Elements provide additional control of SQL Delta and the script process.

Optional Primary Elements

VERSION
SOUND
CFGRANDOMFILE
RANDOMFILE

CONNECTIONS: mandatory
Attributes: None

Syntax: <CONNECTIONS>.. elements ..</CONNECTIONS>

Defines the connection details for one or two databases. Within the CONNECTIONS element there must be at least one LITEM element defining the individual connection.

LITEM: mandatory
Attributes: None
Element of: CONNECTIONS

Syntax: <LITEM>.. elements ..</LITEM>

CONNECTIONS may have one or two LITEM elements, a single element will allow view structure tasks such as reporting and two LITEM elements allows both structure and data compare features to be available. If more than two LITEM elements are defined within CONNECTIONS then the last valid LITEM will be used. When defining a manual connection Windows Authentication is the default authentication method.

Within LITEM the following elements are available:
SIDE, CONNECT, DATABASE, SERVERNAME, USER, PASSWORD, OFFLINE

Each LITEM must have a connection using either CONNECT, the manual connection elements or OFFLINE. If multiple elements exists then CONNECT, OFFLINE will take precedence in order.

Example
```
<CONNECTIONS>
  <LITEM>
    <SIDE>A</SIDE>
    <CONNECT>Test</CONNECT>
  </LITEM>
</CONNECTIONS>
```

SIDE: mandatory
Attributes: None
Element of: CONNECTIONS, LITEM

Syntax: <SIDE>A</SIDE> or <SIDE>B</SIDE>

SQL Delta connections are source and target connections where the target is usually the database to be updated. The SIDE element indicates whether an LITEM is (A) the source or (B) the target. SIDE B may precede SIDE A in the script file.

CONNECT
Attributes: None
Element of: CONNECTIONS, LITEM

Syntax: <CONNECT>Connection friendly name text</CONNECT>

CONNECT retrieves the connection details from the SQL Delta connection manager using the supplied connection friendly name. The connection name must exist in the connection manager otherwise an error is generated and the script processing terminated. CONNECT takes precedence over manual connection elements. The connection friendly name is case insensitive.

SERVERNAME
Attributes: None
Element of: CONNECTIONS, LITEM

Syntax: <SERVERNAME>text</SERVERNAME>

SERVERNAME is part of the manual connection elements and defines the database server. The text can be IP Address, URI or local registered server name. This element is mandatory when using the manual connection elements.

DATABASE
Attributes: None
Element of: CONNECTIONS, LITEM

Syntax: <DATABASE>text</DATABASE>

DATABASE is part of the manual connection elements and text identifies the database within the database server. This element is mandatory when using the manual connection elements.

USER
Attributes: None
Element of: CONNECTIONS, LITEM

Syntax: <USER>text</USER>

USER is part of the manual connection elements and text identifies the user name required for connection to the database server. If connecting with Windows Authentication then USER and PASSWORD are not required elements and must be omitted from LITEM.

PASSWORD
Attributes: None
Element of: CONNECTIONS, LITEM

Syntax: <PASSWORD>text</PASSWORD>

PASSWORD is part of the manual connection elements and text identifies the password required for connection to the database server. If connecting with Windows Authentication then USER and PASSWORD are not required elements and must be omitted from LITEM. The password must be clear text and if encrypted passwords are required then use Windows Authentication or the CONNECT element instead of declaring a manual connection.

OFFLINE
Attributes: None
Element of: CONNECTIONS, LITEM

Syntax: <OFFLINE>filename</OFFLINE>

OFFLINE allows the user to load an offline schema file instead of connecting to a database server. The filename must exist and have a format consistent with Windows path and file naming conventions. OFFLINE mode cannot be used in conjunction with CONNECT or manual connection elements.

eg. <OFFLINE>c:\test.xml</OFFLINE>

STRUCTURE
Attributes: None

Syntax: <STRUCTURE>.. elements ..</STRUCTURE>

Structure command performs a database schema comparison and exports the schema differences to a script file. For security and database integrity SQL Delta cannot run an automatic update of the target database. A script file is only created if one or more differences are found in the databases.

Within STRUCTURE there must be a FILENAME, DIRECTION and a selection element for the comparison and export.

```
<STRUCTURE>
 <FILENAME>x:\sql delta\Test.sql</FILENAME>
 <DIRECTION>B</DIRECTION>
 <SELECTALL>True</SELECTALL>
</STRUCTURE>
```

FILENAME
Attributes: None
Element of: STRUCTURE

Syntax: <FILENAME>.. filename .. </FILENAME>

FILENAME specifies the path and filename for the exported script. The file is only created if a difference is found and if the filename exists then it will be overwritten with the new script.

ALTFILENAME
Attributes: None
Element of: STRUCTURE

Syntax: <ALTFILENAME>.. filename .. </ALTFILENAME>

SQL Delta allows comparisons and scripts to be created in two directions within a single comparison process. ALTFILENAME specifies the path and filename for the exported script of the second script, if one exists. The file is only created if a difference is found and if the filename exists then it will be overwritten with the new script.

The second script side is determined by the DIRECTION element below where <STRUCTURE><DIRECTION>A<DIRECTION> means that the ALTFILENAME will be any changes going to side B and the reverse if the structure direction is B.

DIRECTION
Attributes: None
Element of: STRUCTURE

Syntax: <DIRECTION>.. A/B .. </DIRECTION>

The default direction for comparison can be set to A meaning update the source database with target differences or B meaning update the target database with source differences.

SELECTALL
Attributes: None
Element of: STRUCTURE

Syntax: <SELECTALL>.. True/False .. </SELECTALL>

The selection elements including SELECTALL are toggle commands allowing the user to define which database items are to be included or excluded in a change script. The SELECTALL command is executed first allowing the user to start with all items selected or all items deselected. Further selection processing is not required if <SELECTALL>true</SELECTALL> however further processing is required if false.

SELECTED
Attributes: None
Element of: STRUCTURE

Syntax: <SELECTED>.. elements .. </SELECTED>

This is a container element used then choosing database objects to be included or excluded in script processing. By using a combination of SELECTALL and SELECTED the user can choose all different database objects to be scripted except SELECTED objects or the inverse combination. SELECTED requires one or more SITEM elements.

SITEM
Attributes: None
Element of: STRUCTURE, SELECTED

Syntax: <SITEM>.. elements .. </SITEM>

SITEM is a contain item for each selection item and requires NAME, UPDATE and DIRECTION. One or more SITEM elements can be part of a SELECTED container element.

Example
```
<SELECTALL>true</SELECTALL>
<SELECTED>
  <SITEM>
    <NAME>test_view</NAME>
    <UPDATE>false</UPDATE>
    <DIRECTION>B</DIRECTION>
  </SITEM>
</SELECTED>
```

NAME
Attributes: None
Element of: STRUCTURE, SELECTED, SITEM

Syntax: <NAME>.. text .. </NAME>

NAME identifies the database object to action and must include the owner name in the format [owner name].[database object name]. The database object name must exist and wildcard characters are not supported. The square brackets are not required.

UPDATE
Attributes: None
Element of: STRUCTURE, SELECTED, SITEM

Syntax: <UPDATE>.. true/false .. </UPDATE>

UPDATE element defines if the database object defined by NAME should be included in the change script by using the command true or excluded from the script by using the command false.

DIRECTION
Attributes: None
Element of: STRUCTURE, SELECTED, SITEM

Syntax: <DIRECTION>.. A/B .. </DIRECTION>

The DIRECTION element allows the user to choose the direction of the change. Typically the direction would by B meaning change the target. This feature also allows the user to create two script files by having a two directional script, that is changing A and B within the one comparison.

If the user does choose a two directional script then they must ensure the ALTFILENAME has been defined otherwise the second direction change script will be discarded.

DATA
Attributes: None

Syntax: <DATA>.. elements ..</DATA>

The DATA command performs a data comparison of the two databases and exports a change script. For security and data integrity SQL Delta does not automatically update changes. A script file is only created if changes are found.

For the data compare results report the user must include the DATA command and use the STOREIDENTICAL, STOREMODIFIED, STOREMISSING, STOREADDED commands to ensure the data is available for the report. If a report is not required then STOREIDENTICAL is not usually required however the other STORE elements are required if changes are detected and are to be scripted.

STOREIDENTICAL, STOREMODIFIED, STOREMISSING, STOREADDED
Attributes: None
Element of: DATA

Syntax: <STOREIDENTICAL>.. true/false ..</STOREIDENTICAL>
< STOREMODIFIED >.. true/false ..</ STOREMODIFIED >
< STOREMISSING >.. true/false ..</ STOREMISSING >
< STOREADDED >.. true/false ..</ STOREADDED >

The STORE elements determine whether SQL Delta saves the data locally to hard disk after a data comparison is performed. In many cases STOREIDENTICAL is not required and can save a significant amount of disk space if set to false. To report on identical data items the user must set STOREIDENTICAL.

The other STORE elements are required to script changes if they exist.

FILENAME
Attributes: None
Element of: DATA

Syntax: <FILENAME>.. filename .. </FILENAME>

FILENAME specifies the path and filename for the exported script. The file is only created if a difference is found and if the filename exists then it will be overwritten with the new script.

ALTFILENAME
Attributes: None
Element of: DATA

Syntax: <ALTFILENAME>.. filename .. </ALTFILENAME>

SQL Delta allows comparisons and scripts to be created in two directions within a single comparison process. ALTFILENAME specifies the path and filename for the exported script of the second script, if one exists. The file is only created if a difference is found and if the filename exists then it will be overwritten with the new script.

The second script side is determined by the DIRECTION element below where <DATA><DIRECTION>A<DIRECTION> means that the ALTFILENAME will be any changes going to side B and the reverse if the data direction is B.

DIRECTION
Attributes: None
Element of: DATA

Syntax: <DIRECTION>.. A/B .. </DIRECTION>

The default direction for comparison can be set to A meaning update the source database with target differences or B meaning update the target database with source differences.

DISABLETRIGGERS
Attributes: None
Element of: DATA

Syntax: <DISABLETRIGGERS>.. True/False .. </DISABLETRIGGERS>

When generating the change script the user may require that triggers are not fired during an insert or update process. This command allows the user to specify the option for all table objects.

DISABLEINDEXANDCONSTRAINTS
Attributes: None
Element of: DATA

Syntax: < DISABLEINDEXANDCONSTRAINTS >.. True/False .. </DISABLEINDEXANDCONSTRAINTS >

When generating the change script the user may require that indexes and constraints are disabled prior to an insert or update process. This command allows the user to specify the option for all table objects.

SELECTALL
Attributes: None
Element of: DATA

Syntax: <SELECTALL>.. True/False .. </SELECTALL>

redundant

SELECTED
Attributes: None
Element of: DATA

Syntax: <SELECTED>.. elements .. </SELECTED>

Unlike the structure compare the SELECTED container is required to define which tables are to be included in the data compare. The data compare requires information about the primary key as well as allowing data filters on items and the elements with the SELECTED container provide a mechanism for this selection.

SITEM
Attributes: None
Element of: DATA, SELECTED

Syntax: <SITEM>.. elements .. </SITEM>

SITEM is a contain item for each selection item and requires NAME, KEY, COLUMNS and DIRECTION. One or more SITEM elements can be part of a SELECTED container element.

Example
```
<SELECTED>
  <SITEM>
    <NAME>testtable</NAME>
    <UPDATE>false</UPDATE>
    <DIRECTION>B</DIRECTION>
  </SITEM>
</SELECTED>
```

NAME
Attributes: None
Element of: DATA, SELECTED, SITEM

Syntax: <NAME>.. text .. </NAME>

NAME identifies the database object to action and must include the owner name in the format [owner name].[database object name]. The database object name must exist and wildcard characters are not supported. The square brackets are not required.

KEY
Attributes: None
Element of: DATA, SELECTED, SITEM

Syntax: <KEY>.. text .. </KEY>

KEY identifies the primary key, unique index or comma separated list of key columns making up a unique key for data comparison.

COLUMNS
Attributes: None
Element of: DATA, SELECTED, SITEM

Syntax: <COLUMNS>.. text .. </COLUMNS>

COLUMNS is a quoted and comma separated list of table columns to be used when inserting or updating.

Example
<COLUMNS>”ID”,”Name”,”Address”,”ZIP”,”UNIQUEVAL”</COLUMNS>

SELECTEDCOLUMNS
Attributes: None
Element of: DATA, SELECTED, SITEM

Syntax: <SELECTEDCOLUMNS>.. text .. </SELECTEDCOLUMNS>

SELECTEDCOLUMNS is a quoted and comma separated list of table columns to be used for comparing the two tables.

Example
<COLUMNS>”ID”,”Name”,”Address”,”ZIP”</COLUMNS>

FILTERA
Attributes: None
Element of: DATA, SELECTED, SITEM

Syntax: <FILTERA>.. text .. </FILTERA>

FILTERA is an SQL compliant where clause that allows the user to filter the table data for SIDE A table. The clause does not require the WHERE statement and does not need to be enclosed in quotes. String compare items do need to be enclosed in quotes.

FILTERB
Attributes: None

Element of: DATA, SELECTED, SITEM

Syntax: <FILTERB>.. text .. </FILTERB>

FILTERB is an SQL compliant where clause that allows the user to filter the table data for SIDE B table. The clause does not require the WHERE statement and does not need to be enclosed in quotes. String compare items do need to be enclosed in quotes.

UPDATE: PROPOSE REMOVAL
Attributes: None
Element of: DATA, SELECTED, SITEM

Syntax: <UPDATE>.. true/false .. </UPDATE>

UPDATE element defines if the database object defined by NAME should be included in the change script by using the command true or excluded from the script by using the command false.

UPDATEMODIFIED, UPDATEMISSING, UPDATEADDED
Attributes: None
Element of: DATA, SELECTED, SITEM

Syntax: <UPDATEMODIFIED>.. true/false .. </UPDATEMODIFIED>
< UPDATEMISSING >.. true/false .. </ UPDATEMISSING >
< UPDATEADDED >.. true/false .. </ UPDATEADDED >

The UPDATE elements allow the user to restrict updates to specific changes, for example missing records and modified records which would force a script to perform an insert and update respectively may be acceptable however missing records which cause a delete from the table may not be desirable and the user can set <UPDATEADDED>false</UPDATEADDED> to ensure the script does not delete data from the table.

DIRECTION
Attributes: None
Element of: DATA, SELECTED, SITEM

Syntax: <DIRECTION>.. A/B .. </DIRECTION>

The DIRECTION element allows the user to choose the direction of the change. Typically the direction would by B meaning change the target. This feature also allows the user to create two script files by having a two directional script, that is changing A and B within the one comparison.

If the user does choose a two directional script then they must ensure the ALTFILENAME has been defined otherwise the second direction change script will be discarded.

DISABLETRIGGERS
Attributes: None
Element of: DATA, SELECTED, SITEM

Syntax: <DISABLETRIGGERS>.. True/False .. </DISABLETRIGGERS>

This element allows the user to control disabling triggers at a table level rather than for all. When generating the change script the user may require that triggers are not fired during an insert or update process. This command allows the user to specify the option for all table objects.

DISABLEINDEXANDCONSTRAINTS
Attributes: None
Element of: DATA, SELECTED, SITEM

Syntax: < DISABLEINDEXANDCONSTRAINTS >.. True/False ..
</DISABLEINDEXANDCONSTRAINTS >

This element allows the user to control disabling indexes and constraints at a table level rather than for all. When generating the change script the user may require that indexes and constraints are disabled prior to an insert or update process. This command allows the user to specify the option for all table objects.

REPORTS
Attributes: None

Syntax: <REPORTS>.. elements ..</REPORTS>

Defines the report details for a set of reports to be output by SQL Delta. Within the REPORTS element there must be at least one RITEM element defining a report however many RITEM elements can be defined allowing many reports to be output.

RITEM: mandatory
Attributes: None
Element of: REPORTS

Syntax: <RITEM>.. elements ..</RITEM>

REPORTS may have many RITEM elements, each one defining the report to output, output method and other option report criteria.

NAME: mandatory
Attributes: None
Element of: REPORTS, RITEM

Syntax: <NAME>.. text ..</NAME>

The NAME element must be one of the following named reports:

- Detailed Comparison Report
- Structure Compare Results
- Structure Details
- Data Compare Results

If a data comparison has not been performed then attempting to print a data compare report will be ignored. Also attempting to print a comparison report when a compare has not been performed will be ignored.

OUTPUTTYPE: mandatory
Attributes:
Element of: REPORTS, RITEM

Syntax: <OUTPUTTYPE>.. text ..</OUTPUTTYPE>

The available output type depends on the report and the following options are available:

- Detailed Comparison Report
  - HTML
- Structure Compare Results & Structure Details
  - Printer, HTML, PDF, Excel, RTF, JPEG, BMP
- Data Compare Results
  - Printer, CSV, HTML, XML, Excel, Word

For example:
```
<RITEM>
    <NAME>Structure Details</NAME>
    <OUTPUTTYPE>HTML</OUTPUTTYPE>
    <FILENAME>c:\output\latest.html</FILENAME>
    <SIDE>B</SIDE>
</RITEM>
```

FILENAME
Attributes: None
Element of: REPORTS, RITEM

Syntax: <FILENAME>.. text ..</FILENAME>

Where the output is to a file then a filename must be defined. It is not possible to combine output to printer and a filename to attempt to generate both output to printer and file.

SIDE
Attributes: None
Element of: REPORTS, RITEM

Syntax: <SIDE>.. A/B ..</SIDE>

This element is only appropriate for the Structure Details Report and indicates which database to print. Including two SIDE elements will not produce two reports and the last element will be the defining element.

TYPEFILTERS
Attributes: None
Element of: REPORTS, RITEM

Syntax: <TYPEFILTERS>.. options ..</TYPEFILTERS>

TYPEFILTERS provides a mechanism to limit the reports to specific database object types. By default all filters are off (false) ensuring all data will be output. Options may be one or more of the following elements:

> Tables, Views, Procedures, Triggers, Functions, Defaults, Rules, UDTs, Users, Roles, Catalogs, Properties – not available for Structure Details report.
>
> In the format of <option name>true/false</option name>
>
> For example:
> <TYPEFILTERS>
> <TABLES>true</TABLES>
> </TYPEFILTERS>
> This example will filter out tables and remove all table details from the report.

STATUSFILTERS
Attributes: None
Element of: REPORTS, RITEM

Syntax: <STATUSFILTERS>.. options ..</STATUSFILTERS>

STATUSFILTERS provides a mechanism to limit the reports to specific comparison results for the Detailed Comparison Report and Structure Compare Results. By default all filters are off (false) ensuring all data will be output. Options may be one or more of the following elements:

> Identical, Missing, Additional, Different.
>
> In the format of <option name>true/false</option name>
>
> For example:
> <STATUSFILTERS>
> <IDENTICAL>true</IDENTICAL>
> </STATUSFILTERS>
> This example will filter out identical records and remove all from the report.

DETAILFILTERS
Attributes: None

Element of: REPORTS, RITEM

Syntax: <DETAILFILTERS>.. options ..</DETAILFILTERS>

DETAILFILTERS provides a mechanism to limit the reports to specific database object types for the Structure Details Report Only. By default all filters are off (false) ensuring all data will be output. Options may be one or more of the following elements:

> Columns, PrimaryKeysAndIndexes, Constraints, ForeignKeys, Permissions, Properties, Definitions.
>
> In the format of <option name>true/false</option name>
>
> For example:
> <DETAILFILTERS>
> < FOREIGNKEYS >true</ FOREIGNKEYS >
> </DETAILFILTERS>
> This example will filter out Foreign Keys and remove all details from the report.

STRUCTURESETUP
Attributes: None
Element of: REPORTS, RITEM

Syntax: <STRUCTURESETUP>.. elements ..</STRUCTURESETUP>

This option is available for the Detailed Comparison Report only and provides a mechanism for selecting specific database objects by name to include or exclude from a report.

SELECTALL
Attributes: None
Element of: REPORTS, RITEM, STRUCTURESETUP

Syntax: <SELECTALL>.. true/false ..</SELECTALL>

Provides a starting point for the selection process where all objects may be selected for inclusion in the report, the default or all items are excluded from the report. True will select all and false will deselect all.

SELECTED
Attributes: None
Element of: REPORTS, RITEM, STRUCTURESETUP

Syntax: <SELECTED>.. elements ..</SELECTED>

Provides a mechanism to select or deselect specific objects by name for the Detailed Comparison Report.

SITEM
Attributes: None

Element of: REPORTS, RITEM, STRUCTURESETUP, SELECTED

Syntax: <SITEM>.. elements ..</SITEM>

Provides a mechanism to select or deselect specific objects by name for the Detailed Comparison Report.

NAME
Attributes: None
Element of: REPORTS, RITEM, STRUCTURESETUP, SELECTED, SITEM

Syntax: <NAME>.. text ..</NAME>

Is the name of the database object including the owner name. The name text must match an existing database object name and wildcards are not allowed.

SELECTED
Attributes: None
Element of: REPORTS, RITEM, STRUCTURESETUP, SELECTED, SITEM

Syntax: <SELECTED>.. true/false ..</SELECTED>

Is the corresponding object as defined by NAME to be included (true) or excluded (false) from the report.

DATASETUP
Attributes: None
Element of: REPORTS, RITEM

Syntax: <DATASETUP>.. elements ..</DATASETUP>

This option is available for the Detailed Comparison Report only and provides a mechanism for selecting specific database objects by name to include or exclude from a report.

SPLITDATA
Attributes: None
Element of: REPORTS, RITEM, DATASETUP

Syntax: <SPLITDATA>.. true/false ..</SPLITDATA>

If true then forces the report to produce a separate page for each table. By default this option is false.

SELECTALL
Attributes: None
Element of: REPORTS, RITEM, DATASETUP

Syntax: <SELECTALL>.. true/false ..</SELECTALL>

Provides a starting point for the selection process where all objects may be selected for inclusion in the report, the default or all items are excluded from the report. True will select all and false will deselect all.

SELECTED
Attributes: None
Element of: REPORTS, RITEM, DATASETUP

Syntax: <SELECTED>.. elements ..</SELECTED>

Provides a mechanism to select or deselect specific objects by name for the Data Comparison Report.

SITEM
Attributes: None
Element of: REPORTS, RITEM, DATASETUP, SELECTED

Syntax: <SITEM>.. elements ..</SITEM>

Provides a mechanism to select or deselect specific objects by name for the Detailed Comparison Report.

NAME
Attributes: None
Element of: REPORTS, RITEM, DATASETUP, SELECTED, SITEM

Syntax: <NAME>.. text ..</NAME>

Is the name of the database object including the owner name. The name text must match an existing database object name and wildcards are not allowed.

SELECTED
Attributes: None
Element of: REPORTS, RITEM, DATASETUP, SELECTED, SITEM

Syntax: <SELECTED>.. true/false ..</SELECTED>

Is the corresponding object as defined by NAME to be included (true) or excluded (false) from the report.

EMAILS
Attributes: None

Syntax: <EMAILS>.. elements ..</EMAILS>

Allows the user the ability to send one or more emails based on events that occur during SQL Delta operation. This feature is an ideal mechanism for providing user feedback since there is no visual output during operation of the command line process.

One EITEM element must be declared and multiple EITEM elements may be declared. It is the responsibility of the user to correctly declare and confirm the SMTP details and SQL Delta processes the information as-is without verification or validation.

EITEM
Attributes: None
Element of: EMAILS

Syntax: <EITEM>.. elements ..</EITEM>

Each email event must have an EITEM.

HOST
Attributes: None
Element of: EMAILS, EITEM

Syntax: <HOST>.. text ..</HOST>

The name of the SMTP server, either a URI or IP Address, to use as the SMTP server.

PORT
Attributes: None
Element of: EMAILS, EITEM

Syntax: <PORT>.. text ..</PORT>

The port, by default 25, for access to the SMTP server.

USER
Attributes: None
Element of: EMAILS, EITEM

Syntax: <USER>.. text ..</USER>

If the SMTP server requires a login then USER is the user name. By default this is empty and is not required.

PASSWORD
Attributes: None
Element of: EMAILS, EITEM

Syntax: <PASSWORD>.. text ..</PASSWORD>

If the SMTP server requires a login then PASSWORD is the password corresponding to the user name. By default this is empty and is not required.

TO
Attributes: None
Element of: EMAILS, EITEM

Syntax: <TO>.. text ..</TO>

A semi-comma separated list of email addresses to whom the email shall be sent.

CC
Attributes: None
Element of: EMAILS, EITEM

Syntax: <CC>.. text ..</CC>

A semi-comma separated list of email addresses to whom the email shall be CC sent.

BCC
Attributes: None
Element of: EMAILS, EITEM

Syntax: <BCC>.. text ..</BCC>

A semi-comma separated list of email addresses to whom the email shall be BCC sent.

FROM
Attributes: None
Element of: EMAILS, EITEM

Syntax: <FROM>.. text ..</FROM>

The email address of the sender.

SHOWERRORS
Attributes: None
Element of: EMAILS, EITEM

Syntax: <SHOWERRORS>.. true/false ..</SHOWERRORS>

If an error loading an attachment occurs or the attachment file does not exist and SHOWERRORS is set to true an error message will be added to the end of the email body. By default there is no error shown and SHOWERRORS is false.

NOTIFYALWAYS
Attributes: None
Element of: EMAILS, EITEM

Syntax: <NOTIFYALWAYS>.. true/false ..</NOTIFYALWAYS>

Identifies when the email shall be sent, in this case if true an email will be sent upon completion of the comparison regardless of result.

NOTIFYCHANGES
Attributes: None
Element of: EMAILS, EITEM

Syntax: <NOTIFYCHANGES>.. true/false ..</NOTIFYCHANGES>

Identifies when the email shall be sent, in this case if true an email will be sent upon completion of the comparison if there are changes in the database structure or data.

NOTIFYSCRIPT
Attributes: None
Element of: EMAILS, EITEM

Syntax: <NOTIFYSCRIPT>.. true/false ..</NOTIFYSCRIPT>

Identifies when the email shall be sent, in this case if true an email will be sent upon completion of the comparison if one or more script files was successfully created.

NOTIFYERRORS
Attributes: None
Element of: EMAILS, EITEM

Syntax: <NOTIFYERRORS>.. true/false ..</NOTIFYERRORS>

Identifies when the email shall be sent, in this case if true an email will be sent only if an error occurs.

SUBJECT
Attributes: None
Element of: EMAILS, EITEM

Syntax: <SUBJECT>.. text ..</SUBJECT>

Subject text for the email

BODY
Attributes: None
Element of: EMAILS, EITEM

Syntax: <BODY>.. text ..</BODY>

Body of the email

FILES
Attributes: None
Element of: EMAILS, EITEM

Syntax: <FILES>.. elements ..</FILES>

Indicates that file attachments are to be sent with the email. Sending attachments may be limited or prevented by the SMTP server or local anti-virus software.

FITEM
Attributes: None
Element of: EMAILS, EITEM, FILES

Syntax: <FITEM>.. elements ..</FITEM>

Element enclosing the file name.

NAME
Attributes: None
Element of: EMAILS, EITEM, FILES, FITEM

Syntax: <NAME>.. text ..</NAME>

The filename including path, if necessary, of the attachment to send. There is no test made to ensure the file exists and there is no error produced if the file does not exist.

OPTIONS
Attributes: None

Syntax: <OPTIONS>.. elements ..</OPTIONS>

Options element provides a method of configuring SQL Delta prior to the execution of the database load, compare and script. Each option has the syntax:

<*OPTIONELEMENT*>true/false/text*</*OPTIONELEMENT*>
* Some options are integer or text values and are shown below

If an option is omitted then the existing default for the omitted option is used.

The options are:

CompareTables	(True) compare or (False) don't compare
CompareViews	" " "
CompareProcedures	" " "
CompareTriggers	" " "
CompareFunctions	" " "
CompareDefaults	" " "
CompareRules	" " "
CompareUDTs	" " "
CompareUsers	" " "
CompareRoles	" " "
CompareCatalogs	" " "
CompareDiagrams	" " "
ComparePermissions	" " "
CompareProperties	" " "
PreserveColOrder	(True) Force retaining column order of fields
IgnoreDefaults	(True) Ignore when comparing
IgnoreDefaultNames	" " "
IgnoreBindings	" " "
IgnoreIndexesAndConstraints	" " "

IgnoreStatistics	“ “ “
IgnoreCheckConstraints	“ “ “
IgnoreForeignKeys	“ “ “
IgnoreFillFactors	“ “ “
IgnoreIndexAndConstraintNames	“ “ “
IgnoreWithNoCheck	(True) ignore trusted/not trusted foreign keys
IgnoreCollationOrder	(True) Ignore when comparing
IgnoreFileGroups	“ “ “
IgnoreFulltextIndexes	“ “ “
IgnoreComments	“ “ “
IgnoreCRLF	“ “ “
IgnoreTabs	“ “ “
IgnoreSpaces	“ “ “
IgnoreCase	“ “ “
IgnoreSETStatements	“ “ “
IgnoreDataCase	“ “ “
IgnoreTimestamps	“ “ “
IgnoreBlobs	“ “ “
IgnoreGUIDs	“ “ “
DisableTriggers	(True) Force disable triggers in data script
DisableIndexesAndConstraints	“ “ “
IgnoreObjectNameCase	(True) Ignore when comparing
ConnectionTimeout	Numeric value indicating number of milliseconds to wait before issuing a timeout message. A value of 3600 equates to 1 second
CommandTimeout	Numeric value indicating number of milliseconds to wait before issuing a timeout message. A value of 3600 equates to 1 second
RowDelimiter	When exporting report data, a row can be split using the following options: {CR}{LF}, {CR}, {LF}, Semicolon, Comma, Tab or Vertical Bar. The text name as shown is required not the ASCII character. Example: <ROWDELIMITER> {CR}{LF} </ROWDELIMITER>
ColumnDelimiter	When exporting report data, a column can be split using Comma, Semicolon, Tab or Vertical Bar Example: <COLUMNDELIMITER> Comma </COLUMNDELIMITER>
TextQualifier	When exporting report data, text in a column can be qualified using the following options: Double Quote {"}, Single Quote {'}. Example: <TEXTQUALIFIER> Double Quote {"} </TEXTQUALIFIER>

Example:
<IGNORETABS>true</IGNORETABS>

VERSION
Attributes: None

Syntax: <VERSION>.. text ..</VERSION>

The version element is reserved for future use and allows the script processor to determine the type and format of the script. By including a version within a script there is a greater likelihood that the script can be supported by future SQL Delta versions.

SOUND
Attributes: None

Syntax: <SOUND>.. File name/System name ..</SOUND >

The SOUND element plays either a system sound or wave file. The filename can be a predefined Windows sound such as SystemAsterisk, SystemExclamation, SystemHand, or a .wav filename. When the script process is complete this sound will be played. If the sound cannot be located then no sound is made.

CFGRANDOMFILE
Attributes: ID=integer (mandatory), DIR=string (optional)

Syntax: <CFGRANDOMFILE ID=”9” DIR=”filepath”>..format ..</CFGRANDOMFILE>

CFGRANDOMFILE element provides a mechanism to configure the format of the RANDOMFILE element. The ID attribute determines which RANDOMFILE to apply the format. The DIR element allows the user to configure a path and this path is checked before the file is created. The format text allows the user to format the filename using various date and time options to general a reasonably random filename. There is no validation on the filename created and the user must ensure a valid format structure.

Format variables: c, d, dd, ddd, dddd, m, mm, mmm, mmmm, yy, yyyy, h, hh, n, nn, s, ss, z, zz, t, tt, a/p, am/pm, quoted text.

c : output filename using the current short date global format
d, dd, ddd, dddd : output filename using date day number (d, dd) or day name
m, mm, mmm, mmmm: output filename using date month number (m, mm) or month name.
yy or yyyy: output filename using date year in 2 or 4 digits.
t, tt : output filename using time in short or long format
h, hh : output filename using time hour.
n, nn : output filename using time minutes.
s, ss : output filename using time seconds.
z, zzz : output filename using time milli-seconds.
a/p, am/pm : output filename using am/pm symbols.

Quoted text: output filename using literal text shown within the quotes (single or double quotes).

Examples: mmmddhhnnss”script.sql” = apr26113001script.sql
mmddyyzzz”.sql” = 042605034.sql
“helloworld”zzz”.sql” = helloworld034.sql

RANDOMFILE
Attributes: none

Syntax: <RANDOMFILE>..ID ..</RANDOMFILE>

RANDOMFILE element provides a way of creating a semi-random file name based on the current date and time using the formatting provided through CFGRANDOMFILE element. The ID indicates the filename buffer to store the name with possible values of 0 through to 9. The first time the RANDOMFILE element for a specific ID is detected a new file name will be created. If a CFGRANDOMFILE element does not exist for the ID then for default file name is mmddyyhhnnss”script.sql” (see above definition) for the current date and time.

Each subsequent time the RANDOMFILE element for a specific ID is used the stored filename will be re-used. This allows the user to create a date based filename preventing past scripts from being overwritten and also allowing the script to them email the file back to the user. There are no checks on files name or ID values so if a user specifies a new ID for an email file attachment then a new random file name will be created and no file found to be sent. There is also no guarantee that the file name is random and if the file name format is insufficiently random (for example only the year was used) then on subsequent runs the output will be overwritten.

Example:
.. <STRUCTURE><FILENAME><RANDOMFILE>1</RANDOMFILE></FILENAME> …
.. <EMAILS><EITEM> ..
<FILES><FITEM><NAME><RANDOM>1</RANDOM></NAME>..