Contents

Main Form 2
- Projects 2
- Template Group 2
- Main Form 3

Connections 4
- Data Connections 4
- Smtp Connections 5
- Ado. Net Connections 5
- Flat File Connections 6

Flat File Formats 7

Parameters 8

Packages 9
- Data Flow Package 9
- Execute SQL Package 11
- Code Generator – Fact Table Partition 13
- Code Generator – Dim Table Merge 14
  - Column Naming Scheme 15
- Code Generator – Fact Table Merge 15
- Code Generator – Fact Table Switch 16
- Foreach Data Flow Package 17
- Execute Script Package 18
- Execute Process Package 19
- Sequence Number 20

Environments 21
- Environments Parameter 22

Utilities 22
- Select Statement Generator 22
- Flat File Header Utility 23
- Flat File Column Alias 23

Build 24
- Package Protection Level 24
- VS Deploy 25
- Package Query Report 27
- Build Logs 27

Main Form

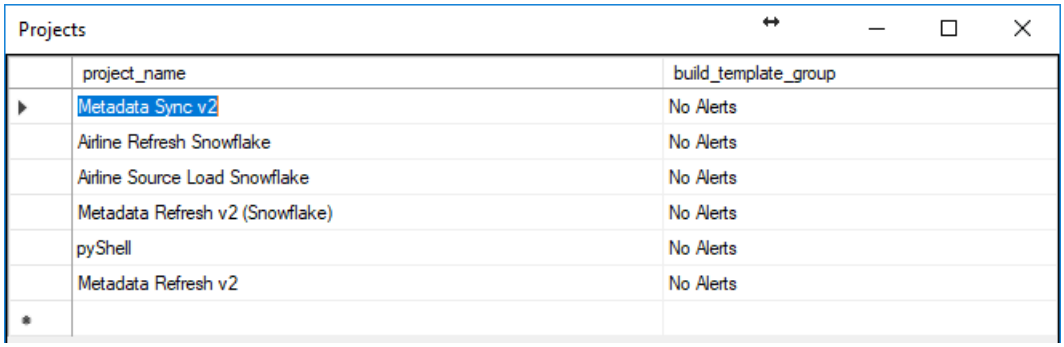
The ‘Open Projects’ button on main form is used to:

- Create a new project
- Change an existing project
- Delete an existing project

A BimlSnap ‘project’ is the overall container which holds project resources such as:

- Connections
- Parameters
- Packages

Projects



To create a new project these values should be added:

- Project Name – optional value, but recommendation is to use the same name as the associated Visual Studio ‘Project Name’.
- Template Group - available values, see section “Template Group”.

Based on the ‘Template Group’ selection, certain ‘Connection and Parameter’ resources will be added automatically to the BimlSnap project.

Template Group

There are three ‘Template Group’ options available:

- Standard
- No Alerts
- No Framework

The ‘Standard’ template provides the following Framework features:

- Project restart ability
- Package row counts

- Package runtimes
- Email alerts
- Package error logging

The 'No Alerts' option provides all the 'Standard' template features except for 'Email alerts'.

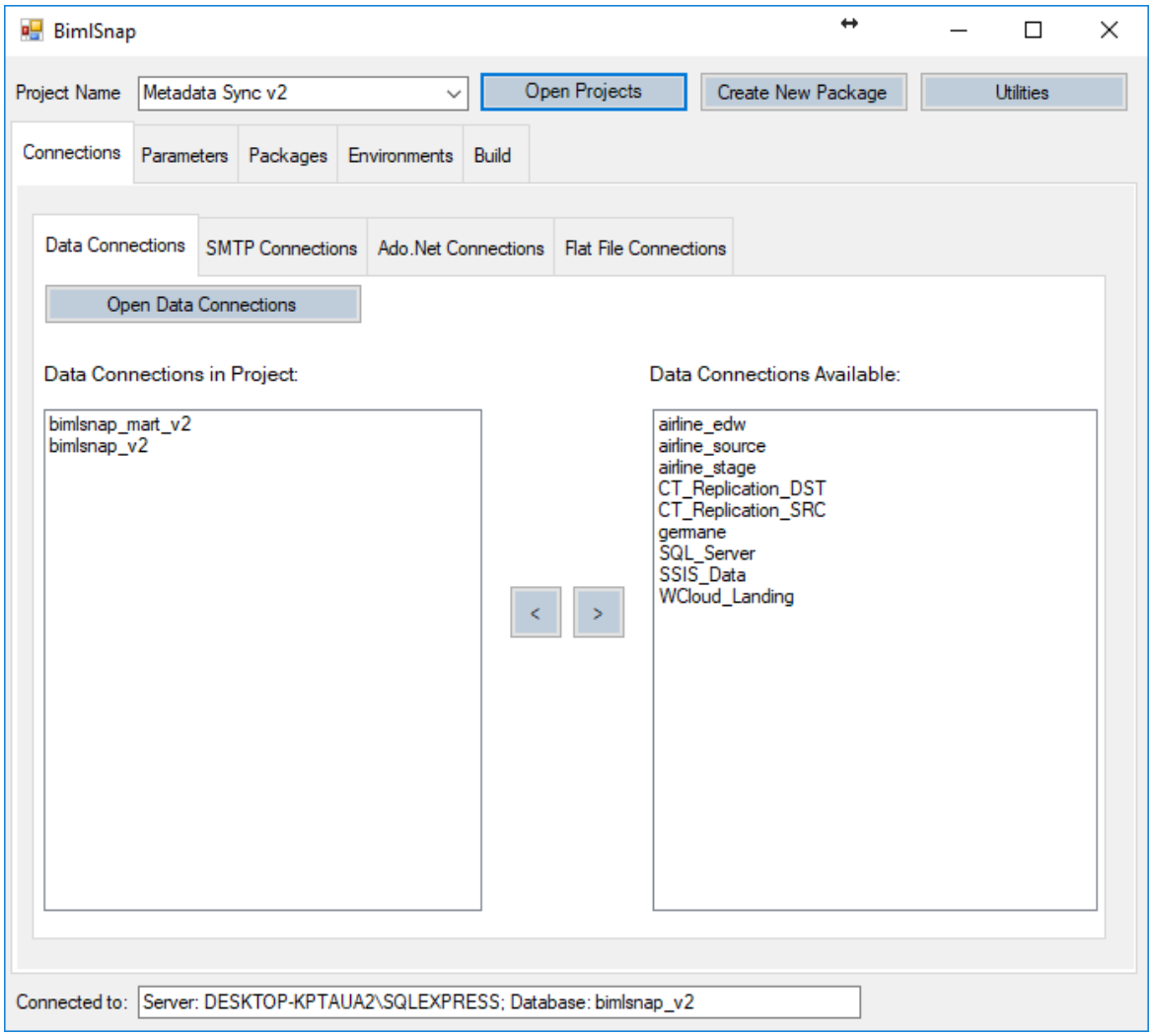
Both the 'Standard' and 'No Alerts' template options require the 'SSIS_Data' database to be available. This SQL Server database (DDL) can be downloaded from: https://www.bimlsnap.com/

The 'No Framework' option is a minimalist template, can be used when none of the above mentioned features are not required.

These Template options can be changed at any time.

Main Form

On application start, the main form is shown.



On the top left corner dropdown with all created project for the user will appear, if projects are not created for the user the dropdown will be empty. On selection change event, resource for selected project will be populated. These resources can be configured manually by clicking on the tabs. Also resources can be moved from left to right and vice versa dependent on the need if resource needs to be in project or not.

On the bottom of the main form is the bar to show used server and database for application. This field is not editable and it's shown as reminder to a user.

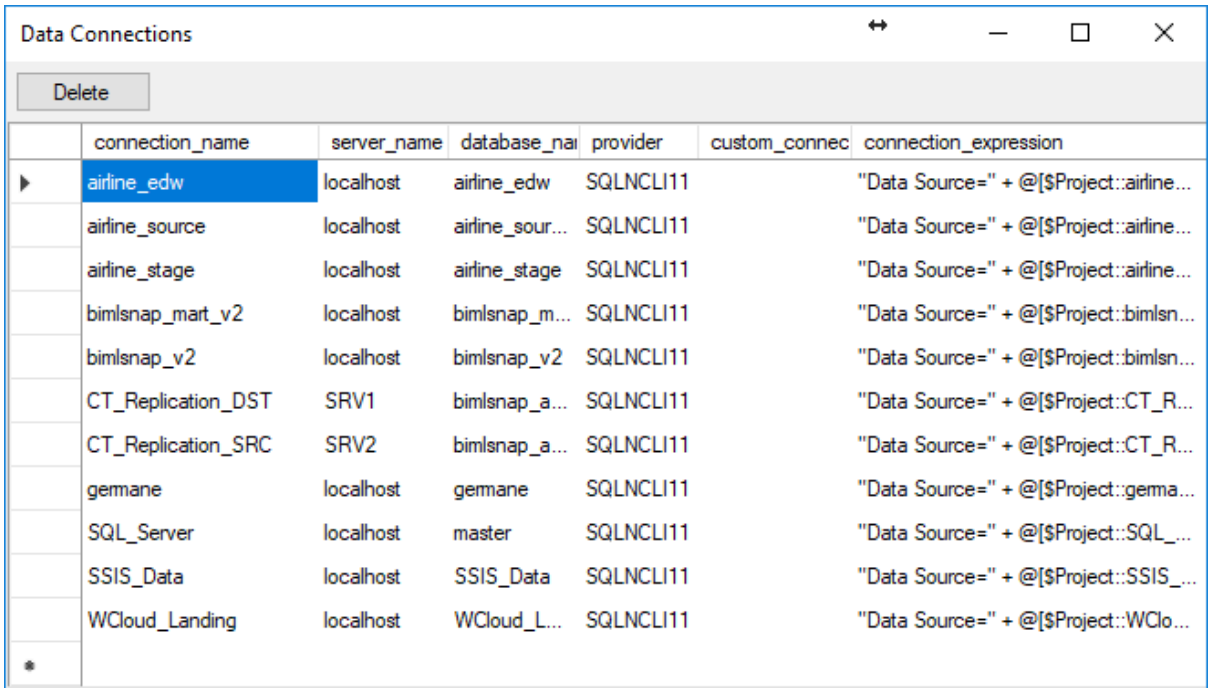
Connections

Besides the data that was automatically added, User can manually add data. For Connections, there are four types: Data (OleDB), SMTP, Ado.Net and Flat File Connections.

Data Connections

Clicking on button "Open Data Connections" OleDB connection can be manipulated. Data Connections form provide options:

- Add a new connection
- Edit an existing connection
- Delete and existing connection



connection_name	server_name	database_nai	provider	custom_connec	connection_expression
airline_edw	localhost	airline_edw	SQLNCLI11		"Data Source=" + @[$Project::airline...
airline_source	localhost	airline_sour...	SQLNCLI11		"Data Source=" + @[$Project::airline...
airline_stage	localhost	airline_stage	SQLNCLI11		"Data Source=" + @[$Project::airline...
bimlsnap_mart_v2	localhost	bimlsnap_m...	SQLNCLI11		"Data Source=" + @[$Project::bimlsn...
bimlsnap_v2	localhost	bimlsnap_v2	SQLNCLI11		"Data Source=" + @[$Project::bimlsn...
CT_Replication_DST	SRV1	bimlsnap_a...	SQLNCLI11		"Data Source=" + @[$Project::CT_R...
CT_Replication_SRC	SRV2	bimlsnap_a...	SQLNCLI11		"Data Source=" + @[$Project::CT_R...
germane	localhost	germane	SQLNCLI11		"Data Source=" + @[$Project::germa...
SQL_Server	localhost	master	SQLNCLI11		"Data Source=" + @[$Project::SQL_...
SSIS_Data	localhost	SSIS_Data	SQLNCLI11		"Data Source=" + @[$Project::SSIS_...
WCloud_Landing	localhost	WCloud_L...	SQLNCLI11		"Data Source=" + @[$Project::WClo...

To create a new 'data connection' based on 'Windows Integrated' security the following needs to be entered:

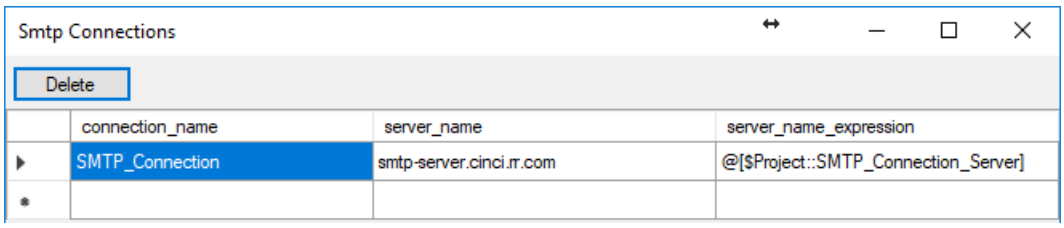
- ‘Connection Name’ – optional value. We recommend making the ‘Connection Name’ the same as the ‘Database Name’. The actual connection will later be driven by parameters (which are automatically created) to configure the connection with a run-time expression
- ‘Server Name’ – actual server name used for this connection
- ‘Database Name’ – existing database name for selected server.
- “Custom Connect String” – connection string needed to use “SQL Standard” security, or to have greater control over the exact connect string. Note that this option will override the ‘Server Name’, and ‘Database Name’ text boxes. The “Custom Connect String” can also be substituted by a parameter expression at run-time.

If connection is not needed anymore it can be deleted by selecting connection and clicking on “Delete” button. Please note that connection can be used in multiple project and deleting it can cause error on the other projects.

Smtp Connections

‘Open SMTP Connections’ button click will open “Smtp Connections” form. This form is used to:

- Create a new SMTP connection
- Change an existing SMTP connection
- Delete an existing SMTP connection



To create a new ‘SMTP connection’ the following values needs to be entered:

- ‘Connection Name’ – optional value
- SMTP ‘Server Name’ – actual SMTP server name

The actual SMTP connection will later be driven by parameters (which are automatically created) to configure the ‘Send mail’ connection with a run-time expression.

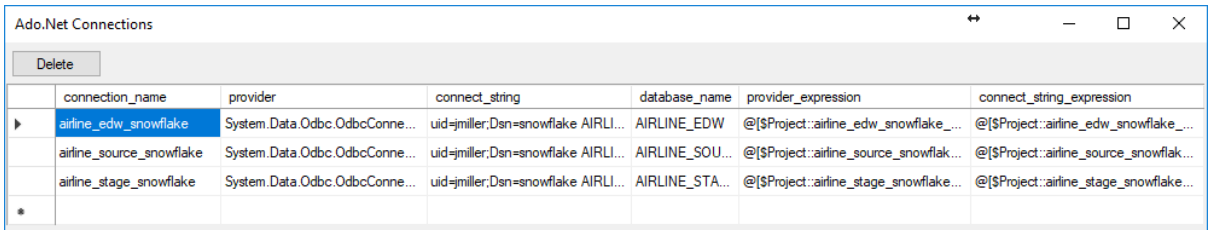
To edit or delete an existing ‘SMTP connection’, select the ‘connection’ row on the lower part of the screen. Once selected, you can change any ‘SMTP connection’, or to completely remove, click ‘Delete’. All SMTP connections are added global connections and deleting already used connection in other project will cause exceptions.

Ado. Net Connections

An Ado. Net connection enables a package to access data sources by using .Net provider. This connection is typically used to access data sources such as Microsoft SQL Server, and also data sources exposed through OLE DB. When you add an ADO.NET connection to project application generate a SSIS package as biml tags into .Biml file.

“Open Ado.Net Connections” button click will open “Ado.Net Connections” form. This form is used to:

- Create a new Ado.Net connection
- Change an existing Ado.Net connection
- Delete an existing Ado.Net connection



connection_name	provider	connect_string	database_name	provider_expression	connect_string_expression
airline_edw_snowflake	System.Data.Odbc.OdbcConne...	uid=jmiller;Dsn=snowflake AIRLI...	AIRLINE_EDW	@[$Project::airline_edw_snowflake_...	@[$Project::airline_edw_snowflake_...
airline_source_snowflake	System.Data.Odbc.OdbcConne...	uid=jmiller;Dsn=snowflake AIRLI...	AIRLINE_SOU...	@[$Project::airline_source_snowflak...	@[$Project::airline_source_snowflak...
airline_stage_snowflake	System.Data.Odbc.OdbcConne...	uid=jmiller;Dsn=snowflake AIRLI...	AIRLINE_STA...	@[$Project::airline_stage_snowflake...	@[$Project::airline_stage_snowflake...

To create a new connection the following values needs to be added:

- Connection name – optional value
- Provider name – actual provider used later in SSIS project
- Connection string
- Database name
- Provider expression and connection string expression values are automatically added and calculated for entered connection

If connection is not needed anymore it can be deleted by selecting wanted connection and clicking on “Delete” button. Please note that all connection are global connections, not project specific connections. Deleting already used connection can cause the issue in other part of a system.

Flat File Connections

The ‘Flat File Connection’ is used to access flat files on a computer. The Flat File Connection is defined by connection name, file path and that are parameters that user need to input. The file path expression and file format expression are generated by application and are used in other parts of application such as are packages. When a new Package is being created user has an option to choose the flat file connection as a destination to export data from table to file.

The ‘Open Flat File Connections’ button is used to open Flat File Connections form. This form is used to:

- Create a new ‘flat file connection’
- Change an existing ‘flat file connection’
- Delete an existing ‘flat file connection’

Flat Connections

Delete

connection_name	file_path	file_format	file_path_expression	file_format_expression
airlines	C:\snowflake\snow_csv\airlines.csv	airlines	@[$Project::airlines_FilePath]	@[$Project::airlines_FileFormat]
flight	C:\snowflake\snow_csv\flight.csv	flight	@[$Project::flight_FilePath]	@[$Project::flight_FileFormat]
performance	C:\snowflake\snow_csv\performance....	performance	@[$Project::performance_FilePath]	@[$Project::performance_FileFormat]

To create a new flat file connection enter the following values:

- 'Connection Name',
- 'File Path'
- 'File Format' – existing flat file format
- 'File Path Expression' will be automatically generated as well as 'File Format Expression'. These two parameters can be used to configure packages that are used file under this connection.

To delete a flat file connection, you need to select row and click on the button 'Delete'.

Flat File Formats

The Flat File Formats are used to define the format of the flat file columns. It is used to map flat file columns with the table in the database as well as creating the new table in the database.

Flat File connection for proper functioning needs to have valid flat file format added. To do that click on "Flat File Format" button. This will open Flat File Format form.

The following form is used to:

- Add a new flat file format
- Update an existing flat file format
- Delete and existing flat file format

Flat File Formats

Delete

file_format	code_page	column_delimiter	row_delimiter	text_qualifer	ColumnNamesInFirstDataRow	IsUnicode	metadata_server	metadata_database	metadata_schema	metadata_table
airlines	1252	Comma	CRLF	_x0022_	false	false	PC3	airline_source	dbo	airlines
flight	1252	Comma	CRLF	_x0022_	true	false	PC3	airline_source	dbo	flight
performance	1252	Comma	CRLF	_x0022_	false	false	PC3	airline_source	dbo	performance

In order to create new flat file format, enter the following values:

- Flat File format name
- Code page – default value is 1252
- Column delimiter – default column delimiter is "comma"
- Row delimiter – default values is "CRLF"
- Text qualifier –default values is "_x0022_"
- Column names in first data row – default values is true
- Is Unicode value – default is true
- Metadata server
- Metadata database
- Metadata schema
- Metadata table

To delete a flat file format, you need to select row and click on the button ‘Delete’.

Parameters

The ‘Open Parameters’ button is used to open parameters form. This form is used to:

- Create a new parameter
- Change an existing parameter
- Delete an existing parameter

Parameters

Delete

parameter_name	parameter_dataty	parameter_value	parameter_reference
Admin_Error_Sendmail_To	String	email@here.com	@[$Project::Admin_Error_Sendmail_To]
Admin_Sendmail_From	String	email@here.com	@[$Project::Admin_Sendmail_From]
Admin_Sendmail_To	String	email@here.com	@[$Project::Admin_Sendmail_To]
airline_edw_Database	String	airline_edw	@[$Project::airline_edw_Database]
airline_edw_Provider	String	SQLNCLI11	@[$Project::airline_edw_Provider]
airline_edw_Server	String	localhost	@[$Project::airline_edw_Server]
airline_edw_snowflake_Connec...	String	uid= ;Dsn=...	@[$Project::airline_edw_snowflake_ConnectString]
airline_edw_snowflake_Database	String	AIRLINE_EDW	@[$Project::airline_edw_snowflake_Database]
airline_edw_snowflake_Provider	String	System.Data.O...	@[$Project::airline_edw_snowflake_Provider]
airline_source_Database	String	airline_source	@[$Project::airline_source_Database]
airline_source_Provider	String	SQLNCLI11	@[$Project::airline_source_Provider]
airline_source_Server	String	localhost	@[$Project::airline_source_Server]
airline_source_snowflake_Conn...	String	uid= ;Dsn=...	@[$Project::airline_source_snowflake_ConnectS...
airline_source_snowflake_Data...	String	AIRLINE_SOU...	@[$Project::airline_source_snowflake_Database]
airline_source_snowflake_Provi...	String	System.Data.O...	@[$Project::airline_source_snowflake_Provider]
airline_stage_Database	String	airline_stage	@[$Project::airline_stage_Database]
airline_stage_Provider	String	SQLNCLI11	@[$Project::airline_stage_Provider]
airline_stage_Server	String	localhost	@[$Project::airline_stage_Server]
airline_stage_snowflake_Conne...	String	uid= ;Dsn=...	@[$Project::airline_stage_snowflake_ConnectStri...

To create a new parameter enter the following values:

- Parameter name
- Parameter data type value – predefined values
- Parameter value – dependent on parameter data type value
- Parameter reference will be calculated automatically

Based on the type of created Connection, Parameters will be automatically created with predefined values.

To edit or delete an existing ‘parameter’, select the ‘parameter’ row on the lower part of the screen. Once selected, you can change any ‘parameter’ attribute, or to completely remove, click ‘Delete’.

Packages

Packages is the part where all the magic happens.

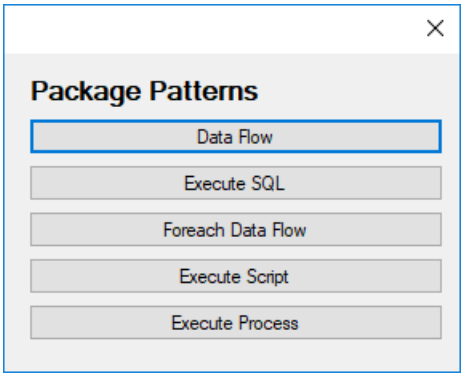
A Package is another Biml root type that corresponds to an SSIS package. They can contain one or more Package elements. They have Tasks collection: Execute SQL and Data Flow.

The Execute SQL task has a Name attribute and its connection name attribute indicates where the task's SQL statement will run.

The Data Flow task is surrounded by a C# foreach loop, that is enclosed by the special <# #> tags which mark the beginning and ending of BimlScript code nuggets.

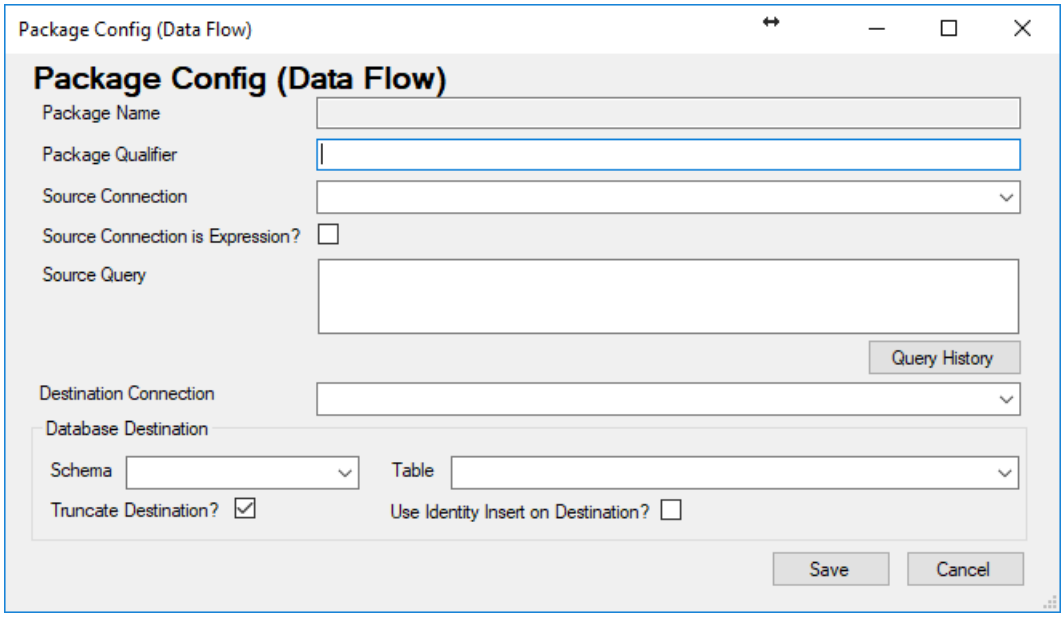
For each table, a Dataflow task is added to the package, where the Dataflow's name includes the table's name. Each Dataflow task contains a *Transformations* collection and can contain data flow source and destination definitions for the task. Like the *ExecuteSQL* task, the *OleDbSource* transformation uses a *DirectInput* element to specify what data will be retrieved.

A User can create several different package patterns: Data Flow, Execute SQL, Foreach Data Flow, Execute Script and Execute Process. These patterns can be found by clicking on button "Create New Package".



Data Flow Package

Selecting "Data Flow" package pattern the following form will be visible.

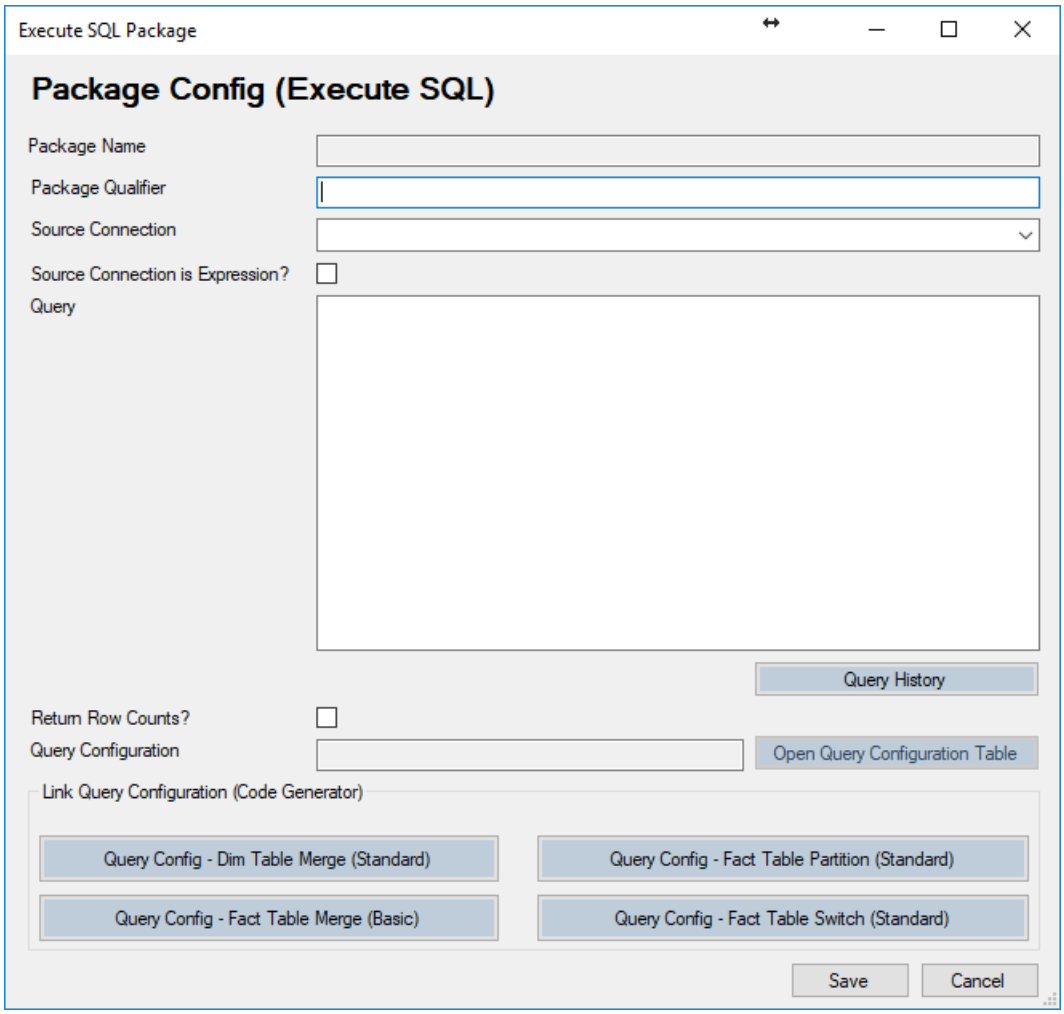


In order to create new Data Flow package the following values needs to be entered:

- Package qualifier – unique name used to construct package name
- Source connection – dropdown with all available connections for selected project
- Is source connection expression or not – if checked value is true
- Source query – valid SQL query
- Destination connection – dropdown with all available connections for selected project
- Destination schema name - values can be selected from dropdown or manually added to the control
- Destination table name - values can be selected from dropdown or manually added to the control
- Truncate destination – default value is true
- Use identity insert on destination
- Package name – value is constructed using pattern type, “Data Flow” in this case, package qualifier value and destination table value. These values are separated by dash.

Execute SQL Package

Selecting “Execute SQL” package pattern the following form will be visible.



The Execute SQL Package runs SQL statements or stored procedures. It can contain either a single SQL statement or multiple SQL statements that run sequentially.

The Execute SQL task is one of the handier components in SQL Server Integration Services (SSIS) because it lets you run Transact-SQL statements from within your control flow. When using an ‘ELT’ approach to data integration, the Execute SQL Package pattern will generally follow the Data Flows ‘EL’ (ie., Extract and Loading) to perform the ‘T’ (data Transformations).

The ‘Execute SQL’ button is used to:

- Create a new ‘Execute SQL’ Package
- Change an existing ‘Execute SQL’ Package
- Delete an existing ‘Execute SQL’ Package

- Clone and existing ‘Execute SQL’ Package

When creating the **Execute SQL** Package you will need to provide the Package Qualifier and OleDb Connection. For the Query field you can use SQL statement(s) or you can use Code Generator which will auto-populate the Query field based on built-in T-SQL code generators.

To use the Code Generator, click the ‘Query Config – Fact Table Partition’ button.

The Execute SQL Package runs SQL statements or stored procedures. It can contain either a single SQL statement or multiple SQL statements that run sequentially. For the Query field you can use SQL statement(s) or you can use Code Generators which will auto-populate the Query field. To use the Code Generators, click one of the:

- ‘Query Config – Dim Table Merge (Standard)’
- ‘Query Config – Fact Table Partition (Standard)’
- ‘Query Config – Fact Table Merge (Basic)’
- ‘Query Config – Fact Table Switch (Standard)’

In order to create new Execute SQL package the following values needs to be entered:

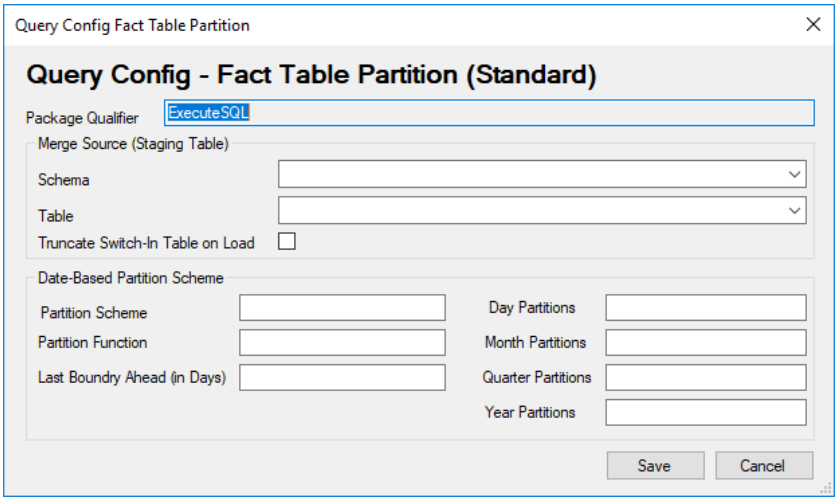
- Package qualifier – unique name used to construct package name
- Source connection – dropdown with all available connections for selected project
- Is source connection expression or not – if checked value is true
- Query – valid SQL query can be added manually or by code generators
- Return row counts or not.
- Package name – value is constructed using pattern type, “Execute SQL” in this case and package qualifier value separated by dash.

This package pattern has option for “Link Query Configuration”. These options will use selected code generator in order to create query needed for a package.

In order to change the Query, just click the ‘Open Query Configuration Table’ and change any data that is needed, and click ‘Save’. After that the Query field will be updated with the new Query that was just generated.

Code Generator – Fact Table Partition

Choosing option "**Query Config - Fact Table Partition"** new form will be displayed.



After the User saves the data that was filled in, he will be redirected to the previous form for Execute SQL Package and the generated query will appear in the query field.

Very large tables can be difficult to manage because of their size and the amount of time it takes to maintain. BimlSnap includes 'code generators' which can be used to 'partition' these large tables into manageable segments.

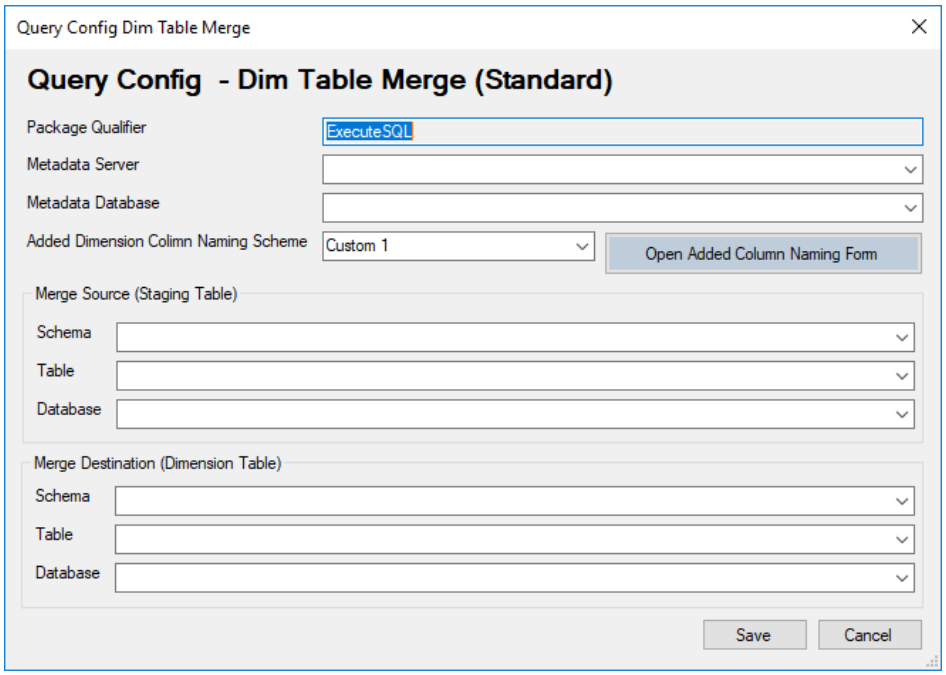
Before using the table partitioning code generators, you must first manually create the table's associated partition function and scheme. For more information on this task, see: https://docs.microsoft.com/en-us/sql/relational-databases/partitions/partitioned-tables-and-indexes . In addition, for the BimlSnap code generators to work, the partioning column must be defined as a 'date' datatype, and all indexes for the table need to be 'partitioned aligned'.

First, you will need to specify the Package Qualifier and select the OleDb Connection, then click the 'Query Config – Fact Table Partition' button which will lead you to a new form. Then you will have to provide the data for each empty field (all fields are required). Select the Partitioned Source Table Schema and Table, check if you want to truncate Switch-In Table on Load (recommended), and populate the Date-Based Partition scheme. After that click 'Save' and the for the Query will be generated during the next 'build' (note that for this to work properly, your BimlSnap metadata must be up-to-date).

After saving, you will be redirected to the main page, where the newly created Package should appear. To make changes, double click the package name, and in the form that opens, you can 'Delete', 'Edit' or 'Clone' the package definition.

Code Generator – Dim Table Merge

By choosing option **Query Config - Dim Table Merge** user is redirected to a new form for that query generator.



The generated Merge statement conforms to the Kimball 'Type 1' and 'Type 2' dimension model. Use the SnapMart metadata application to specify the dimension type to be used for each attribute The MERGE statement is used when you want to apply changes that include inserts and updates of data based on the contents of the associated 'staging' table.

In order to do that by using the Code generator 'Query Config – Dim Table Merge', you will need to provide the Metadata Server and Metadata Database (because desktop application has an option for multiple servers), that is database parameter for the specific OleDb Connection. After that you will need to provide the Merge Source (Staging Table) Schema, Table and Database Parameter and Merge Destination (Dimension Table) Schema, Table and Database Parameter, and the Added Dimension Column Naming Schema.

The Added Dimension Column Naming Schema can be chosen from the dropdown as predefined 'Standard' or can be added by clicking 'Open Added Column Naming Form' button.

After that click 'Save' button and the Query expression field will be populated with the generated expression.

In order to change the Query expression, just click the 'Open Query Configuration Table' and change any data that is needed, and click 'Save'. After that the Query expression field will be updated with the new Query expression that was just generated.

Column Naming Scheme

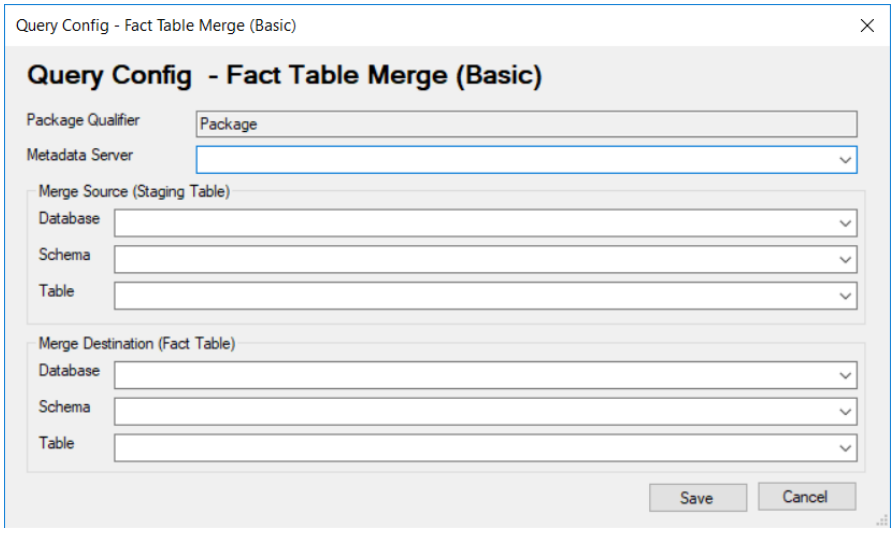
The Dimension Column Naming Scheme is used when generating the 'Query Config – Dim Table Merge' Code generator. This table allows you to customize column names used in your MERGE 'destination' table:

Results | Messages

	added_dim_column_names_id	added_dim_column_names_tag	row_is_current	row_effective_date	row_expiration_date	row_insert_date	row_update_date
1	1	Standard	row_is_current	row_effective_date	row_expiration_date	row_insert_date	row_update_date
2	2	Custom 1	ETL_Is_Current	ETL_Effective_Date	ETL_Expiration_Date	ETL_Insert_Date	ETL_Update_Date
3	3	Standard UC	ROW_IS_CURRENT	ROW_EFFECTIVE_DATE	ROW_EXPIRATION_DATE	ROW_INSERT_DATE	ROW_UPDATE_DATE

In order to save a new Dimension Column Naming Scheme, provide the needed data and click 'Save'. The saved Dimension Column Naming Scheme will appear in the dropdown list on 'Query Config – Dim Table Merge' form and can be chosen for the same Code generator.

Code Generator – Fact Table Merge



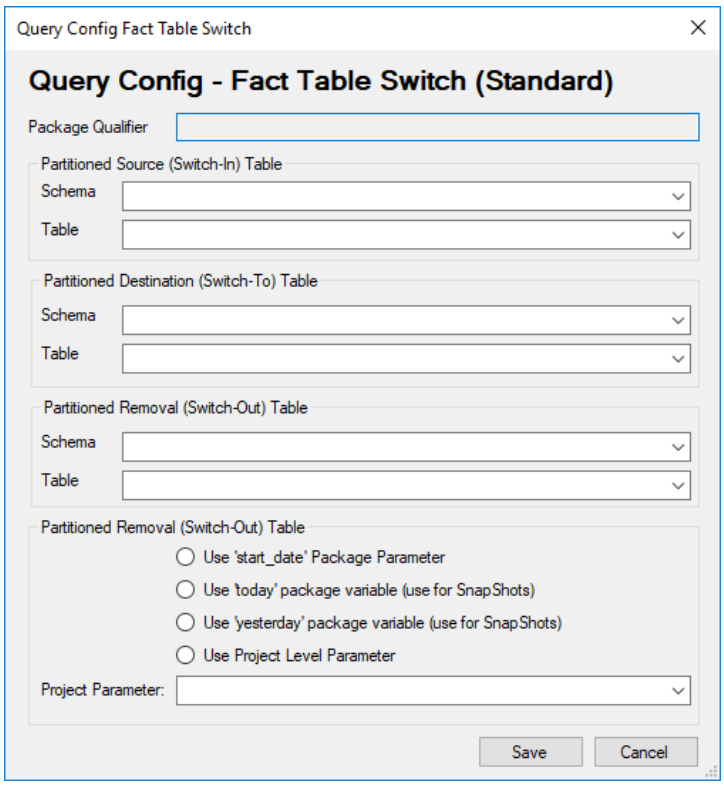
The Fact Table Merge Code generator is almost the same as a Dim Table Merge Code Generator, but instead of dimensional table it uses fact table.

The generated Merge statement conforms to the Kimball 'Type 1' and 'Type 2' dimension model. Use the SnapMart metadata application to specify the dimension type to be used for each attribute The MERGE statement is used when you want to apply changes that include inserts and updates of data based on the contents of the associated 'staging' table.

In order to do that by using the Code generator 'Query Config – Fact Table Merge (Basic)', you will need to provide the Metadata Server, that is database parameter for the specific OleDb Connection. After that you will need to provide the Merge Source (Staging Table) Schema, Table and Database Parameter and Merge Destination (Fact Table) Schema, Table and Database Parameter.

After that click 'Save' button and the Query expression field will be populated with the generated MERGE Statement.

Code Generator – Fact Table Switch



One of the benefits of the table partitioning is that you can speed up loading the data by using partition switching. Partition switching moves entire partition between tables almost instantly.

To switch the partitions between tables, you need to follow the following requirements:

- The source and destination partitions must have identical columns, indexes and use the same partition column, scheme and function
- The source and destination partitions must exist on the same filegroup
- The 'switch-to' destination partition must be empty

In order to generate the Source Expression with the 'Query Config – Fact Table Switch' Code Generator, click the 'Query Config – Fact Table Switch' button. Provide the data for Partitioned Source (Switch-In) Table, select or type in the Table Schema and Table. Select or type in the

Table Schema and Table for Partitioned Destination (Switch-To) Table and repeat the same for the Partitioned Removal (Switch-Out) Table.

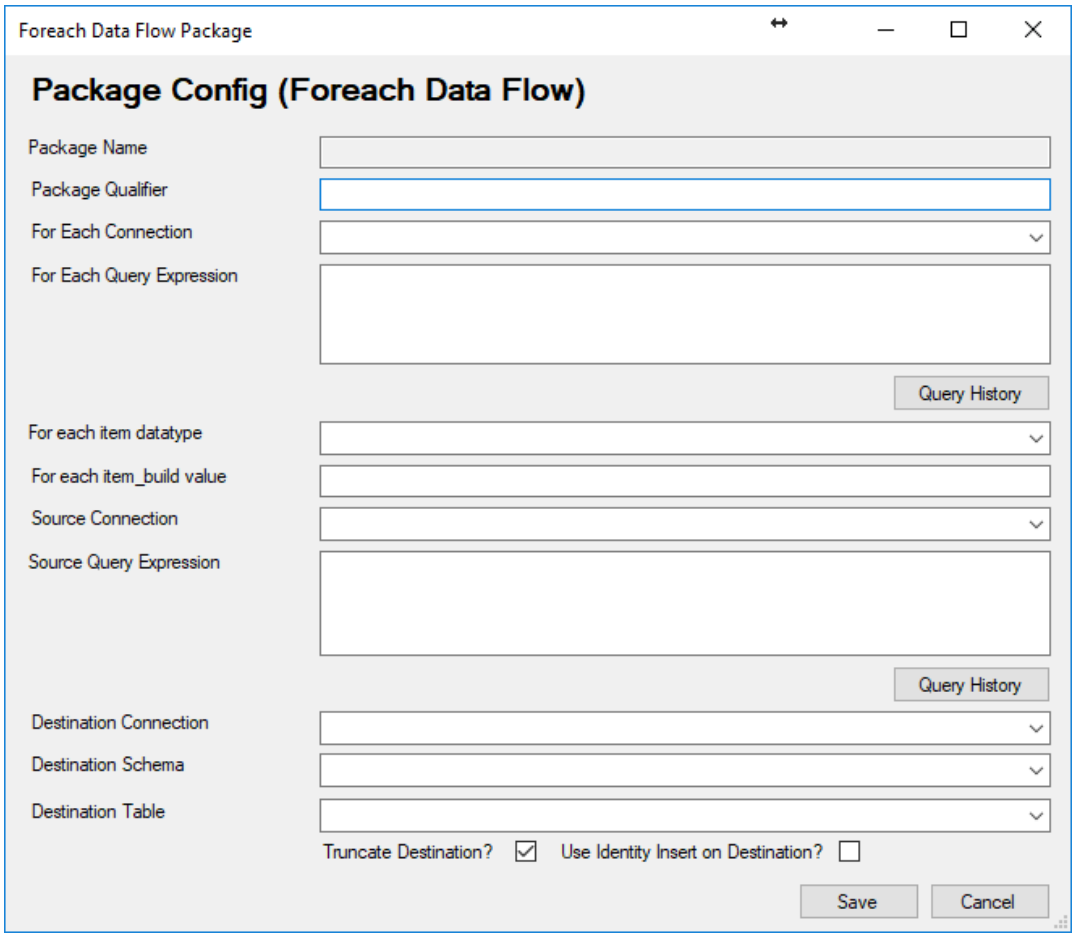
Then select one of the options for the Switch Option 'Start' Date. Note that when you choose the 'use Project Level Parameter' option, you will need to select a Project Parameter. Otherwise, it's not needed.

When you finish, click the 'Save' button and the Query field will be populated with the generated expression.

In order to change the Query expression, just click the 'Open Query Configuration Table' and change any data that is needed, and click 'Save'. After that the Query expression field will be updated with the new Query expression that was just generated.

In the event you manually entered a query, you can also click on the 'Query History' button to retrieve any Query Expression that was previously used.

Foreach Data Flow Package



The 'Foreach Data Flow' button is used to:

- Create a new 'Foreach Data Flow' Package
- Change an existing 'Foreach Data Flow' Package
- Delete an existing 'Foreach Data Flow' Package
- Clone an existing 'Foreach Data Flow' Package

The Foreach Loop container provides a query driven looping structure that iterates through a collection of objects or data values. At run-time, this component results in a 'data flow' (also query

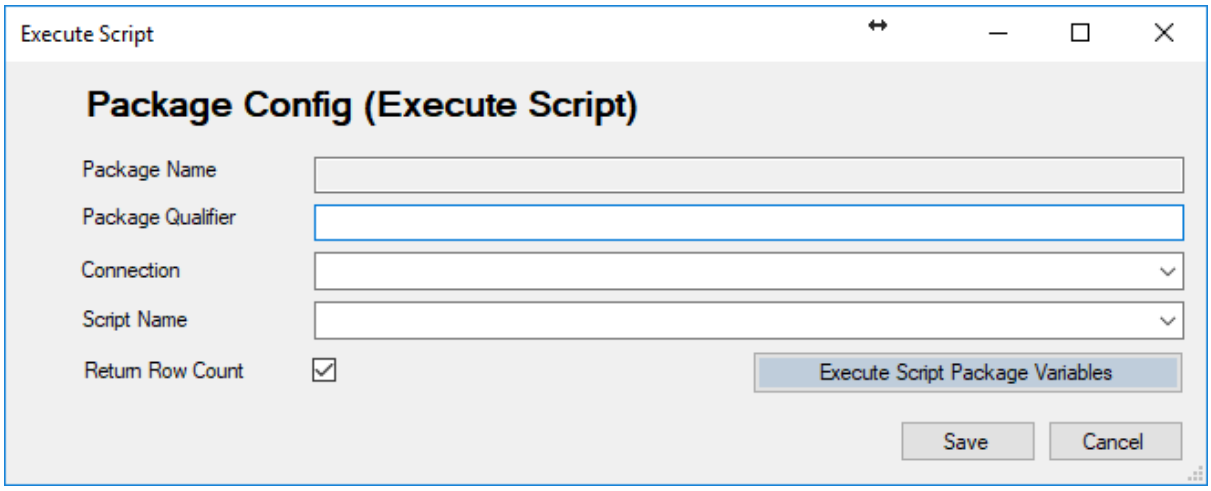
driven) for each item found. The Foreach Loop pattern is a good fit for loading multiple source files or tables into a single destination.

You will need to specify the Package Qualifier, choose the Source and Destination OleDb Connection, For Each Query Expression, Datatype, Build Value, Source Query, Destination Schema, Destination Table and to check whether you want to Truncate Destination or Use Identity Insert on Destination.

To clone, edit or delete an existing 'Foreach Data Flow' Package, double click on the package name located under 'Packages' tab.

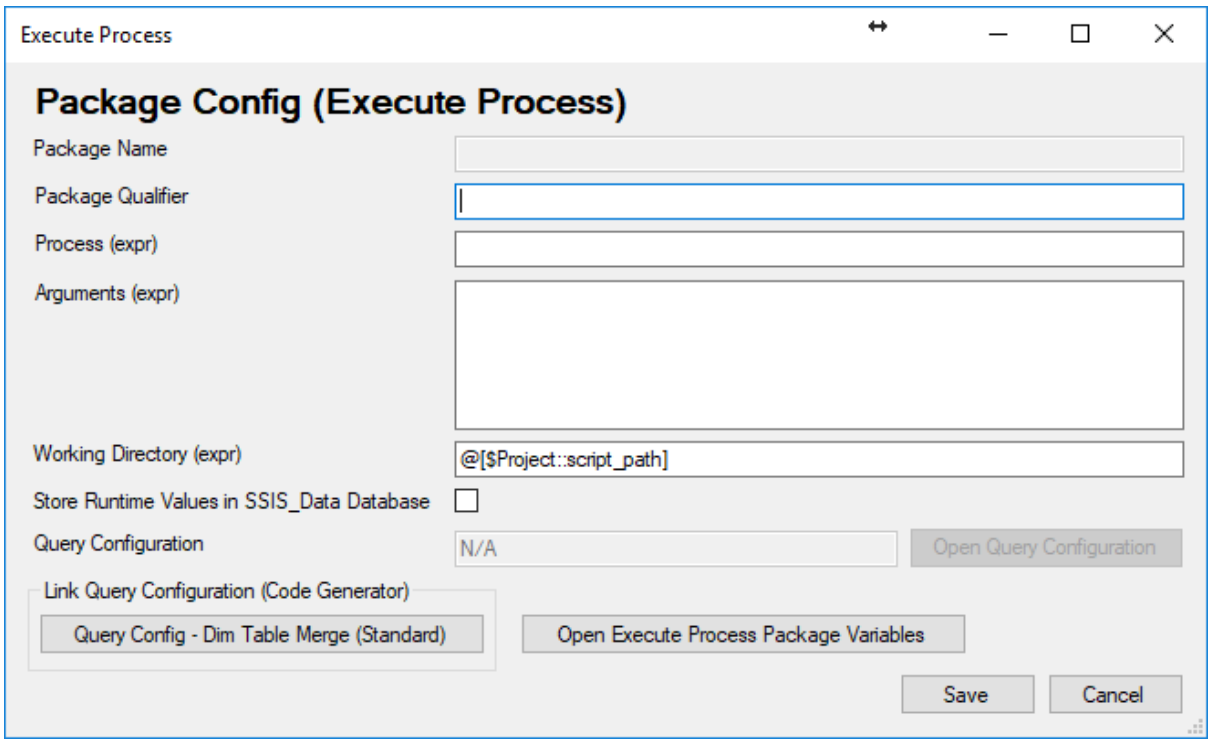
- Package Qualifier - unique name used to construct package name
- For Each Connection – connection where the table or source file is
- For Each Query Expression – the SQL statement which is going to execute on every row of table
- For each item datatype – data type of every value on which will SQL query be executed
- For each item_build value
- Source Connection – dropdown with all connections
- Source Query Expression – the SQL query to be executed on the source connection
- Destination Connection – dropdown with all connections
- Destination Schema – dropdown with all schemas for selected destination connection
- Destination table - dropdown with all tables for selected destination schema

Execute Script Package



- Package Qualifier - unique name used to construct package name
- Connection – the connection where the script need to be executed
- Script Name – name of the script
- Return Row Count – option for the returning count of rows executed in script, default value is True

Execute Process Package

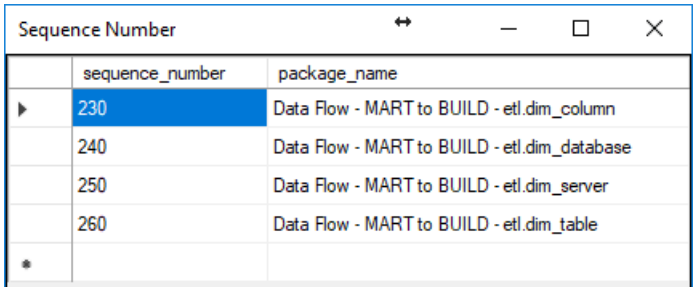


In the 'Execute Process' package user can use his own executable application to run it. I.e. user can run an application for importing data to the database or to move files from one directory to another. The Process fields are reserved for an input process name. The process name must be quoted. The working directory is the folder where the application process is saved.

- Package Name - Name of the package generated automatically
- Package Qualifier - unique name used to construct package name
- Process (expr) - Name of the process inside the working directory specified in the parameters table
- Arguments (expr) – optional arguments for the process which is going to run
- Working Directory – directory where is the process stored
- Store Runtime Values in SSIS_Data Database -

Sequence Number

On “Packages” tab the “Sequence Package Execution” button is available. This button click will open Sequence number execution form. This form is used to update sequence number.



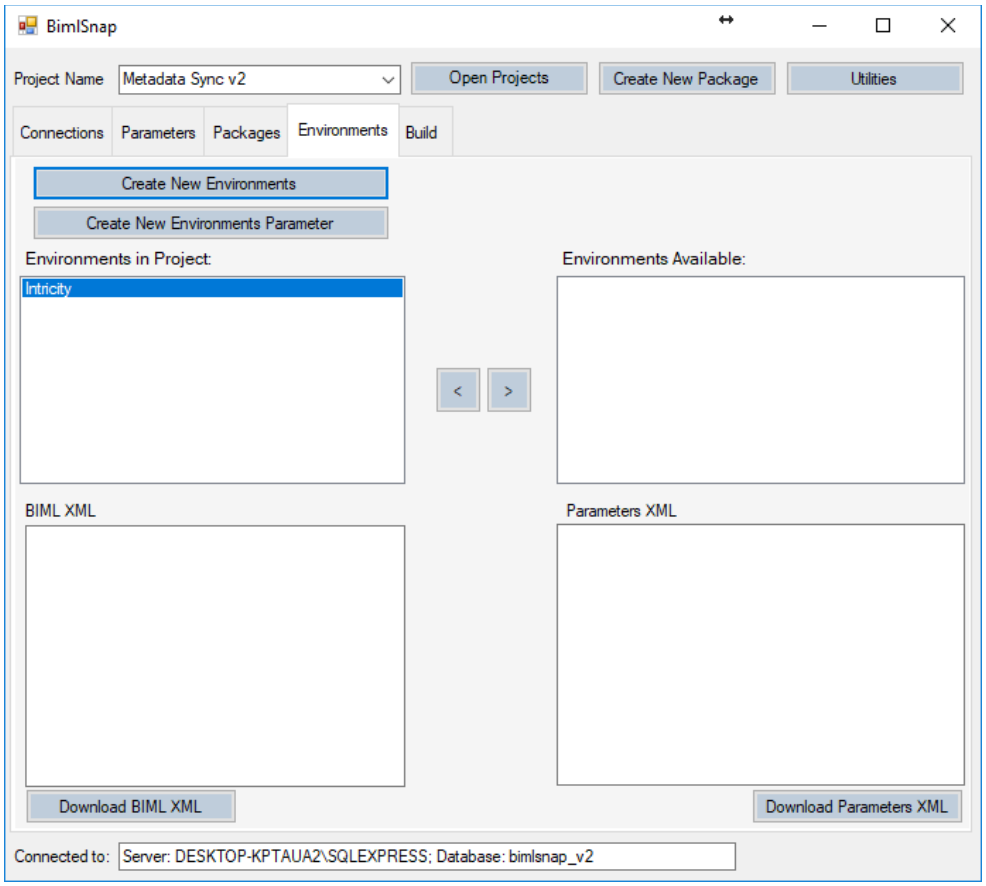
sequence_number	package_name
230	Data Flow - MART to BUILD - etl.dim_column
240	Data Flow - MART to BUILD - etl.dim_database
250	Data Flow - MART to BUILD - etl.dim_server
260	Data Flow - MART to BUILD - etl.dim_table

All packages added to the project will have sequence number zero at first. This number is used to determine package execution order. If two packages have same sequence number it will run in parallel, otherwise the packages will run sequentially (from lower number to higher).

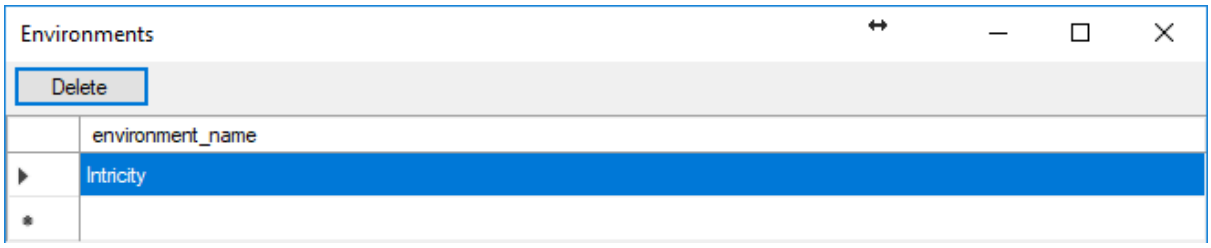
Environments

Besides Package Patterns, there is Environments tab. This part is used for environment manipulation. Using this tab user can do the following:

- Add a new environment
- Delete an existing environment
- Move environment to the project or vice versa
- Download Biml XML and Parameters XML for selected environment.



Clicking on 'Create New Environment' button new form will be shown:



User can add a new Environment by adding environment name or delete the existing environment selecting environment and clicking the button 'Delete'. After adding New Environment, it will appear on the main page under the Environment tab within the selected project.

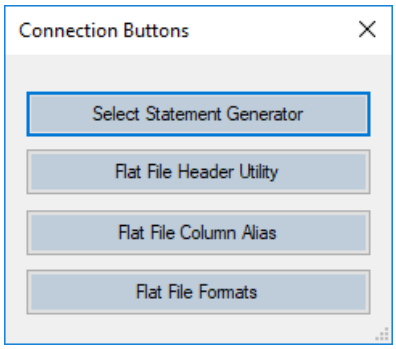
After the project is built, all environments in selected project will be built as well. Clicking on the selected environment added to the project, values for BIML XML and Parameters XML will be shown. These values can be selected using text boxes or downloaded to the local machine by clicking on the appropriate buttons.

After the document is downloaded to the machine popup message will be shown. The message will contain the location where the file is saved.

Environments Parameter

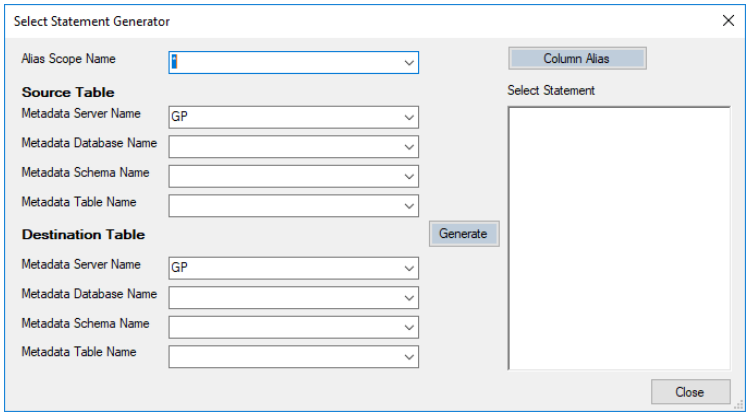
The main purpose of the Environment Parameter is that the packages can run easily in any specified environment that is created. So, what that means is that you can create one package and use parameters to create two separate environments of the same package.

Utilities



Utilities button provides some useful features for manipulating flat files or to create table.

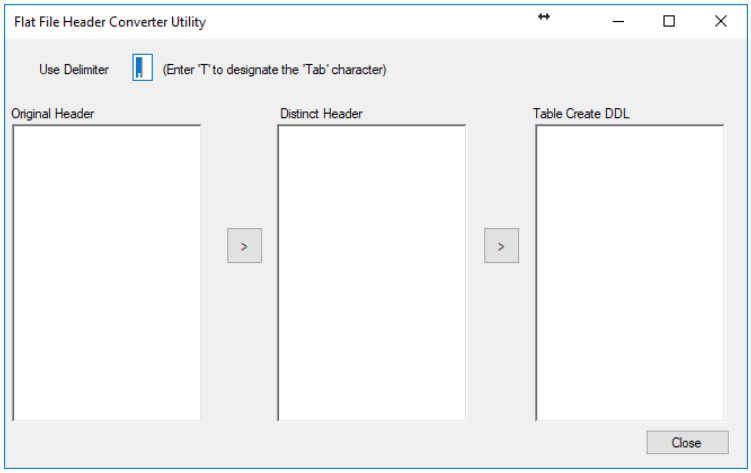
Select Statement Generator



The Select Statement Generator is designed for the 'source' tables based on a flat file. In that table, the column names are non-standard. The 'destination' table is a related table, but with standard Column names. To auto-generated a SELECT statement, we would generate a column mapping using 'AS' for the aliases. This mapping would be defined in Column Aliases.

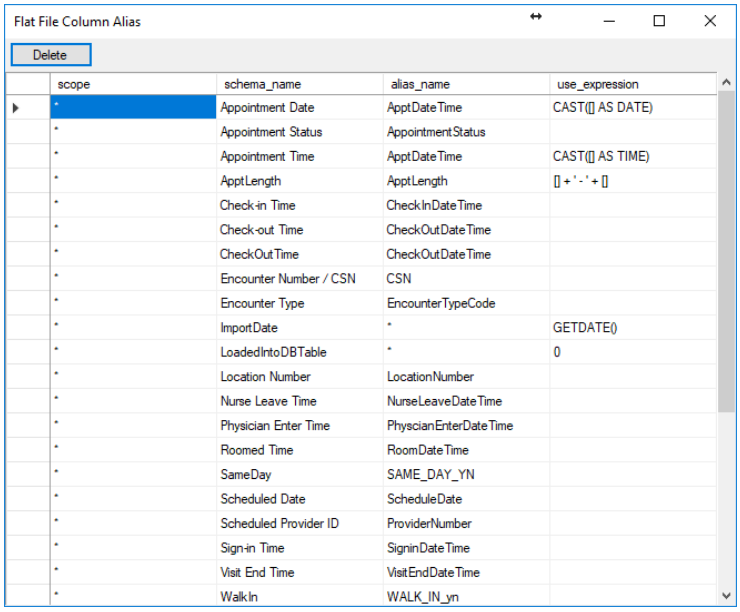
Column names that are the same in both 'source' and 'destination' are automatically mapped in the select statement. In there are no mappings found, then a NULL value is assigned.

Flat File Header Utility



Flat File Header Utility is helpful when the user need to create a new table with columns listed in the box called 'Original Header'. If there is duplicated header names (column names) they will be listed in the box called 'Distinct Header' with added underscore and number of occurrences in the list (i.e. id_1).

Flat File Column Alias

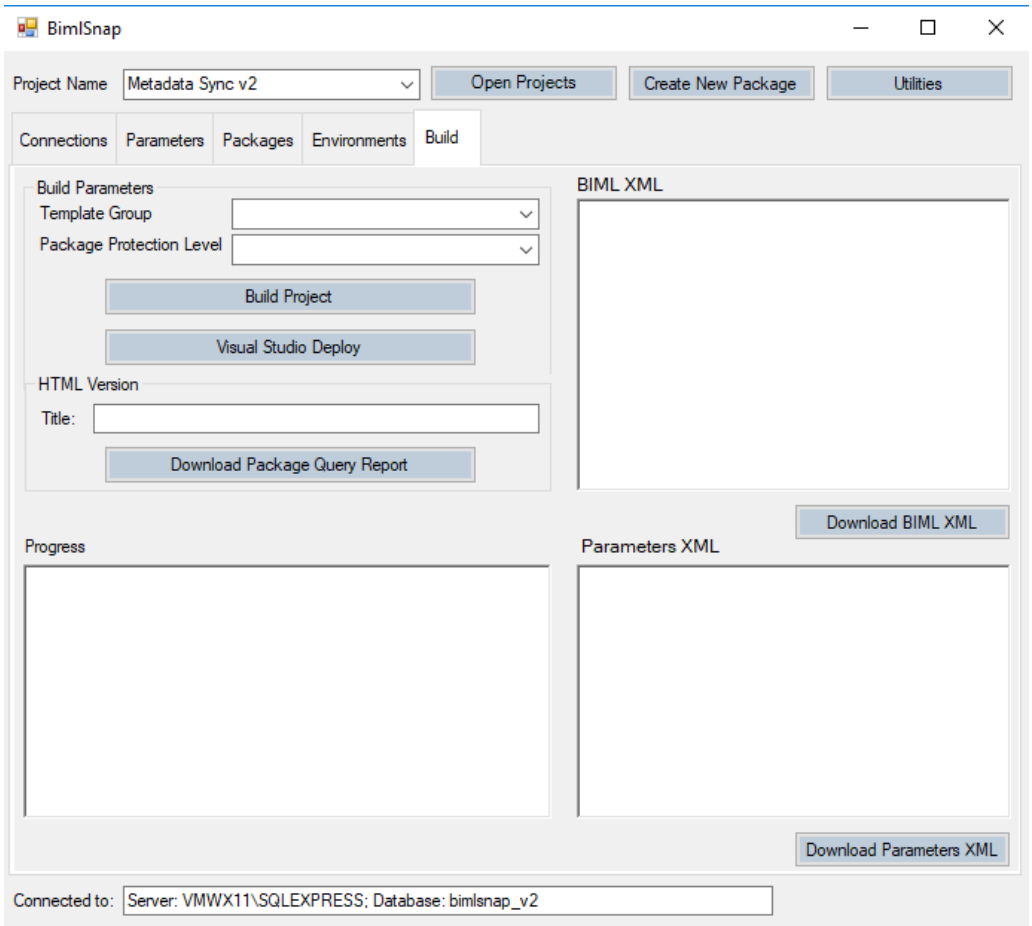


scope	schema_name	alias_name	use_expression
*	Appointment Date	ApptDateTime	CAST([] AS DATE)
*	Appointment Status	AppointmentStatus	
*	Appointment Time	ApptDateTime	CAST([] AS TIME)
*	ApptLength	ApptLength	[] + ' - ' + []
*	Check-in Time	CheckInDateTime	
*	Check-out Time	CheckOutDateTime	
*	CheckOutTime	CheckOutDateTime	
*	Encounter Number / CSN	CSN	
*	Encounter Type	EncounterTypeCode	
*	ImportDate	*	GETDATE()
*	LoadedIntoDBTable	*	0
*	Location Number	LocationNumber	
*	Nurse Leave Time	NurseLeaveDateTime	
*	Physician Enter Time	PhysicianEnterDateTime	
*	Roomed Time	RoomDateTime	
*	SameDay	SAME_DAY_YN	
*	Scheduled Date	ScheduleDate	
*	Scheduled Provider ID	ProviderNumber	
*	Sign-in Time	SigninDateTime	
*	Visit End Time	VisitEndDateTime	
*	WalkIn	WALK_IN_yn	

The flat file column alias is used in code generator 'Dim Table Merge' to customize column names used in your MERGE 'destination' table.

Build

Build tab is used to build project. By clicking “Build” tab the following will be shown.



The Build functionality transforms the previously defined connection, parameters, queries, and packages into Biml (XML) scripts.

By choosing different Package Level Protection and Template Group, or leaving it the way it is and clicking the ‘Build Project’ button, all the data that is created for the selected Project will be processed and based on that data the Biml and Parameters XML will be generated. Build progress can be tracked within the ‘Build Progress’ box. After the build is finished, you can review any error and everything that is going on behind the scene by clicking the ‘Build Logs’ button.

Those generated files can be copied or downloaded and are used to generate the SSIS packages.

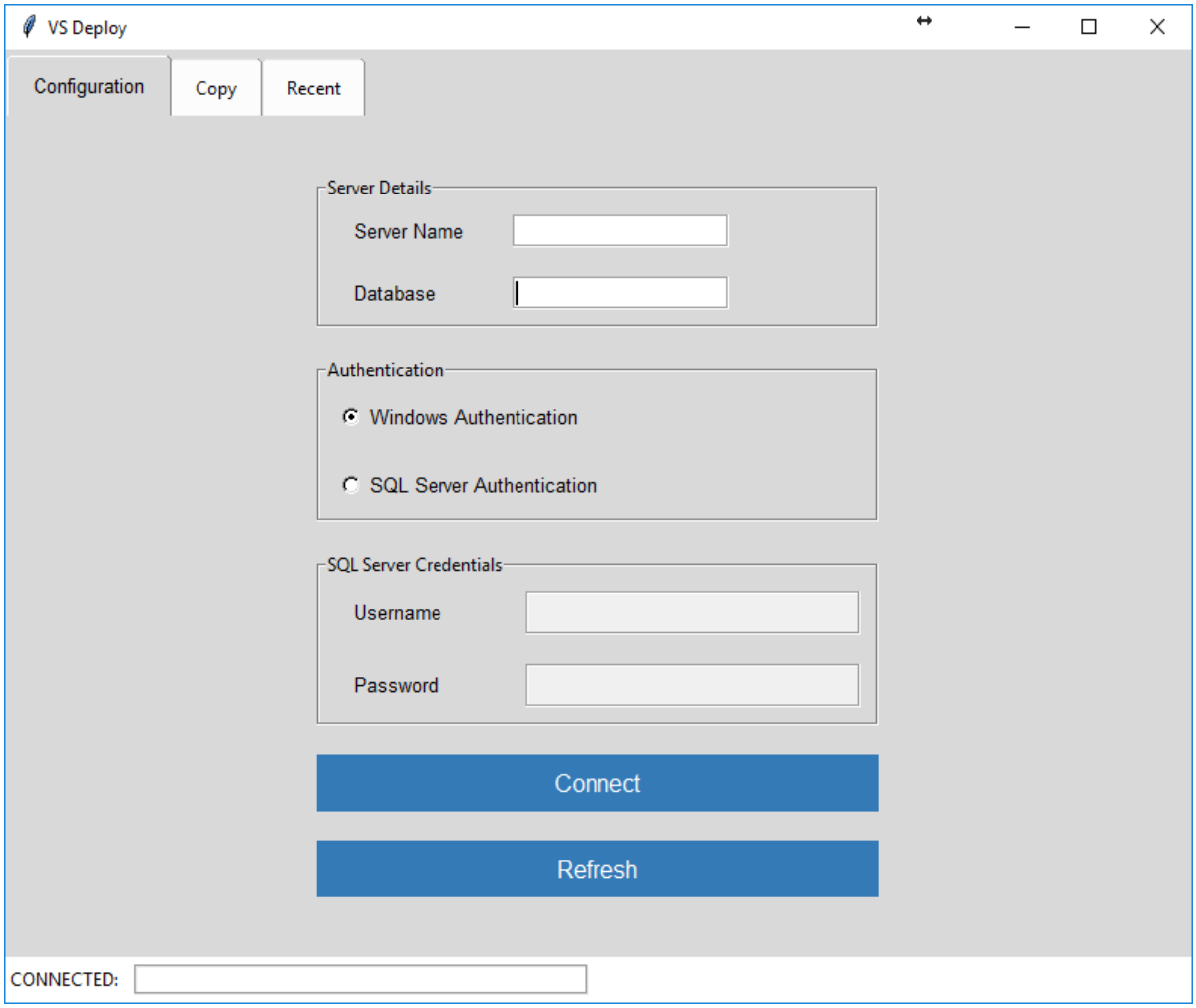
Package Protection Level

There are two ‘Template Group’ options available:

- Encrypt Sensitive with User Key
- Don’t Save Sensitive

‘Encrypt Sensitive with User Key’ option is used to encrypt sensitive information based on the credentials of the user that created (or generated) the package. This is also the default protection level that is used when creating a new project in Visual Studio (SSDT).

VS Deploy

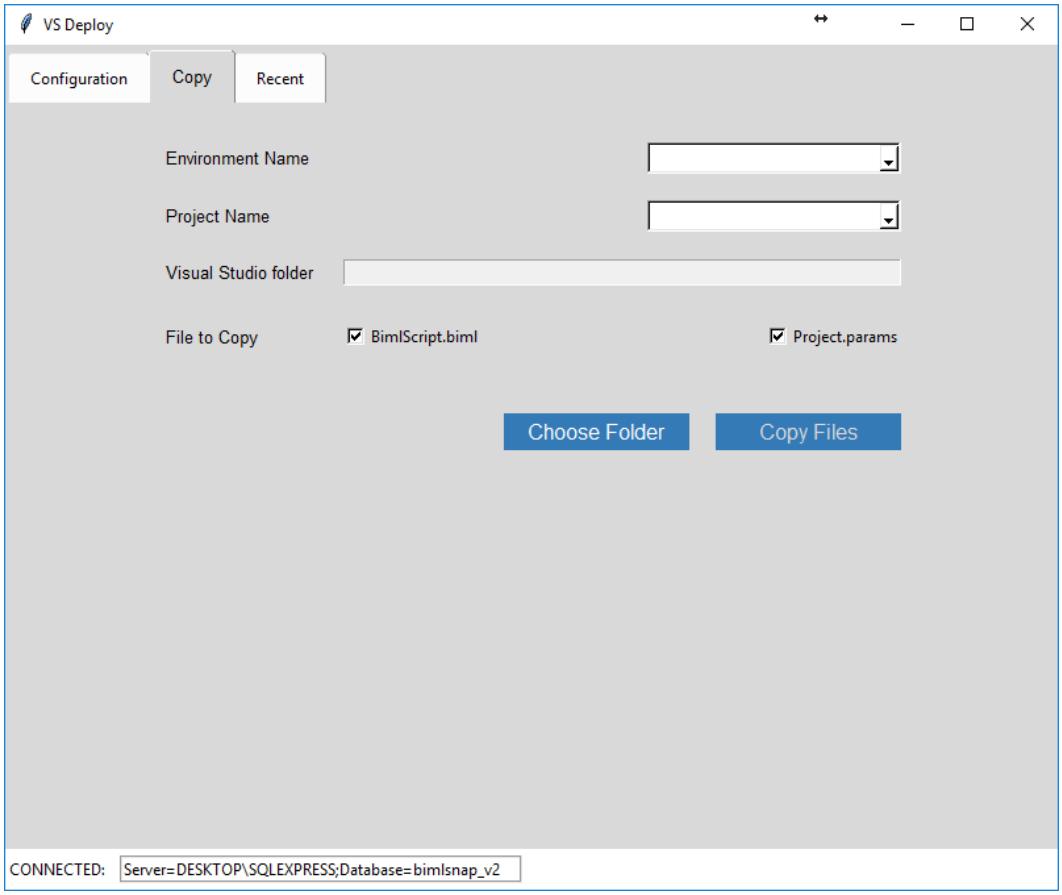


The VS Deploy application is developed to help users of BimlSnap Desktop application to easily copy files to the folder where is the Visual Studio Project for the generating SSIS packages.

The Application provide two types of connections to the database:

- Windows Authentication
- SQL Server Authentication

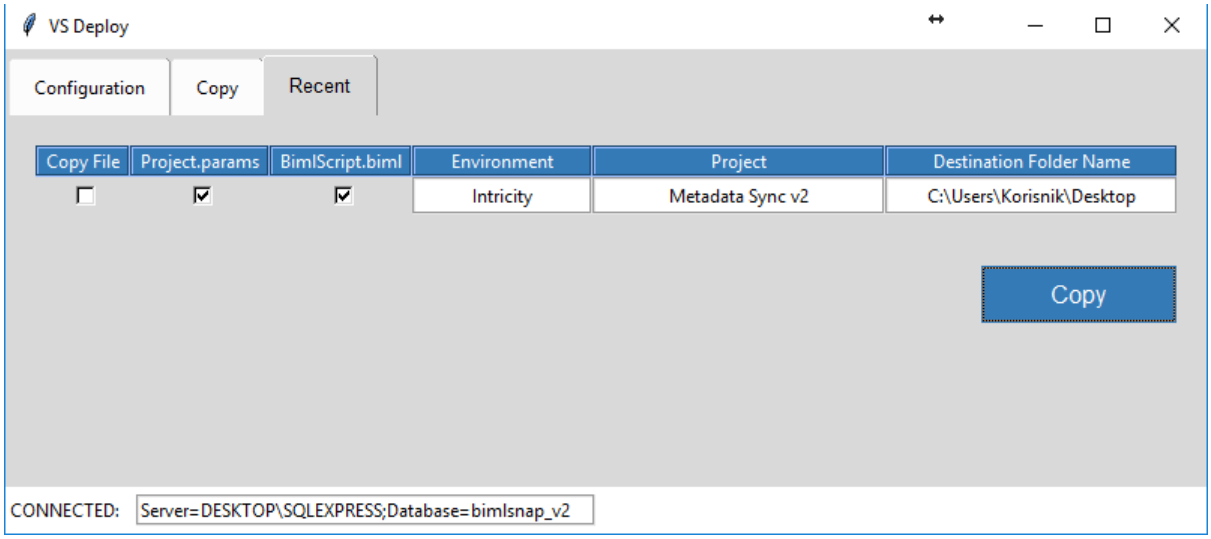
After successful connection to the database application will move to the second tab (Copy)



On the 'Copy' tab user need to choose Environment Name, Project Name from dropdowns and Visual Studio folder. After these 3 parameters are selected user can check which of the files want to be copied.

By click on the button 'Copy Files' files will be copied to the selected folder.

If user already copy some files and want to do it again he can do it on the tab 'Recent'



It only need to check in the column 'Copy File' which files will be copied. By clicking on the button 'Copy' files will be copied to their destination folders.

Package Query Report

Package Query Report is a HTML rendering of your connections and package configurations. By providing the title and clicking on the 'Download Package Query Report' the .html file will be downloaded and by opening it all the data will be displayed formatted by the groupings for each independent unit (e.g. Project, Connections, Parameters etc.).

Build Logs

The 'Build Logs' button is used to preview all the things that happened while the Build of the selected project was happening. Any potential errors can be found listed, so you know what mistake you made and you can make needed changes before doing the Build of that project again.
'Don't Save Sensitive' option can be used to omit sensitive data. This 'package' setting needs to correspond to the equivalent 'project' level setting in Visual Studio.