The SeETL Design Time

User Guide

Version 3.2.00
1st January 2019
Instant Business Intelligence
info@instantBI.com

Table of Contents

1. CHANGE CONTROL LOG 5

2. AUDIENCE 6

3. INTRODUCTION 7

4. SEETL DESIGNTIME WORKBOOK 8

4.1. SeETL Tabs 17

4.1.1. Overview 17
4.1.2. Summary 17
4.1.3. Processing 19
4.1.3.1. Parameters Added to the Processing Tab for 3.1.02 30
4.1.3.2. Parameters Added to the Processing Tab for 3.2.00 32
4.1.4. Mapping01 33
4.1.5. Batch Schedule 36
4.1.6. Last Key used 40
4.1.7. Aggregation Control 41
4.1.8. Dim Table Key Definitions 43
4.1.9. Dim Table Type2 Col Defs 45
4.1.10. Dim Table Load Control 46
4.1.11. Dim Table Key Fields 48
4.1.12. Source File Definitions 49
4.1.13. Sequence Numbers 51
4.1.14. Mappings Documentation Infa 52
4.1.15. Other Views 54
4.1.16. Dim Table Current Flags 55
4.1.17. Dim Table Levels 56
4.1.18. Update BO Universe 57
4.1.19. Translate RS Reports 58
4.1.20. Segmentation Type Code 61
4.1.21. Segmentation Version 63
4.1.22. Segment Codes 64
4.1.23. Segment Codes Cols 66
4.1.24. Product Markers 67
4.1.25. Data Model Views 68
4.1.26. Data Model Joins 72
4.1.27. Generate Stored Procedures 73
4.1.28. Document Stored Procedures 77
4.1.29. Compound Views 82
4.1.30. Create Tables 83
4.1.31. Create Count SQL 85
4.1.32. Create Runstats SQL 87
4.1.33. Create Table as Select 88
4.1.34. Create Index 89
4.1.35. Create Storage Object 90
4.1.36. Create Access Permission 91
4.1.37. Create Generic Object 92
4.1.38. SQL Statement Load 93
4.1.39. Data Profiling Worksheet 95
4.1.40. Document Reports 102
4.1.41. Project Plan 107
4.1.42. Business Requirements 112
4.1.43. Project Objectives 114

4.1.44. Critical Success Factors 116
4.1.45. Major Dimensions 118
4.1.46. Major Facts 120
4.1.47. Defined Measures 122
4.1.48. Subject Areas 124
4.1.49. Cross Reference 01 126
4.1.50. Cross Reference 02 128
4.1.51. Cross Reference 03 130
4.1.52. Cross Reference 04 132
4.1.53. Cross Reference 05 134
4.1.54. Solicitation Transaction Email 136
4.1.55. Response Transactions Email 139
4.1.56. Email Address Bounce 141
4.1.57. Do Not Email Request 142
4.1.58. Campaign Groups 143
4.1.59. Campaigns 145
4.1.60. Sub Campaigns 147
4.1.61. BI4ALL Codes Tables 149
4.1.62. LinkedIn Connections 150
4.1.63. Email Connections 151
4.1.64. Restart Batch 152
4.1.65. Variable Substitution 153
4.1.66. Delta Definitions 154
4.1.67. Product Definitions 156
4.1.68. List Definitions 158
4.1.69. List Members 159
4.1.70. List Member Deletes 161
4.1.71. Client Security Definitions 163
4.1.72. Client Software Key Definitions. 166
4.1.73. Party Interests 169
4.1.74. Source Table Definitions 171
4.1.75. Source Column Definitions 172
4.1.76. Source Index Definitions 174
4.1.77. Source To Stage Mapping Definitions 175
4.1.78. Pre Mapping 01 Definitions 177

5. THE SEETL MEMORY USAGE UPGRADES 180

6. THE SEETL DATA MODEL DOCUMENTATION DATABASE 181

7. SQL GENERATION OPTIONS 183
7.1. Configurations for the SQL Generation Option 183
7.1.1. One Step Processing 185
7.1.1.1. Type 1 dimension table processing 186
7.1.1.2. Type 2 dimension table processing without field level sensitivity. 194
7.1.1.3. Type 2 dimension table processing with field level sensitivity. 203
7.1.1.4. Fact table processing without a key on the fact table. 214
7.1.1.5. Fact table processing with a key on the fact table. 224
7.1.2. Two Step Processing without Help Key Tables 236
7.1.2.1. Type 1 dimension table processing 237
7.1.2.2. Type 2 dimension table processing without field level sensitivity. 242
7.1.2.3. Type 2 dimesnion table processing with field level sensitivity. 247
7.1.2.4. Fact table processing without a key on the fact table. 254
7.1.2.5. Fact table processing with a key on the fact table. 258
7.1.3. Two Step Processing with Help Key Tables 263
7.1.3.1. Type 1 dimension table processing 264
7.1.3.2. Type 2 dimension table processing without field level sensitivity. 268
7.1.3.3. Type 2 dimension table processing with field level sensitivity. 273
7.1.3.4. Fact table processing without a key on the fact table. 279
7.1.3.5. Fact table processing with a key on the fact table. 280

1. CHANGE CONTROL LOG

#	Date	Name	Description
3.0.01	1/10/2006	IBI Dev	Initial version for public release with SeETL^{DT}.
3.0.02	1/01/2008	IBI Dev	Updates for version 3.0.02 SeETL^{DT}.
3.1.00	1/01/2012	IBI Dev	Updates for version 3.1.00 SeETL^{DT}.
3.1.01	1/01/2013	IBI Dev	Updates for version 3.1.01 SeETL^{DT}.
3.1.02	1/01/2014	IBI Dev	Updates for version 3.1.02 SeETL^{DT}.
3.2.00	1/01/2019	IBI Dev	Updates for version 3.2.00 SeETL^{DT}. Please note some features were removed from this release.

2. AUDIENCE

The intended audiences for the SeETL[DT] User Guide are:

- Technical developers working on Enterprise Business Intelligence projects using SeETL[DT].
- IT managers considering the purchase of SeETL[DT].

3. INTRODUCTION

As the Instant Data Warehouse software evolved and developed one of the 'key innovations' was the ability to define all mappings into a spreadsheet and then generate the ETL subsystem from that spreadsheet. The 'Mapping Spreadsheet' started life as 'something necessary' as we all must document our mappings somewhere and a spreadsheet is as good as any other tool for documenting mappings.

As time went along it became clear that if we 'enhanced' the content in the spreadsheet and then used that content to generate the ETL subsystem we could actually remove the step called 'coding ETL'. This seemed like a great idea at the time, even though we had no idea how to do it.

SeETL DesignTime brings with it huge benefits and even more significant 'promise'.

The benefits today are as simple as they are outstanding:

1. There is no ETL coding exercise when using SeETL.
2. We have increased development productivity for ETL subsystems by a factor of ***10.***
3. We have reduced the cost of acquiring ETL software by a factor of ***10.***
4. All ETL is highly visible through a reporting portal based on Microsoft SQL Server Report Services.
5. All data in the SeETL Workbook can be loaded into any ODBC compliant database and can be queried in any way supported by that database.

The 'promise' of SETL is:

1. We are working on putting more and more 'ETL Definition data' into the workbook along with the functionality to exploit that data into the application.

2. It is clear that we are now able to generate ETL subsystems in any ETL tool a client would like to use. It will be as a result of client requests that we are will be motivated to write such extensions to SeETL.

SeETL has revolutionised the world of ETL development completely.

Once you have used SeETL you will wonder why you ever did anything else.

The BIG innovation of SeETL 3.2.00 is that it is being released for FREE!

We look forward to your comments and requests for enhancements on our blog posts!!!!

4. SEETL DESIGNTIME WORKBOOK

The SeETL[DT] Excel Workbook contains a number of tabs. These tabs describe processing and data to be processed that sets up information for SeETL[RT] when it starts. We have placed help text into each heading cell in the workbook. However, excel still minimises the size of the comment text on each save of the workbook so we have decided to also document these comments into this PDF based document for easier reading for our clients.

The tabs in the workbook as of SeETL[DT] Version 3.1.02 are as follows:

Tab Name	Description
Overview	This tab contains overview information for SeETL including its capabilities, copyright notice, welcome notice and pointers to documentation.
Summary	This tab defines the name of the spreadsheet to SeETL since the internal name can be different to the file name, the version, as well as a number of specific variables that need to be known by SeETL during the generation process such as database SeETL is generating for.
Processing	This tab contains a number of key pieces of information: 1. This tab defines the processing that the SeETL application will perform when it is invoked. 2. This tab tells SeETL where to place the output that is generated or the names of many of the tables into which data is placed. The names of these tables can, in some cases, be changed. It is recommended they are not changed. 3. The SeETL target database that will be used to house the target data from the ETL processing.
Mappings01	This tab is the 'heart' of SeETL. It contains all the source to target mappings for the ETL subsystem. It is the equivalent of combining all the programmed source to target mappings in an ETL subsystem into a single workbook in a spreadsheet. It is intended to be developed over time as the Architect decides how to map more of the data.
Batch Schedule	This tab allows the definition of the batch schedule to be run every period. It allows the definition of: 1. Batch pre-requisites. 2. Process Group pre-requisites. 3. Processes in the Process Groups 4. The commands that will actually be run in the processes under the OS 'system' interface.
Last Key used	This tab defines the first values of the last keys used for each dimension table. Once a last key used is loaded it should never be changed unless the Architect wants to reset the last key used for some reason. The last key used will be maintained by SeETL once it is loaded into the catalog.
Aggregation Control	This tab defines the levels of data inside dimension and fact tables in a multi-level implementation. SeETL[DT] does not yet support multi-level implementations. It is a feature planned for future development and release. SeETL[RT] is fully multi-level aware.
Dim Table Key Definitions	This tab defines the set of fields on the input stream for dimension tables that constitute the key for the dimension table.
Dim table Type 2 Col Defs	This tab defines the fields on the input stream of a Type 2 dimension that should be checked for change to determine when to close an existing record and to open a new one.
Dim Table Load Control	This tab defines if and how dimension tables will be loaded into memory. SeETL[RT] has excellent memory management facilities available to the Architect to maximise throughput while still not over-allocating memory. SeETL[RT] has linear scalability with increased processors due to how memory is managed by this worksheet.
Dim Table Key Fields	SeETL[RT] expects certain naming conventions to be applied to keys of views or tables. However, some models where SeETL[RT] has been used do not have this naming standard. Hence this tab defines the key fields of dimension tables when the tables do not comply with the naming standard. One design feature of SeETL[RT]

	is that it does not insist on any specific naming standard for anything.
Source File Definitions	This tab defines information about all source files that have been considered as input to this ETL exercise. It has proven particularly useful to extend this tab to include all fields considered, even if they are later rejected.
Sequence Numbers	This tab defines sequence numbers for fact tables. It defines the starting number and the increment of the number of numbers to allocate when with each new set of numbers allocated. It is only required to load the new sequence names once in the lifetime of any fact table.
Mappings Documentation Infa	SeETL[DT] is free and can be used in conjunction with other ETL tools. At one client where we built the prototype with SeETL[RT] and the client wanted to build the real ETL in Informatica we wrote the capability to generate mapping documentation for the Informatica programmers directly from the SeETL[RT] workbook. This saved many hundreds of hours of work.
Other Views	This tab contains any other views the Architect might like to define for an ETL implementation.
Dim Table Current Flags	A client came up with a good idea as to how we can have dimension tables 'morph' between being treated as type 1 and type 2 dimensions. This worksheet is used to define a mapping where ther type 1 dimension views are created with a current flag =1 constraint. This allows type 1 processing against a table that is actually a type 2 table.
Dim Table Levels	This tab defines how many levels of aggregation will be defined for the dimension table. In years gone past we have always built dimension tables with 9 levels of aggregation plus the detailed level. With SeETL we have made it variable by dimension table to give the user a little more flexibility in how many levels are created. The maximum is still 9 levels of aggregation per dimension table.
Update BO Universe	One of the rael problems with Business Objects is managing the names of all the objects when you would like to change them. The object name and the help text are time consuming and difficult to maintain. There is also the issue that saving universes and reports in multiple languages is very time consuming and expensive. This workbook allows the user to store the object name and the help text for the vast majority of business objects items. The key is based on the table name and the column name. By storing different sheets in the workbook it is possible to maintain multiple sets of languages for Business Object Universes. Should more people require (and pay for) better multi-language support we will further develop this application.
Translate RS Reports	BI4ALL is a new poduct that we are working on. Currently there are 60+ MicroSoft Report Services Reports in the product. We have developed the ability to translate a base report (written in English) to up to 10 other languages. For example, we have already completely translated all reports to German and we are able to re-run the translation process at will. This is a huge breakthrough because it means that the translation of reports between languages is automatic. It also means there is only one base set of reports to maintain.
Segmentation Type Code	The SeETL[RT] Segmentation Engine requires a number of tables to be created and made available to present the data to the end user correctly and with the correct descriptions while allowing those descriptions to be flexible. The Segmentation Type Codes allow the user to define the text of the types of Segmentation that will be implemented in this instance of the Segmentation Engine.
Segment Codes	The SeETL[RT] Segmentation Engine allows the user to define the text that will appear on reports for the segment codes. This tab contains the segment codes and their proper descriptions to be placed onto reports. This set of rows is the text that will be used for segment codes where the text is placed as row headings in a column down the side of reports. Therefore the option was created to have different text for the same segment codes depending on whether the description was to be used in row headings or column headings.
Segment Codes Cols	The SeETL[RT] Segmentation Engine allows the user to define the text that will appear on reports for the segment codes. This tab contains the segment codes and their proper descriptions to be placed onto reports. This set of rows is the text that will be used for segment codes where the text is placed across the top of reports.

	Therefore the option was created to have different text for the same segment codes depending on whether the description was to be used in row headings or column headings.
Product Markers	The SeETL[RT] Segmentation Engine uses a point's based system allocated to products to calculate the placement of customers into segments based on their product purchase or usage history. The Product Markers table is the table used to allocate points to sales or usage to place customers into those segments.
Data Model Views	The SeETL[DT] Workbook has now been updated to be able to store the ***BI4ALL Data Models*** in the workbook as well. This is a major upgrade to the SeETL[DT] and further represents our commitment and ability to place as much as is possible into the workbook and to manage as much of the development process as possible out of the workbook with the complementary SQL Server database and reports. Naturally, if the client wishes to put their own data model into the workbook the workbook accommodates this.
Data Model Joins	The SeETL[DT] Workbook has been upgraded to support the definition of all joins in the ***BI4ALL Data Models*** into the workbook. This is used to be able to update business objects universes with all the joins needed to be placed into the universe. It also allows the SeETL[DT] to be able to generate Referential Integrity Constraints into the ***Data Model Documentation Database*** that can be re-engineered into various data modelling tools. Naturally, if the client wishes to put their own data model into the workbook the workbook accommodates this.
Generate Stored Procedures	The SeETL[DT] Workbook has been upgraded to enable to manipulate input text of stored procedures to then produce numerous output files. One output file is produced for each language desired for each stored procedure. This is a very important update because it allows, for the first time, the ability to manage multi-language stored procedures in a reasonably sensible and manageable way.
Document Stored Procedures	The SeETL[DT] Workbook has been upgraded to enable the documentation of stored procedures so that these stored procedures can be easily searched and reported on as a catalogue of stored procedures. IBI believes that in the future Stored Procedures will be much more important as they are created and catalogued to be able to be reused by a variety of front end tools.
Compound Views	The SeETL[DT] Workbook has been upgraded to allow the definition of compound views. A compound view is one that uses a statement such as 'union all' to join the result sets of two or more other select statements. It has been developed at the request of a client to be able to manage unions. The initial implementation is relatively simplistic but it is at least a starting point.
Create Tables	The SeETL[DT] Workbook has been upgraded to allow the definition of create table statements. This has proven problematic as the databases have such different create table statements available to them. The approach that has been taken is to allow large quantities of text to be entered into the create table statements. This will not be ideal for all situations and we accept this. We are not proposing the SeETL[DT] Workbook is a replacement for a decent Data Modeling tool. If the data modelling tool can write to our documentation database that would also be a great new feature. We have introduced the ability to create tables using the SeETL[DT] to the best effect we can. We trust that clients will find this useful.
Create Count SQL	One of our major clients asked us if we were willing to significantly extend the workbook to store and create many of the other objects necessary to build a data warehouse using the ***BI4ALL Data Models*** and our standard SeETL Processing Techniques. This seemed like a really good idea that we had not really ventured into before. The main reason we did not venture into this before was that there are many small amounts of code fragments that are created during a project. They take very little time to develop and so, from IBIs point of view, it was not cost justifiable to do so. However, this client pointed out that they saw great value in being able to have ALL the objects that needed to be managed in the workbook and ALL those objects recorded in the dictionary. When one considers the idea of ALL objects being managed by the workbook

	process then the value of doing so is immediately apparent. It becomes possible to query the database to know and understand all the objects that are related to a BI implementation. This is a LOT more objects than just those objects that are required for use by SeETL. For example? Reports can also be catalogued into the dictionary. As can project plans. These are two items that we would NEVER have considered placing into the workbook as we do not manage them via the workbook on our projects. However. If they ARE in the Workbook it is immediately obvious how valuable that becomes. Especially in the ability to link fields on a reports/screen back to the source or to link project tasks with reports, ETL, Table development. Hence. In this release we see a large amout of sheets added for these 'extra' fragments of code. The SeETL[DT] Workbook has been upgraded to generate the 'count the number of input rows' sql. This is the simple generation of SQL fragments that are normally written by hand to count input rows for auditinbg purposes.
Create Runstats SQL	The SeETL[DT] Workbook has been upgraded to store the commands that perform runstats on the various flavours of databases. It simply stores the command that can then be generated to a file. The command can also be loaded into the database and scheduled into the schedule by the system architect if so desired.
Create Table as Select	The SeETL[DT] Workbook has been upgraded to store the commands that perform create table as select on the various flavours of databases. The current version simply captures the full command. It is obvious that it would be desirable to be able to have more complex definitions. For the time being we recommend that any more complex definition be achieved by creating a view and then having that view be the object from which the table is created with a select *. In this way the dictionary will maintain a version of the view which can be used in any query against the dictionary.
Create Index	The SeETL[DT] Workbook has been upgraded to store the commands that perform create indexes as select on the various flavours of databases. The current version simply captures the full command. The differences in 'create index' statements across the various databases is very wide. Therefore we have not attempted to generate the statement itself.
Create Storage Object	The SeETL[DT] Workbook has been upgraded to store the commands that perform create storage objects such as tablespaces and filegroups. The current version simply captures the full command and allows a 'drop and create' of the storage object.
Create Access Permissions	The SeETL[DT] Workbook has been upgraded to store the commands that perform the drop and create of access permissions to databases. The current version simply captures the full command.
Create Generic Object	The SeETL[DT] Workbook has been upgraded to store the commands that perform the drop and create of any generic object that is not one of the objects currently supported to be specifically dropped and created. It is a 'catch all' for any other database object that is required to be managed.
SQL Statement Load	The SeETL[DT] Workbook has been upgraded to load SQL Statements that are stored on file systems and load them into the the data warehouse database. This is a major enhancement that is discussed in detail in the documentation. It provides the ability to maintain versions of sql statements inside the data warehouse database to vastly enhance the auditability and security of statements that can be executed against the database. The idea is that all statements that can be executed against the production database are loaded into the SQL Statement table and executed from there using the statement execution utility. This utility then logs the start/stop times of the SQL statement as well as the number of rows processed. This marks a major enhancement for the ability to execute SQL against the database and the auditability and security of those statements. This is because all such statements can only be updated and executed by the authorised userids that have update access to these tables. This will provide a major deterrent to any

	malicious efforts to alter data as well as a great new 'safety net' for any accidential updates to data. In terms of a diagnostic tool. Because it will now be possible to audit all statements executed it will be possible to detect the exact statements that were run against a database at any given time, who ran it, and how many rows it affected. When used on conjunction with the normal logs found in most databases this provides a powerful new tool for protecting the asset of the data in the data warehouse.
Document Reports	The SeETL[DT] Workbook has been upgraded to store documentation of reports across all reporting tools used by a company. This is, of course, nothing to do with data models and ETL. However, virtually every client we talk to has no repository of the reports that they have in production and no desire to build that repository despite the fact it would be very useful. Further? No vendor has any interest in making life easier for clients across the various reporting tools. We have decided to provide such a feature with a set of reports to go with it. These reports will allow a client to be able to better understand all the reports, online and printed, that are available across the organisation. It will also record who owns them, who supports them and how often they run. Naturally this is a 'gabage in garbage out' proposition and will required maintenance efforts from the client to keep the repository up to date.
Project Plan	The SeETL[DT] Workbook has been upgraded to being able to enter and maintain the project plan in the SeETL[DT] Workbook. We see that managing the project plan out of the SeETL[DT] Workbook as the next logical step improving the overall functionality and breadth of the product to allow more and more of the required data to run the project placed into the workbook.
Project Objectives	The SeETL[DT] Workbook has been upgraded to store documentation of Project Objectives for any particular project though in this case the emphasis is on Business Intelligence Projects. The Project Objectives entered are made available to all users via a suite of reports.
Project Objectives	The SeETL[DT] Workbook has been upgraded to store documentation of Project Objectives for any particular project though in this case the emphasis is on Business Intelligence Projects. The Project Objectives entered are made available to all users via a suite of reports.
Critical Success Factors	The SeETL[DT] Workbook has been upgraded to store documentation of Critical Success Factors for any particular project though in this case the emphasis is on Business Intelligence Projects. The Critical Success Factors entered are made available to all users via a suite of reports.
Major Dimensions	The SeETL[DT] Workbook has been upgraded to store documentation of Major Dimensions for any Business Intelligence Projects. The purpose of this WorkBook is to serve as a documentation step early in the project rather than to actually start designing the data model. This is a mandatory step for companies that are not buying a data model into the project. The Major Dimensions entered are made available to all users via a suite of reports.
Major Facts	The SeETL[DT] Workbook has been upgraded to store documentation of Major Facts for any Business Intelligence Projects. The purpose of this WorkBook is to serve as a documentation step early in the project rather than to actually start designing the data model. This is a mandatory step for companies that are not buying a data model into the project. The Major Facts entered are made available to all users via a suite of reports.
Defined Measures	The SeETL[DT] Workbook has been upgraded to store documentation of Defined Measures for any Business Intelligence Projects. The purpose of this WorkBook is to serve as a documentation step early in the project rather than to actually start designing the data model. This is a mandatory step for companies that are not buying a data model into the project. The Defined Measures entered are made available to all users via a suite of reports.
Subject Areas	The SeETL[DT] Workbook has been upgraded to store documentation of Subject Areas for any Business Intelligence Projects. The purpose of this WorkBook is to

	serve as a documentation step early in the project rather than to actually start designing the data model. This is a mandatory step for companies that are not buying a data model into the project. The Subject Areas entered are made available to all users via a suite of reports.
Cross Reference 01	This Sheet contains the SeETL Cross References for Project Objectives and Business Requirements.
Cross Reference 02	This Sheet contains the SeETL Cross References for Project Objectives, Business Requirements and Major Dimensions.
Cross Reference 03	This Sheet contains the SeETL Cross References for Project Objectives, Business Requirements and Major Facts.
Cross Reference 04	This Sheet contains the SeETL Cross References for Project Objectives, Business Requirements and Measures.
Cross Reference 05	This Sheet contains the SeETL Cross References for Tables to Subject Areas.
Solicitation Transactions Email	The SeETL[DT] Workbook has been upgraded to store transaction requests for outbound Emails. Outbound emails are a part of any companies marketing needs today. The purpose of this WorkBook is to serve as a data entry device for email solicitation transactions. Email transactions can be created via other tools and integrated into the data warehouse. This is just a simple tool that allows SeETL to also generate the email solicitation transactions if desired. The Email transactions entered are processed into the BI4ALL data models and made available to all users via a suite of reports.
Response Transactions Email	The SeETL[DT] Workbook has been upgraded to store response transactions from outbound Emails. Email marketing is a part of any companies marketing needs today. The purpose of this WorkBook is to serve as a data entry device for email response transactions. Email response transactions can be created via other tools and integrated into the data warehouse. This is just a simple tool that allows SeETL to also capture the responses from email solicitation transactions if desired. The Email response transactions entered are processed into the BI4ALL data models and made available to all users via a suite of reports.
Email Address Bounce	The SeETL[DT] Workbook has been upgraded to store email bounce indicators from outbound Emails. The Email address bounce transactions entered are processed into the BI4ALL data models and made available to all users via a suite of reports.
Do Not Email Request	The SeETL[DT] Workbook has been upgraded to store Do Not Email requests that have been received via whatever mechanism this information is captured. The Do Not Email requests transactions entered are processed into the BI4ALL data models and made available to all users via a suite of reports.
Campaign Groups	The SeETL[DT] Workbook has been upgraded to store Campaign Groups as part of the campaign definition data model. This covers all sorts of Campaigns which may be created and run to optimise company performance. This is not limited to marketing campaigns. Campaign management is a part of any companies needs today. The purpose of this WorkBook is to serve as a data entry device for Campaign Groups. Campaign Groups can be created via other tools and integrated into the data warehouse. This is just a simple tool that allows SeETL to also capture Campaign Groups if desired. The Campaign Groups entered are processed into the BI4ALL data models and made available to all users via a suite of reports.
Campaigns	The SeETL[DT] Workbook has been upgraded to store Campaigns as part of the campaign definition data model. This covers all sorts of Campaigns which may be created and run to optimise company performance. This is not limited to marketing

	campaigns. Campaign management is a part of any companies needs today. The purpose of this WorkBook is to serve as a data entry device for Campaigns. Campaigns can be created via other tools and integrated into the data warehouse. This is just a simple tool that allows SeETL to also capture Campaigns if desired. The Campaigns entered are processed into the BI4ALL data models and made available to all users via a suite of reports.
Sub Campaigns	The SeETL[DT] Workbook has been upgraded to store Sub Campaigns as part of the campaign definition data model. This covers all sorts of Campaigns which may be created and run to optimise company performance. This is not limited to marketing campaigns. Campaign management is a part of any companies needs today. The purpose of this WorkBook is to serve as a data entry device for Sub Campaigns. Sub Campaigns can be created via other tools and integrated into the data warehouse. This is just a simple tool that allows SeETL to also capture Sub Campaigns if desired. The Sub Campaigns entered are processed into the BI4ALL data models and made available to all users via a suite of reports.
BI4ALL Codes Tables	The SeETL[DT] Workbook has been upgraded to store input for the BI4ALL Codes Tables. This is to provide a simple interface with which to load and maintain the BI4ALL Codes Tables. The BI4ALL Codes Tables entered are processed into the BI4ALL data models and made available to all users via a suite of reports.
LinkedIn Connections	The SeETL[DT] Workbook has been upgraded to store input from LinkedIn into the vm_party table of BI4ALL. This is to provide a simple interface with which to load and maintain the LinkedIn contacts for a company and the sales reps and other appropriate staff for a company. There are a lot of people on LinkedIn now and maintaining those links into the data warehouse is a useful thing to do. The LinkedIn connections that are downloaded from LinkedIn are processed into the BI4ALL data models and made available to all users via a suite of reports.
Email Connections	The SeETL[DT] Workbook has been upgraded to store input from Email Connections into the vm_party table of BI4ALL. This is to provide a simple interface with which to load and maintain the many Email Connections that people manage to gather for a company. Currently these email connections are merely loaded into the vm_party table. There is no attempt to do any further processing at the current point in time. There are a lot of people on email now and maintaining those links into the data warehouse is a useful thing to do. The email connections that are downloaded from various sources such as outlook or personal email packages are processed into the BI4ALL data models and made available to all users via a suite of reports.
Restart Batch	The SeETL[DT] Workbook has been upgraded to store input from the ETL Administrator as to which batch to restart. This is a simple interface so that manual restarts of batches are no longer needed. This is a rare event and the batch restart is rarely needed. Today it is the simply issuance of the appropriate SQL statement by the authorised administrator. This function merely means that there is a log in the workbook that the restart took place. There is no log in the database that the restart statement was issued.
Variable Substitution	The SeETL[DT] Workbook has been upgraded to be able to perform variable substitution of any unique string in any cell in the workbook into another string. This capability is intended to allow the user to alias database names, table owner names, directory paths and similar things.
Delta Definitions	The SeETL[DT] Workbook has been upgraded to be able to perform the generation of delta detection between two tables, the current table and the minus 1 version of the

	table. The delta processing can also optionally forward the deltas to the staging area should the designer wish. All delta detection is performed in SQL inside the staging area or via link to the current version of the table. This feature is a "must have" feature for all people who work in regression testing for systems that use relational databases.
Product Definitions	The SeETL[DT] Workbook has been upgraded to be able to store and load product definitions for the BI4ALL Data Models. This is part of the extension of SeETL[DT] and BI4ALL Integration that was started with 3.1.00.
List Definitions	The SeETL[DT] Workbook has been upgraded to be able to store List Definitions. These are the names and attributes of lists that might be kept by any company for such things as marketing campaigns. These list definitions can be loaded into the BI4ALL Data Models.
List Members	The SeETL[DT] Workbook has been upgraded to be able to store List Member Definitions. These are the names and attributes of partys that lists that might be members of lists that are kept by any company for such things as marketing campaigns. These list member definitions can be loaded into the BI4ALL Data Models.
List Member Deletes	The SeETL[DT] Workbook has been upgraded to be able to store List Member Definitions. These are the names and attributes of partys that lists that might be members of lists that are kept by any company for such things as marketing campaigns. These list member definitions can be loaded into the BI4ALL Data Models. This Worksheet records the List Members that have been deleted from a List along with the reason code for the deletion from the list.
Client Security Definitions	The SeETL[DT] Workbook has been upgraded to be able to store Client Security Definitions. These are the names and attributes of companies and partys that are allowed to access portions of a data warehouse. In 3.1.02 we are adding support to the BI4ALL Data Models to optionally allow customers to co-habit data in one database so as to get lower price support for their hosted BI4ALL data warehouse. In such a data warehouse each row will be tagged with a client number. Each user who accesses the data warehouse will be authenticated and the client number associated with the user will be retrieved. From this point the user will only be able to access rows with the correct client number on the row. This will allow the co-habitation of many clients data warehouses in one database on one machine thus reducing maintenance costs.
Party Interests	The SeETL[DT] Workbook has been upgraded to be able to store the interests of partys with respect the items that they might wish to be contacted about. These Party Interests will be loaded into the BI4ALL data models and made available to analysts who wish to select Partys for campaigns of all sorts.
Source Tables	The SeETL[DT] Workbook has been upgraded to be able to store the definitions of source tables as part of the automatically generated delta detection processing. These source tables are used to define what tables will be placed in to the staging area via the delta detection process.
Source Columns	The SeETL[DT] Workbook has been upgraded to be able to store the definitions of source columns from source systems as part of the automatically generated delta detection processing. These source columns are used to define what columns will be placed in to the staging area via the delta detection process.
Source Indexes	The SeETL[DT] Workbook has been upgraded to be able to store the definitions of source indexes from source systems as part of the automatically generated delta detection processing. These source indexes are used to define what columns are parts of primary keys in the source tables and the fact they are primary keys is used in the generation of the delta detection processing as well as in the generation of the landing and staging tables.
Source To Stage Maps	The SeETL[DT] Workbook has been upgraded to be able to store high level maps of the source to stage mappings. This workbook is used to generate a variety of objects for the source to stage mappings including the delta detection sql that performs this task.
Pre Mappings	The SeETL[DT] Workbook has been upgraded to be able to store what we have

	called “Pre Mappings”. These “Pre-Mappings” are high level mappings of staging tables to target data warehouse tables which are placed in to the dictionary and used to generate the mappings to place in to the mapping workbook. The reason these are so interesting is that this gives about a 7X productivity improvement over just using the mapping spreadsheet.

4.1. SeETL Tabs

This section documents the tabs in the workbook in detail.

4.1.1. Overview

The Overview tab is used to store

1. A welcome message to the use of SeETL.

2. The Copyright Notice. The SeETL Workbook is copyrighted and using the workbook requires registration on our web site and allocation to our forum to be SeETL^{DT} client. This is free and is merely a mechanism for us to know how many people are using SeETL^{DT}.

3. Pointers to documentation. The values of fields are documented in this document as well as the SeETL ^{RT} User Guide.

4. Description of 'General Use' fields.
   Some fields provide similar functionality in each of the sheets in which they are used.

5. Notice regarding specific fields.
   Fields which are not highly common across workbooks are noted as having comments inside the cells in the workbook. Unfortunately Excel does not properly resize the comment cells when the workbook is re-opened which has motivated us to write this document.

4.1.2. Summary

The Summary tab is used to store some SeETL Workbook wide parameters that are required to be known when the ETL Subsystem is generated.

The Summary tab contains the following fields.

Field Name	Description
Spreadsheet Name	This is the name of the spreadsheet that will be stored in the metadata database. Reports can be written against this spreadsheet name.
Spreadsheet Version	This is the version number of the spreadsheet that will be stored in the metadata database. Reports can be written against this spreadsheet version number. When a spreadsheet with a specific version number is reloaded the portions being reloaded will be updated. When the version number is incremented the old data is left in the metadata database. In this way all prior versions of a spreadsheet may be maintained in the metadata database. This means that updates to the spreadsheet can be versioned. Generally we do not recommend maintaining copies of 'every version'. We recommend that only significant changes are versioned.
Number of mappings	Not currently used.
Batch Schedule Rows	Not currently used.
Processing Rows	Not currently used.
Staging Area Database	The database type of the staging area. Currently supported databases are: • Oracle • SQL Server • DB2 • Sybase IQ • Sybase ASE
Data Warehouse Database	The database type of the Data Warehouse/target database. Currently supported databases are: • Oracle

	• SQL Server • DB2 • Sybase IQ • Sybase ASE

4.1.3. Processing

The Processing tab is used to store processing directives to be interpreted by the SeETL[DT] application. The directives are set to Yes and No depending on whether the Architect wants that portion of the SeETL Workbook loaded into the Metadata database or generated.

The Processing tab contains the following fields.

Field Name	Description
Create Source Views	This parameter tells the Generator to generate the staging area views. Currently only one staging area is supported. We have multiple staging area support on the 'to-do' list.
Create Target Views	This parameter tells the Generator to generate the SeETL views.
Create Presentation Views	This parameter tells the Generator to generate the presentation views.
Create Other Views	This parameter tells the Generator to generate the other views.
Create Compound Views	This parameter tells the Generator to generate the compound views.
Load Mapping Spreadsheet	This parameter tells the Generator to delete the current instance of the mapping spreadsheet and then load the mappings01 worksheet from this workbook.
Load Other Views Documentation	This parameter tells the Generator to delete the current instance of the Other Views Documentation and then load the Other Views Documentation worksheet from this workbook.
Load Compound Views Documentation	This parameter tells the Generator to delete the current instance of the Compound Views Documentation and then load the Compound Views Documentation worksheet from this workbook.
Load Batch Schedule	This parameter tells the Generator to truncate and load the Batch Schedule that is contained in the Batch Schedule tab. The entire batch schedule will be deleted and reloaded if this field is set to 'Y'.
Truncate Last Key Used	Truncate the 'Last Key Used' table whose name is defined in the column 'FQN Last Key Used'.
Truncate Aggregation Control	Truncate the 'Aggregation Control' table whose name is defined in the column 'FQN Aggregation Control'.
Truncate Dim Table Key Definitions	Truncate the 'Dim Table Key Definitions' table whose name is defined in the column 'FQN Dim Table Key Definitions'.
Truncate Dim Table Load Control	Truncate the 'Dim table Load Control' table whose name is defined in the column 'FQN Dim table Load Control'.
Truncate Dim Table Type2 Col Defs	Truncate the 'Dim table Type 2 Column Definitions' table whose name is defined in the column 'FQN Dim table Type 2 Col Defs''.
Truncate Source File Definitions	Truncate the 'Source File Definitions' table whose name is defined in the column 'FQN Source File Definitions'.
Truncate Sequence Numbers	Truncate the 'Sequence Numbers' table whose name is defined in the column 'FQN Sequence Numbers'.
Truncate Mappings Documentation INFA	Truncate the 'Mappings Documentation INFA' table whose name is defined in the column 'FQN Mappings Documentation INFA'.
Truncate Data Model Joins	Truncate the 'Data Model Joins' table whose name is defined in the column 'FQN Data Model Joins'.
Restart Abended Batch	This parameter allows the user to restart a batch that has partially or fully failed for whatever reason. It is set to "Yes" to restart the specified batch.
Load Last Key Used	Load the Last Key Used worksheet into the 'Last Key Used' table whose name is defined in the column 'FQN Last Key Used'.
Load Aggregation Control	Load the Aggregation Control worksheet into the 'Aggregation Control' table whose name is defined in the column 'FQN Aggregation Control'.
Load Dim Table Key Definitions	Load the Dim Table Key Definitions worksheet into the 'Dim Table Key Definitions' table whose name is defined in the column 'FQN Dim Table Key Definitions'.
Load Dim Table Load Control	Load the Dim Table Load Control worksheet into the 'Dim table Load Control' table whose name is defined in the column 'FQN Dim table Load Control'.
Load Dim Table Type2 Col Defs	Load the Dim Table Type 2 Col Defs worksheet into the 'Dim table Type 2 Column Definitions' table whose name is defined in the column 'FQN Dim table Type 2 Col Defs''.

Load Source File Definitions	Load the Source File Definitions worksheet the 'Source File Definitions' table whose name is defined in the column 'FQN Source File Definitions'.
Load Sequence Numbers	Load the Sequence Numbers worksheet into the 'Sequence Numbers' table whose name is defined in the column 'FQN Sequence Numbers'.
Load Mappings Documentation INFA	Load the Mappings Documentation INFA worksheet into the 'Mappings Documentation INFA' table whose name is defined in the column 'FQN Mappings Documentation INFA'.
Update BO Universe	This parameter tells the Generator to update the BO Universe.
Update BO Universe Joins	This parameter tells the generator to update the joins in the BO Universe that you propose to update.
Translate Report Services Reports	This parameter tells the generator to translate the Report Services Reports from the base language to up to 10 languages according to the Translate RS Reports tab in the workbook.
Generate Data Model Views	This parameter tells the generator to generate the views as defined in the Data Model Views worksheet.
Load Data Model Views Documentation	This parameter tells the generator to load the Data Model Views into the documentation database so that it can be published and reported out to the IT Support and business community.
Generate Input Views Row Count	This parameter tells the generator to generate the select statements specified to be generated in the worksheet called "Create Count SQL".
Generate SQL for Runstats	This parameter tells the generator to write to the files the create sunstats statements that are stored in the worksheet called "Create Runstats SQL".
Generate SQL for Create Table As	This parameter tells the generator to write to the files the create table as statements that are stored in the worksheet called "Create Table As Select".
Generate SQL for Indexes	This parameter tells the generator to write to the files the create table as statements that are stored in the worksheet called "Create Index".
Generate SQL for Access Rights	This parameter tells the generator to write to the files the create access rights statements that are stored in the worksheet called "Create Access Persmission".
Generate SQL for Storage	This parameter tells the generator to write to the files the create storage objects statements that are stored in the worksheet called "Create Storage Object".
Generate SQL for Generic Objects	This parameter tells the generator to write to the files the create generic objects statements that are stored in the worksheet called "Create Generic Object".
Generate Email Solicitations	This parameter tells the generator to read the Solicitation Transactions Email worksheet and generate the outbound emails based on the information in the worksheet.
Load Email Solicitations into Staging	Load the Solicitation Transactions Email worksheet into the 'Staging Email Solicitations' table whose name is defined in the column 'FQN Staging Solicitation Email table'.
Load Email Responses into Staging	Load the Response Transactions Email worksheet into the 'Staging Email Responses' table whose name is defined in the column 'FQN Staging Response Email table'.
Load Email Address Bounces into Staging	Load the Email Address Bounce worksheet into the 'Staging Email Address Bounces' table whose name is defined in the column 'FQN Email Address Bounce table'.
Load Do Not Email Addresses into Staging	Load the Do Not Email Request worksheet into the 'Staging Do Not Email Address' table whose name is defined in the column 'FQN Do Not Email table'.
Load Campaign Groups into Staging	Load the Campaign Groups worksheet into the 'Staging Campaign Groups' table whose name is defined in the column 'FQN Staging Campaign Groups table'.
Load Campaigns into Staging	Load the Campaigns worksheet into the 'Staging Campaigns' table whose name is defined in the column 'FQN Staging Campaigns table'.
Load Sub Campaigns into Staging	Load the Sub Campaigns worksheet into the 'Staging Sub Campaigns' table whose name is defined in the column 'FQN Staging Sub Campaigns table'.
Load Linkedin Connections into Staging	Load the LinkedIn Connections worksheet into the 'Staging Linked In Connections' table whose name is defined in the column 'FQN Linkedin Connections table'.
Load Email Connections into Staging	Load the Email Connections worksheet into the 'Staging Email Connections' table whose name is defined in the column 'FQN Email Connections table'.

Load BI4ALL Codes into Staging	Load the BI4ALL Codes Tables worksheet into the 'Staging BI4ALL Codes' table whose name is defined in the column 'FQN BI4ALL Codes table'.
Create Data Profiling SQL	This parameter tells the generator to generate the Data Profiling SQL that can be run on any database.
Load Input Views Row Count Documentation	This parameter tells the generator to load the definitions of the statements to generate row counts from the worksheet called "Create Count SQL" into the database.
Load Runstats Documentation	This parameter tells the generator to load the create runstats statements stored in the worksheet called "Create Count SQL" into the database.
Load SQL for Create Table As Documentation	This parameter tells the generator to load the create tables as statements stored in worksheet called "Create Table As Select" into the database.
Load SQL for Indexes Documentation	This parameter tells the generator to load the create index statements stored in worksheet called "Create Index" into the database.
Load SQL for Access Rights Documentation	This parameter tells the generator to load the create index statements stored in worksheet called "Create Access Persmission" into the database.
Load SQL for Storage Documentation	This parameter tells the generator to load the create storage objects statements stored in worksheet called "Create Storage Object" into the database.
Load SQL for Generic Objects Documentation	This parameter tells the generator to load the create generic objects statements stored in worksheet called "Create Generic Object" into the database.
Load Data Model Joins	This parameter tells the generator to load the Data Model Joins into the documentation database so that it can be published and reported out to the IT Support and business community.
Load Stored Procedure Documentation	This parameter tells the generator to load the Stored Procedure Documentation into the documentation database so that it can be published and reported out to the IT Support and business community.
Load Reports Documentation	This parameter tells the generator to load the Reports Documentation into the documentation database so that it can be published and reported out to the IT Support and business community.
Load IBI Project Plan	This parameter tells the generator to load the Project Plan Documentation into the documentation database so that it can be published and reported out to the IT Support and business community.
Load SQL Statements	This parameter tells the generator to load the specified SQL Statements into the data warehouse database so that these statements can be later executed from the database directly.
Connect to SMTP Serve Once Only	This parameter tells the emailer portion of SeETL to only connect to the SMTP server once at the beginning of the email sending and to reuse that connection for all emails. This is the most standard way of sending out emails. However, when set to "No" the emailer will read the smtp server name, userid, and password for each email to send out and may send each individual email from a different email address or even email server if that is what the user desires.
Connect to SMTP Serve Once Per Email	This parameter tells the emailer portion of SeETL to connect to the SMTP server once for each email message that is being sent. It has to be specifically set to "Yes" to cause this to happen. We have introduced BOTH parameters so there can be no question that the user is making an explicit decision as to how he is going to use the email program.
Number of Seconds to Sleep Between Emails	This parameter tells the emailer portion of SeETL to sleep between emails so as to not overload the smtp server. Some web hosting providers are not configured to handle large amounts of emails being sent out via their smtp servers. So they ask that any emailer program be able to throttle the rate at which emails are sent out. Most commonly we have been told that 5 to 10 seconds between emails has been asked for. Of this value is set to zero or omitted then the emailer program will not sleep between emails. It will send them just as fast as the smtp server responds.
Perform Variable Substitution	This parameter tells the generator to read the Variable Substitution Worksheet and to apply these variable substitutions for every cell read from the workbook.
Retrieve Comments from	A customer users the Tortoise repository for their development efforts. They

Tortoise SVN	retrieve comments from Tortoise to put into the comments for their generated views. We are adding this capability for other clients who want to introduce Tortoise as their repository manager with SeETL.
Script Database Use Statements	This option is for SQL Server clients who would like SeETL to generate “use database” code as part of the generation of the SQL for views. It reduces the accidental chances of creating the views in a database that is not intended due to user error.
Script Drop View Statements	This option allows the user to turn off the “drop view” statement generation. It is intended to be able to get the correct results when the views do not exist and to assist avoid dropping views accidentally.
Generate Data Model Tables	This parameter tells the generator to read the worksheet called “Create Tables” and generate the SQL Statements for the tables into the target directory provided.
Generate SeETL SQL Using Case	This parameter tells the generator in which case to write the sql statements that it writes. The valid values are “Upper” and “Lower”.
Load Data Model Tables Documentation	This parameter tells the generator to load the SeETL Table Definitions in the worksheet called “Create Tables” into the documentation database so that it can be published and reported out to the IT Support and business community.
Data Model Table Documentation Source Database	This parameter tells the generator the name of the ***source*** database to use if the tables to be created are to be created by CTAS statements rather than using the Table Definitions in the “Create Tables” workbook.
Data Model Table Documentation Target Database	This parameter tells the generator the name of the ***target*** database to use if the tables to be created are to be created by CTAS statements rather than using the Table Definitions in the “Create Tables” workbook.
Load Businesss Requirements	This parameter tells the generator to load the Business Requirements Documentation into the documentation database so that it can be published and reported out to the IT Support and business community.
Translate Stored Procedures	This parameter tells the generator to read the worksheet called “Generate Stored Procedures” and read the stored procedures listed in the workbook and to apply the edits to those stored procedures to produce the stored procedures of the desired language to the target directories. This is a mechanism to store a stored procedure in a single text document but allow it to be delivered in any number of output languages for tatget databases.
Generate SQL DML Statements	This parameter directs the generator to generate the SQL DML statements in the SQL Generation option for SeETL. It is only turn on in the case where the user wishes to generate SQL to perform the processing for the ETL subsystem.
Generate Type 1 Dimensions	When creating the SQL Generation option it was decided to provide optional generation of portions of the SQL DML to be able to more effectively manage the deployment of the code generated. This option controls the generation of the Type 1 dimension processing.
Generate Type 2 Dimensions	This option controls the generation of the Type 2 dimension processing.
Field Sensitive Type 2 Dimensions	In the sql generation option there are two options available for the generation of sql with respect to type 2 dimensions. They can be generated with or without reference to fields to tests for type 2 changes. In the C++ engine version no such option exists. All type 2 dimensions are required to have at least one field defined as the type 2 field to check for change. In the C++ option it is generally the audit timestamp field that is checked for changed. In the C++ engine there is no overhead for this check. Hence there was no reason to create an option to not do it. However, in the sql generation version there is an overhead for checking for field changes and hence there is a reason to provide an option to NOT do it. When this field is set to ‘Yes’ then ALL type 2 dimensions are required to have a field to check for change. This option is spreadsheet wide for ALL type 2 dimensions. It can not be turned on and off for specific dimension processing.
Dim Temp Tables Based On Target	The prototype of the sql generation tool based the intermediate temp tables on the view that is created over the top of a dimension table. This was an ‘accident of development’ and was not done for any specific purpose. During deployment in early adopters it was found that it was an advantageous option to be able to model the intermediate work table on the target table itself. This

	allows the bi4all view column to have a different name to the target column. This is very helpful on the netezza platform where the views are not updateable. The ability to have different names for the target column and bi4all view column creates more flexibility in the use of names in the mapping worksheet. Prior to this feature being added it was required that the bi4all view column and the target column were the same on netezza due to the non updatable views. This restriction has now been lifted for Netezza. This option has not been required on other databases.
Generate Fact Tables	This option controls the generation of the fact table processing.
Generate Asoc Tables	This option controls the generation of the association table processing. (Note: Note yet implemented)
Generate Sequence Numbers for Dims	The C++ engine maintains and manages sequence numbers for dimension and fact tables. However, in the sql generation version these numbers must be created using the database as there is no break out to other C++ engine. The sample worksheet shows how to do this. On Netezza there are sequences and these are recommended for use when deploying with netezza. When this is option is set to 'Yes' (required for Netezza) a field will be created to maintain the sequence number which is held separately and redundantly on dimension tables to the primary key. The primary reason for doing this was to maintain complete compatibility with the C++ engine and the sql generation version. It is possible to switch between the two mechanisms with NO CHANGE to the mapping spreadsheet. This was seen as an important option in early development. It has turned out that clients have gone for one or the other. We had originally believed clients would 'mix and match' their processing as desired.
Sequence Field Suffix	This option defines the suffix for the sequence numbers generated when the above option is 'Yes'. It defaults to "_seq".
Flag Set First Character of Tables	On netezza the target table must be updated in the sql generation option. In our first implementation of the ***BI4ALL Data Models*** onto Netezza the client decided that the physical target table would be named in a pattern to the views and the field names would be meaningful. The standard that was chosen was that for a view the first character of the view would be 'V' and that for the underlying table the first character of the table would be 'T'. This is a widely adopted standard. If this parameter is set to 'Yes' the generator assumes that it must change the first letter of the 'target table' from whatever value it is to the letter that is in the next parameter.
First Character of Tables	This parameter is the value that will be substituted as the first character of the 'target table' in all the mapping rows. It is used only if the prior option is set to 'Yes'. The default value is 'T'.
Flag Set Temp Table Suffix	The temp tables have a suffix that can be added to all tables. This parameter is the directive to set a suffix for temp tables. It is not recommended to turn this directive off. It defaults to 'Yes'.
Temp Table Suffix	This is the suffix for the temp tables that are used for intermediate processing. They are named the same as the mapping name plus this suffix. The default is "_t1".
Type 1 Dimension Prefix	In the sql generation version a lot of sql script files are created to be placed into the scheduler. Since mappings may be named anything it was decided to allow prefixes and suffixes for these sql script files so that they would be more easily identified in the list of files that were produced to run the ETL subsystem. These prefixes and suffixes are all passed as directives and can be set to any character string. The default for this prefix is "dim1p_".
Type 1 Dimension Suffix	The default for this suffix is "_dim1s".
Type 2 Dimension Prefix	The default for this prefix is "dim2p_".
Type 2 Dimension Suffix	The default for this suffix is "_dim2s".
Fact Table Prefix	The default for this prefix is "factp_".
Fact Table Suffix	The default for this suffix is "_facts".
Asoc Table Prefix	The default for this prefix is "asocp_".
Asoc Table Suffix	The default for this suffix is "_asocs".
Dimension Table Lookup Suffix	The C++ engine reads dimension tables and loads them into memory mapped arrays and only ever performs one read/load of a dimension table per batch run. Because of this we have implemented very expensive views over the top

	of dimension tables to simplify ETL programming. When using the sql generation option it is obvious that these expensive views will be invoked for each fact table being processed. This is expensive. Hence, an option was introduced to load these views into lookup tables that would then be joined to the fact table in the code to perform joins. Because these tables would exist inside the same database it was required to introduce a naming standard so as to not have table name clashes. The standard for these lookup tables is "_t3". The suffix "_at" is used by the C++ engine. And there are views that have a standard of "_at2" and this name must NOT be used for this suffix. The "_at2" tables are documented at the end of this document.
Source First Qualifying Name	The sql generation version of the software can be deployed and generated in a number of configurations. The standard configuration is that there will be a staging database and a target database. Most companies do not put the staging data in the same database as the target database. Of course, if this is desired it is supported but it is very rare in production. This parameter is the database name of the source database. The first qualifier. It defaults to "SEETL3000".
Source Second Qualifying Name	This is the second qualifier that will be used in the three part qualifier for the sql generated for ETL for the source database. It currently defaults to "dbo".
Target First Qualifying Name	This parameter is the database name of the target database. The first qualifier. It defaults to "SEETL3000".
Target Second Qualifying Name	This is the second qualifier that will be used in the three part qualifier for the sql generated for ETL for the target database. It currently defaults to "dbo".
Two Step SQL Processing	The sql generation version is deployed in two major configurations. The first configuration is to have a single one step processing type of command. This is the 'normal' implementation method. It will move data with the minimum of statements and the minimum of redundancy. However, it does mean that there is a lot of cross database processing between the source and the target database and if these two databases reside on different machines this 'normal' option will require that the sql is distributed and that it will generate a lot of traffic. This was considered 'normal' in the C++ engine version as well as for the leading ETL tools. All the leading ETL tools assume that the data comes into the tool to be manipulated. Therefore having the staging database and target databases on different machines has never been 'an issue'. When using sql exclusively for ETL processing the distribution of databases onto different machines becomes an 'issue' to be dealt with. When this option is set to 'No' sql statements will be generated that move data in single statements and will perform numerous joins between the source and target databases. When this option is set to 'Yes' the sql generated makes every effort to minimise the cross database processing that will occur. Of course this can not be eliminated but it can be reduced. This option makes every effort to reduce the cross database processing when turned on.
Use Help Keys Tables	There are two configurations for deployment where 'two step processing' is turned on depending on what the client wishes. One client wished that the staging database maintained an exact copy of the data warehouse for all data that is likely to be updated and that just those records that have been inserted or changed are propagate forward. This has a number of advantages. 1. The staging area becomes a viable 'hot backup' in case of failure of the production machine. 2. The configuration lends itself to being able to split machines more easily. 3. The configuration lends itself to using database utilities for deploying data to different databases. 4. The configuration lends itself to being able to deploy copies of the data warehouse to different machines and know they are in sync. When the 'two step processing' directive is set to 'Yes' and this directive is set to 'No' the sql generator expects that an exact copy of the data warehouse data structures exists in the staging area. When this directive is set to 'Yes' the sql generator assumes that there is no copy of the target data warehouse in the staging area. This option is the 'two

	step with minimal redundant data' option. The idea is that it provides many of the advantages of the 'copy of production' but not all the advantages. In this configuration the staging database maintains copies of relationships between source system keys and the integer keys maintained by the data warehouse. These relationships are ALSO stored in the target tables. However, since the target tables are in a different database and perhaps on a different machine they are held redundantly inside the staging area in hlp_xxx_k1 tables.
Help Keys Tables Suffix	This is the suffix used for the help key tables if the directives 'two step processing' and 'Use help keys tables' are both set to 'Yes'. The default value is "_k1". It is recommended that this is not changed.
Location of Temporary Tables	This option is only effective when one step processing is used. This option gives the user the option of placing the temporary tables in the "source" database or in the "target" database. Therefore the valid values of this field are "Source" and "Target" depending on where the user would like to place the temporary tables. When configured with two step processing temporary tables can only be in the "Source" database thereby making this parameter redundant for two step processing.
Location of Control Tables	This option is only effective when one step processing is used. This option gives the user the option of placing the control tables in the "source" database or in the "target" database. Therefore the valid values of this field are "Source" and "Target" depending on where the user would like to place the control tables. When configured with two step processing control tables can only be in the "Source" database thereby making this parameter redundant for two step processing.
Not Found Dimension Key Value	When the attribution processing is performed the natural keys are translated to integer keys. There will be cases where the lookup fails and a default key is to be placed into the target fact table to say that the lookup failed. The C++ engine uses the default of "0" and this default value has been found to be viable over the last 20 years. In the sql generation option this value is set to this directive. It defaults to "0". It is not recommended that this default be changed.
Apply Not Applicable Rules	One of our clients asked us to add a feature which tells the generator that this lookup is NOT APPLPICABLE at the current point in time for this fact table. This is particularly useful for building fact tables where there are plans to bring new dimension data into the data warehouse but the fact tables are delivered prior to the time the dimension tables are available. In these cases the mappings can contain the lookups for the not applicable values but have this value emitted and placed into the fact table instead. When the lookup data becomes available the lookup is there merely 'turned on' in the mapping spreadsheet. The default value for this directive is "-99". It is not recommended that this default be changed.
Not Applicable Dimension Key Value	This is the value that will be placed into keys when the Apply Not Applicable Rules processing directive is set to Yes and the row is inficated as not applicable. The default is " -99 "
Add Comments to SQL	This directive is not yet implemented. The intention was to make commenting of the sql code 'optional'. However the client who asked for commenting of the sql code being generated suggested to us that there should be no option to turn the commenting off, otherwise programmers might just do that! The option remains in the spreadsheet and we are open to comments as to whether clients would like to have this option.
SQL Comment Start Characters	Different databases use different commenting characters in sql scripts. The usual standards are the '—' at the start of the line or /* */ combination. Therefore, we have created the option of providing the sql comment start, fill and end characters that the client might want to use. The default for this field is "--"
SQL Comment Fill Character	The value of this parameter is defaulted to "-".
SQL Comment End Characters	The value of this parameter is defaulted to "--".
View Statement Termination	Different databases and different people like different termination statements

String	for view processing. SQL Server happens to like 'go' while most others happen to like ';'. This parameter is the value that will be places into the create view processing statements as the statement termination string. It is usually 'go' or ';'. It does not affect the terminator on the sql generated for ETL processing. That is always ';'.
Place Row Limits on Dimension Tables	In the C++ engine there is a parameter called 'NumberRowsToProcess'. This parameter is used in testing to limit the number of rows to process. However, in the sql generation version this parameter is not used. Hence an option has been introduced to limit the number of rows processed in the testing of the ETL subsystem. To limit the number of rows to be processed in dimension processing for testing this parameter is set to 'Yes'.
Row Limit for Dimension Tables	This parameter is the number of rows to process in dimension table processing for testing purposes. The default is 100. It is currently only implemented for Netezza.
Place Row Limits on Fact Tables	This parameter tells the sql generator if the number of rows for fact tables should be limited for testing purposes. If it is set to 'Yes' then the number of rows processed for fact table processing will be limited.
Row Limit for Fact Tables	This parameter is the number of rows to process in fact table processing for testing purposes. The default is 100. It is currently only implemented for Netezza.
Table Number Clause for Two Step Processing	When the two directives 'Two Step Processing' is set to 'Yes' and 'Use Help Keys Tables' is set to 'No' the sql generation version assumes that there is a full copy of the data warehouse structure in the staging area and it updates this copy first. It must then move the changed rows forward to the data warehouse. To minimise the data being moved should the data warehouse architect have decided to co-habit data in physical tables the table number field is used in an equality clause. This means only those rows that MUST be moved will be moved. This table number is used in conjunction with the batch number to only move those rows that were updated in the current batch run of the ETL subsystem. If this option is set to 'Yes' an 'and' clause will be added to each statement moving data from the source database to the target database that says the table number of the tables will be equal. The field 'table_number' is then mandatory on all tables should this option be turned on. The field 'table number' is mandatory in all ***BI4ALL Data Models*** tables in any case.
Batch Number Clause for Two Step Processing	This directive tells the sql generator to place a condition on all sql statements moving data from the source database to the target database to only move those rows that have been touched in the current batch.
Generated SQL Directory	The sql generation capability must place the sql generated into a directory of the users' choice. This directive names the output directory of the sql generator for the current processing. It is recommended to be a temp directory and that the movement of code from this directory into the library structure and repository management process is handled by a person so as to give better control.
Source Views File Name	The name of the file where the staging views will be written. Please note that embedded blanks are not tested or supported.
Target Views File Name	The name of the file where the SeETL views will be written. Please note that embedded blanks are not tested or supported
Presentation Views File Name	The name of the file where the presentation views will be written. Please note that embedded blanks are not tested or supported.
Other Views File Name	The name of the file where the other views will be written. Please note that embedded blanks are not tested or supported
Compound Views File Name	The name of the file where the compound views will be written. Please note that embedded blanks are not tested or supported
Data Profiling SQL File Name	The name of the file where the Data Profiling SQL will be stored for later execution by the Data Analyst.
FQN Batch Pre-Requisites	The names of the tables used for the batch schedule are user definable. Therefore the Generator needs to be told the fully qualified name for each of the tables that are used by the scheduler. Although the control tables are not user definable we have also insisted that the user provide their fully qualified names in this worksheet as well.
FQN Process Group Pre-	Fully Qualified Name of the Process Group Pre-Requisites table in the target

Requisites	data warehouse database.
FQN Batch Commands	Fully Qualified Name of the Batch Commands table in the target data warehouse database.
FQN Mapping Spreadsheet	Fully Qualified Name of the Mapping Spreadsheet Table in the target data warehouse database.
FQN Process Commands	Fully Qualified Name of the Process Commands Table in the target data warehouse database.
FQN Last Key Used	Fully Qualified Name of the Aggregate Keys Control table in the target data warehouse database.
FQN Aggregation Control	Fully Qualified Name of the Aggregation Control table in the target data warehouse database.
FQN Dim Table Key Definitions	Fully Qualified Name of the Dimension Table Key Definitions table in the target data warehouse database.
FQN Dim Table Load Control	Fully Qualified Name of the Dimension Table Load Control table in the target data warehouse database.
FQN Dim Table Type2 Col Defs	Fully Qualified Name of the Dimension Table Type 2 Column Definitions table in the target data warehouse database.
FQN Source File Definitions	Fully Qualified Name of the Source File Definitions table in the target data warehouse database.
FQN Sequence Numbers	Fully Qualified Name of the Sequence Numbers table in the target data warehouse database.
FQN Mappings Documentation Infa	Fully Qualified Name of the Mappings Documentation Infa table in the target data warehouse database.
FQN Data Model Views Documentation	Fully Qualified Name of the root table used to load the SeETL Views Documentation. These tables are used to provide reports to users of the data model.
FQN Stored Procedure Documentation	Fully Qualified Name of the root table used to load the SeETL Stored Procedure Documentation. These tables are used to provide reports to users of the Stored Procedure Generator and the Stored Procedures themselves.
FQN Reports Documentation	Fully Qualified Name of the root table used to load the SeETL Reports Documentation. These tables are used to provide reports to users of the Reports that are in use and properly documented for the client.
FQN Project Plan	Fully Qualified Name of the root table used to load the SeETL Project Plan Documentation. These tables are used to provide reports to users of the Project Plans for projects that have been planned using the SeETL Project Planning tool.
FQN Data Model Joins Documentation	Fully Qualified Name of table used to load the Data Model Joins Documentation. These tables are used to provide reports to users of the Data Model Joins. It is also used to update Business Objects Universe.
FQN Other Views Documentation	Fully Qualified Name of the Other Views Documentation. This table is used to provide reports for users of the 'Other Views' tab in the workbook.
FQN Compound Views Documentation	Fully Qualified Name of the Compound Views Documentation. This table is used to provide reports for users of the 'Compound Views' tab in the workbook.
FQN Table Documentation	Fully Qualified Name of the Table Views Documentation. This is the root name of the tables that are used to provide reports for users of the 'Create Tables' worksheet.
FQN Input View Row Counts Documentation	Fully Qualified Name of the Create Count SQL Documentation. This table is used to provide reports for users of the 'Create Count SQL' worksheet.
FQN Runstats Documentation	Fully Qualified Name of the Create Runstats SQL Documentation. This table is used to provide reports for users of the 'Create Runstats SQL' worksheet.
FQN Create Table as Select Documentation	Fully Qualified Name of the Create Table as Select SQL Documentation. This table is used to provide reports for users of the 'Create Table as Select' worksheet.
FQN SQL for Indexes Documentation	Fully Qualified Name of the Create Index SQL Documentation. This table is used to provide reports for users of the 'Create Index' worksheet.
FQN SQL for Access Rights Documentation	Fully Qualified Name of the Create Access Permissions SQL Documentation. This table is used to provide reports for users of the 'Create Access Permission' worksheet.
FQN SQL for Storage Documentation	Fully Qualified Name of the Create Storage Object SQL Documentation. This table is used to provide reports for users of the 'Create Storage Object'

	worksheet.
FQN SQL for Generic Objects Documentation	Fully Qualified Name of the Create Generic Object SQL Documentation. This table is used to provide reports for users of the 'Create Generic Object' worksheet.
FQN SQL Statement Load Table	Fully Qualified Name of the table that SQL Statements will be loaded into when they are loaded from files into the execution database that may be different to the data warehouse database.
FQN Business Requirements	Fully Qualified Name of the root table used to load the SeETL Business Requirements Documentation. These tables are used to provide reports to users of the Business Requirements for projects that have been planned using the SeETL Business Requirements tools.
Load VM_Segmentation_Version Table	This parameter tells the Generator to process the Segmentation Version Worksheet in the workbook.
Load VM_Segmentation_Type_Code Table	This parameter tells the Generator to process the Segmentation Type Code Worksheet in the workbook.
Load VM_Segment_Code Table	This parameter tells the Generator to process the Segment Code Worksheet in the workbook.
Load VM_Segment_Code_Cols Table	This parameter tells the Generator to process the Segment Code Cols Worksheet in the workbook.
Load VM_Product_Markers Table	This parameter tells the Generator to process the Product Markers Worksheet in the workbook.
FQN VM_Segmentation_Version	Fully Qualified Name of the VM Segmentation Version Table. This is the table that the Segmentation Version Workbook is loaded into.
FQN VM_Segmentation_Type_Code	Fully Qualified Name of the VM Segmentation Type Code Table. This is the table that the Segmentation Type Code Workbook is loaded into.
FQN VM_Segment_Code	Fully Qualified Name of the VM Segment Code Table. This is the table that the Segment Code Workbook is loaded into.
FQN VM_Segment_Code_Cols	Fully Qualified Name of the VM Segment Code Cols Table. This is the table that the Segment Code Cols Workbook is loaded into.
FQN VM_Product_Markers	Fully Qualified Name of the VM Product Markers Table. This is the table that the Product Markers Workbook is loaded into.
FQN Staging Solicitation Email table	Fully Qualified Name of the Staging Solicitation Email Table. This is the table that the Solicitation Transactions Email Workbook is loaded into
FQN Staging Response Email table	Fully Qualified Name of the Staging Response Email Table. This is the table that the Response Transactions Email Workbook is loaded into.
FQN Email Address Bounce table	Fully Qualified Name of the Staging Email Bounce Table. This is the table that the Email Address Bounce Workbook is loaded into.
FQN Do Not Email table	Fully Qualified Name of the Staging Do Not Email Table. This is the table that the Do Not Email Request Workbook is loaded into.
FQN Staging Campaign Groups table	Fully Qualified Name of the Staging Campaign Groups Table. This is the table that the Campaign Groups Workbook is loaded into.
FQN Staging Campaigns table	Fully Qualified Name of the Staging Campaigns Table. This is the table that the Campaigns Workbook is loaded into.
FQN Staging Sub Campaigns table	Fully Qualified Name of the Staging Sub Campaigns Table. This is the table that the Sub Campaigns Workbook is loaded into.
FQN BI4ALL Codes table	Fully Qualified Name of the Staging Solocitation Email Table. This is the table that the BI4ALL Codes Tables Workbook is loaded into.
FQN Batch Run Log	Fully Qualified Name of the SeETL Batch Run Log Table. This is the table that records the status of batches. In this case it is used to restart batches that have partially or fully failed.
FQN Linkedin Connections table	Fully Qualified Name of the Staging LinkedIn Connections Table. This is the table that the LinkedIn Connections Workbook is loaded into.
FQN Email Connections table	Fully Qualified Name of the Staging Email Connections Table. This is the table that the Email Connections Workbook is loaded into.
TARGET ODBC Connection	This is a valid ODBC DSN that will be used by the generator to connect to the target Data Warehouse Database. The generator simply passes this string as the ODBC DSN connection string. It can connect to any ODBC DSN that is visible from the PC on which it is running.
Source Area Connection	Currently not used. However we are planning multiple staging area connections to be available in a future release.

4.1.3.1. Parameters Added to the Processing Tab for 3.1.02

This section documents the parameters added to the Processing tab for SeETL[DT] 3.1.02. We are breaking out this section of parameters as they are quite different to what has come before.

Please note that some of these parameters have been removed from 3.2.00.

Parameters added for 3.1.02	
Delta First Qualifying Name	This is the first level qualifier for the generation of delta detection SQL. It is usually the database name.
Delta Second Qualifying Name	This is the second level qualifier for the generation of delta detection SQL. It is usually the owner name for the tables involved.
Delta Table Suffix	This is the suffix added to the name of the table for which deltas are being generated and placed into. Default value: "_d1"
Minus One Table Suffix	This is the suffix added to the name of the table for which deltas are being generated and the minus one version of the data is being placed into. Default value "_m1"
Generate Delta Definitions	This is the flag that determines if SeETL will generate Delta SQL based on the Delta Processing Worksheet being present in the workbook.
Propagate Deltas to Target Table	This is the flag that determines if SeETL will generate SQL that will also migrate the deltas that have been detected to the Staging Area. The Staging table and the column names in the staging area can be named differently to the delta table and the columns in the delta table. Further, the columns that are not tested can also be moved forward to the staging area regardless of not being involved in the delta detection process.
Append Standard 4 Flags	BI4ALL has standard processing where 4 flags are added to the staging tables to allow for easier delta detection and validation processing in the staging area. These fields are by no means standard across all BI projects. Therefore SeETL has a flag available to place these 4 flags into the sql that moves the data from the delta "_d1" table to the staging table.
Delta SQL Prefix	This is a prefix that is added to the delta SQL files that are generated by SeETL. Default Value: "deltap_"
Delta SQL Suffix	This is a sufix that is added to the delta SQL files that are generated by SeETL. Default Value: "_sufdelta"
Load Delta Definitions	This is a flag that defines whether the definitions of the delta processing will be loaded into the staging area.
FQN Delta Definitions Table	This is the fully qualified name of the table that the delta definitions will be loaded into in the staging area.
Load Product Definitions	This is a flag that defines whether the product definitions for products are to be loaded into the target BI4ALL databases will be loaded into the staging area.
FQN Product Definitions Table	This is the fully qualified name of the table that the product definitions will be loaded into in the staging area.
Load List Definitions	This is a flag that defines whether the List definitions will be loaded into the staging area for later processing into the BI4ALL table.
FQN List Definitions Table	This is the fully qualified name of the table that the List definitions will be loaded into in the staging area.
Load Client Security Definitions	This is a flag that defines whether the Client Security definitions will be loaded into the staging area for later processing into the BI4ALL table.
FQN Client Security Definitions Table	This is the fully qualified name of the table that the Client Security definitions will be loaded into in the staging area.
Load Client Software Key Definitions	This is a flag that defines whether the Client Software Key definitions will be loaded into the staging area for later processing into the BI4ALL table.
FQN Client Software Key	This is the fully qualified name of the table that the Client Software Key

Definitions Table	definitions will be loaded into in the staging area.
Load Party Interests Definitions	This is a flag that defines whether the Party Interests definitions will be loaded into the staging area for later processing into the BI4ALL table.
FQN Party Interests Table	This is the fully qualified name of the table that the Party Interests definitions will be loaded into in the staging area.
SQL Statement Termination String	In SeETL 3.1.02 it is now possible to define the sql termination string for SQL statement files other than creation of views. There are some instances where it is better to use “go” for SQL Server than the more traditional “;”.

4.1.3.2. Parameters Added to the Processing Tab for 3.2.00

This section documents the parameters added to the Processing tab for SeETL[DT] 3.2.00. We are breaking out this section of parameters as they are quite different to what has come before.

Parameters added for 3.1.02	
Load Source Table Definitions	This is a flag that defines whether the Source Table Definitions will be loaded in to the SeETL control tables.
FQN Source Table Definitions Table	This is the fully qualified name of the table that the Source Table Definitions will be loaded into in the staging area.
Load Source Column Definitions	This is a flag that defines whether the Source Column Definitions will be loaded in to the SeETL control tables.
FQN Source Column Definitions Table	This is the fully qualified name of the table that the Source Column Definitions will be loaded into in the staging area.
Load Source Index Definitions	This is a flag that defines whether the Source Index Definitions will be loaded in to the SeETL control tables.
FQN Source Index Definitions Table	This is the fully qualified name of the table that the Source Index Definitions will be loaded into in the staging area.
Load Source To Stage Mapping Definitions	This is a flag that defines whether the Source To Stage Mapping Definitions will be loaded in to the SeETL control tables.
FQN Source To Stage Mapping Definitions Table	This is the fully qualified name of the table that the Source To Stage Mapping Definitions will be loaded into in the staging area.
Load Pre Mapping 01 Definitions	This is a flag that defines whether the Pre Mapping 01 Definitions will be loaded in to the SeETL control tables.
FQN Pre Mapping 01 Definitions Table	This is the fully qualified name of the table that the Pre Mapping 01 Definitions will be loaded into in the staging area.

4.1.4. Mapping01

The tab Mappings01 contains the detailed field level mappings that are used to create the views for SeETL [RT] to use in ETL processing. Virtually all mappings that need to be defined can be defined through the Mappings01 worksheet. In any specific implementation it is the completion of this worksheet that constitutes the bulk of the effort in terms of data modelling and mapping.

The Mappings01 tab contains the following fields.

Field Name	Description
Use	Tells the Generator to use the row or not. Rows with 'N' will be ignored. This cell cannot be 'empty'.
Type	The type of row is set as follows: • **M** - 'M'apping. The source column is moved to the target column with no change. • **D** - 'D'erived. The valid sql in the Source Column row will be placed into the view 'as is'. You may also place double quotes around the valid sql if the excel spreadsheet is interpreting the sql to mean something. • **F** - 'F'rom clause. The text in source column name will be passed to the database as a from clause. • **W** - 'W'here clause. The text in the source column name will be passed to the database as a where clause. • **G** - 'G'roup By - The text in the source column name will be passed to the database as a group by clause. • **O** – 'O'rder By – The text in the source column name will be passed to the database as an order by clause. Note that in from, where, group by and order by clauses the text for the sql statement goes into column name so that the new reports can use table name as a parameter for selecting data for reports.
Source Connection	Not currently used. However, we plan to use the staging name as the name of the ODBC connection defined in the processing worksheet.
Owner Name	The owner of the source table. It is used in the qualification of sql generated.
Table Name	The name of the physical table or a view of a table in the staging area. Note that in from/where/group by rows this column contains the text that will be passed to the database. This text may change from one database to another.
Table Desc	Not used by the mapping generator. It is a free format text field where you can store the description of the table for your own use if you please. The table description will be loaded into the metadata database.
Col Num	Not used by the generator. It is the number of the column in the source table.
Col Name	The name of the column in the source table.
Col Desc	Not used by the mapping generator. It is a free format text field where you can store the description of the column for your own use if you please. The column description will be loaded into the metadata database.
Use	Not used by the generator. It is a place where you can record whether the source column is to be used in the data warehouse or not.
Data Type	Not used by the generator. It is the data type of the source column. It is usually loaded into the spreadsheet so the architect can tell the data type of the source data being written to a column without having to refer to the source table directly.
Length	Not used by the generator. It is the length of the source column. It is usually loaded into the spreadsheet so the architect can tell the length of the source data being written to a column without having to refer to the source table directly.
Precision	Not used by the generator. It is the precision of the source column. It is usually loaded into the spreadsheet so the architect can tell the precision of the source data being written to a column without having to refer to the source table directly.

Field Name	Description
Scale	Not used by the generator. It is the scale of the source column. It is usually loaded into the spreadsheet so the architect can tell the scale of the source data being written to a column without having to refer to the source table directly.
Null	Not used by the generator. It indicates if the source column can be null. It is usually loaded into the spreadsheet so the architect can tell the nullability of the source data being written to a column without having to refer to the source table directly.
Reqd	Not used by the generator. It is a flag which can be filled in to say if the column is required in the data warehouse. It is similar to the previous used column.
Tbl Type	This field defines the type of the table. Valid values are: • **d1** - Type 1 dimension table • **d2** - Type 2 dimension table • **f** - Fact table • **fa** - Association fact table
Target Owner Name	Owner of the target table in the data warehouse.
SeETL View	The name of the view that will be created in both the staging area and SeETL view ddl. Note. It is recommended that this column is filled in for all columns for a SeETL view even the from, where, group by rows. This is because this column and the next column (column number) are used to sort the spreadsheet prior to running the generator.
Col Num	Not used by the generator. This is the number of the column in the SeETL View. It is recommended that this column is filled in for all columns for the view as this field used with the SeETL View column is used to sort the spreadsheet for preparation in running the generator.
SeETL View Col	The name of the column to be created in the view for both the staging area and the target SeETL view.
Target Table	The name of the physical table/view that the SeETL will write to through the view. In a 'F'rom row this column stores the sql that will be passed to the 'from' clause of the database. Note that this clause varies from database to database.
Table Exists In Target	Not used by the generator. It is a flag to indicate to the DBA performing table maintenance whether the physical table already exists in the data warehouse.
Target Column	The name of the physical column that the SeETL will write to via the view.
Col Exists In Target	Not used by the generator. It is a flag to indicate to the DBA performing table maintenance whether the physical column already exists in the data warehouse.
Key	Not used by the generator. It is a field that indicates to the DBA performing table maintenance whether this field is a part of the key for the table.
Use In PV	This column defines whether the row will be exposed in a presentation view. It allows for fields to be mapped and placed into the DW but not exposed to any presentation view.
PV Owner Name	Owner of the target table presentation views in the data warehouse.
Pres View	This is the name of the presentation view through which the data will be presented.
Col No	This is the number of the column in the presentation view. The user should note that when generating the presentation view the generator will sort the columns according to table name and this column. This allows for the ability to merge data from multiple tables into one presentation view. It also facilitates presenting data from one table in multiple presentation views. This also allows for the ability to sort columns and to present them in the order the user wants to. Please note that the limitation today is that the spreadsheet does not support presenting one column in one physical DW table through 2 presentation views. In this case you must write a separate view statement.

Field Name	Description
Pres Col Name	This is the name of the column through which the presentation view will present data.
Act Req	Not used by the generator. It is a field that can be used as a flag to note that further action is required for this row. For example, the column must be defined into the target data warehouse.
Notes 1, 2 and 3	Not used by the generator. It is a field to store free format text as notes about the mapping row or anything else that you would like to keep notes about.
UserCol 1-10	We have provided 10 user columns in case you have columns you want to add and load into the dw_mapping_ss table that are not needed in any other area of the generator.

4.1.5. Batch Schedule

The Batch Schedule tab is used to store the definition of the batch schedules that will be run periodically by the SeETL Scheduler. The SeETL scheduler is a reasonably sophisticated scheduler that can start batches based on a variety of pre-requisites and so can be used as a generic scheduler on the data warehouse machine.

The Batch Schedule tab contains the following groups and their fields.

Batch Pre-Requisites	
These rows represent the Batch Pre-Requisites. You may have as many batches as you like with as many pre-requisites per batch as you like. It is recommended that every batch have some form of pre-requisite even if it is only a file that always exists. Batch Pre-Requisites are mandatory. A batch will not execute unless it has a pre-requisite that has 'passed'.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **bpr** - Batch Pre-Requisite row. Each of these rows is written to a different table in the set of tables that manage the batch processing of the scheduler.
Batch Name	The name that the Architect chooses to give to a specific batch. It is recommended to be meaningful as it is selected from 'drop downs' in the reporting subsystem.
Pre Req Number	A batch may have many pre-requisites and they are tested in order based on this number. The Architect is advised to create pre-requisites that test in the most efficient order.
Pre Req Type	There are many types of pre-requisites for batches. They are as follows: • **FileExists** – The name of the file supplied in File Name must exist for this pre-requisite to be 'passed'. • **Daily** – The batch may run daily based on the system time being greater than the start hour and start minute for the 'daily' run. • **Weekly** – the batch may run weekly based on the day of the week being equal to 'day of week' field and the system time being greater that the start hour and start minute. • **Monthly** – the batch may run monthly based on the day of the month expressed as a number and the system time being greater than the start hour and start minute. Note. If the Architect would like a batch to run on the last day of the month then he/she should put '99' in the day number. This will be interpreted as the last day of the month no matter how many days are in the month. • **OneTime** – The batch may be started on a 'One Time' basis. That is the batch will be run exactly once if it has never run before and the system time is greater than the 'start timestamp'. This is useful for testing to make sure batches are not accidentally re-run by changing the batch number by hand.
File Name	This field contains the name of a file that must exist when used on conjunction with the Pre-Req Type of 'FileExists'. When this file exists the pre-requisite will be considered 'passed'.
Day of Week	This field defines the 'day of week' that a batch will run when used in conjunction with the Pre-Req Type of 'Weekly'. It is set to the name of the day as in 'Sunday', 'Monday', 'Tuesday' etc.
Day of Month	This field defines the 'day of month' that a batch will run when used in conjunction with the Pre-Req Type of 'Monthly'. It is set to the number of the day as in '1', '2', '3' etc. The number '99' means the last day of the month.
Start Hour	The Start Hour is the hour expressed in 24 hour format that is the start hour for the

	batch. It is used in conjunction with the Pre-Req Types of Daily, Weekly, Monthly.
Start Minute	The Start Minute is the minute within the start hour expressed as 0 to 59 that is the start minute within the start hour for the batch. It is used in conjunction with the Pre-Req Types of Daily, Weekly, Monthly.
Start Timestamp	This is a timestamp field in DD/MM/YYYY HH:MM format. It is used in conjunction with the Pre-Req Type of 'OneTime'. If the Pre-Req Type is 'OneTime' and the system time is greater than the Start TimeStamp and the batch has never been run before it will be started.
Short Description	This column contains a short description of the Batch that will be printed onto the reports for the batch processing. It is intended to assist the support personell understand the batches a little more easily.
Long Description	This column contains a long description of the Batch that will be printed onto the reports for the batch processing. It is intended to assist the support personell understand the batches a little more easily.
Processs Group Pre-Requisites	
These rows represent the Process Group Pre-Requisites. You may have as many Process Groups as you like with as many pre-requisites per Process Group as you like. It is recommended that you only create Process Group Pre-Requisites if you really need them. A Process Group can only be started if the Batch passes its pre-requisites. Process Groups are optional. It is not mandatory to create a process group for a batch.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **pgpr** - Process Group Pre-Requisite row. Each of these rows is written to a different table in the set of tables that manage the batch processing of the scheduler.
Batch Name	The name that the Architect chooses to give to a specific batch. It is recommended to be meaningful as it is selected from 'drop downs' in the reporting subsystem. The batch name must exist in the Batch Pre-Requisites Section of the workbook if it is to appear in the Process Group Pre-Requisites.
Group Name	The name that the Architect chooses to give to a specific Process Group. It is recommended to be meaningful as it is selected from 'drop downs' in the reporting subsystem. It is usually given some name like 'GROUP001' - 'GROUP999'. 'GROUP999' is usually included even if it does nothing to give a step of finalising a process group. When Process Groups are made dependent on each other or files the '999' group is used to perform that finalisation process. See the example workbook on the website to see how this works in detail.
Pre Req Number	Process Group pre-requisites are evaluated in numeric order based on this field. A pre-requisite will not be tested until all prior pre-requisites have 'passed'.
Pre-Req Type	Process Groups have only two pre-requisited types. They are: • **FileExists** – The file names in the File Name field must exist for this type of pre-requisite to be 'passed'. • **ProcessGroup** – Process Groups may be made dependent on other process groups in the batch. When this is done the Process Group named in the Pre-Req Process Group field must be completed before this Process Group will start.
File Name	Used in conjunction with the Pre-Req type of 'FileExists'. When the Pre-Req type is set to 'FileExists' then the file named in this field must exist for the Process Group to be started.
Pre Req Process Group	Used in conjunction with the Pre-Req type of 'ProcessGroup'. When the Pre-Req type is set to 'ProcessGroup' then the Process Group named in this field must have already successfully completed for this Process Group to be started.
Short Description	This column contains a short description of the Process Group that will be printed onto the reports for the batch processing. It is intended to assist the support personell understand the batches a little more easily.
Long Description	This column contains a long description of the Process Group that will be printed onto the reports for the batch processing. It is intended to assist the support personell

	understand the batches a little more easily.
Batch Commands	
This set of rows defines all the commands that will be run by the scheduler. The scheduler can run any command that can be typed into the 'system' prompt. Given that on windows and UNIX virtually all commands can be written as batch commands the scheduler allows virtually any command to be run as part of the scheduler. Commands will only run if the preceding command executed successfully. There is no intelligence return code manipulation or testing possible in the current version of the scheduler. Any non-zero return code will stop the portion of the batch dependent on the process where the error occurred with an error message. Processes not dependent on the process that failed will continue through to completion.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **bcom** – Batch Command row. Each of these rows is written to a different table in the set of tables that manage the batch processing of the scheduler.
Batch Name	The name that the Architect chooses to give to a specific batch. It is recommended to be meaningful as it is selected from 'drop downs' in the reporting subsystem. The batch name must exist in the Batch Pre-Requisites Section of the workbook if it is to appear in the Batch Commands.
Group Name	The name that the Architect chooses to give to a specific Process Group. It is recommended to be meaningful as it is selected from 'drop downs' in the reporting subsystem. It is not mandatory that the Process Group appear in the Process Group Pre-Requisites of the Workbook.
Process Step Number	Within a Process Group commands are executed in the order of the Process Step Number. The Architect sets this number in the sequence within the Batch and Process Group to run commands in sequence. Each command must complete successfully, defined by a zero return code, prior to the next command in the Process Group being started.
Process Command Name	This is a 'human readable' field to describe the command being run. It is often used to be able to construct batches of very large numbers of commands. In these cases it is given specific values in a 'working spreadsheet' and then sorted prior to the Process Step Number being assigned. In one recent example each table had to undergo 7 specific operations and we had some 150 tables that had to undergo those operations. So over 1000 commands were created. This field was used to sort those commands so that they would be executed in the desired order.
Process Program Name	This is the name of the Process or Program that will be run at this step of the batch. This name may be the name of a program/process that is visible to the Operating System when the scheduler is executed under the privileges provided or it may be a command that is stored in the Process Commands table which is loaded from the next portion of the worksheet. Fundamentally, any command visible to the userid running the scheduler that can be executed without terminal input can be executed by the SeETL Scheduler.
Process Program Parms	If a program requires parameters to be passed to it such as table names or file names this field contains those parameters.
Process Program Name Fail	Should a process/program fail the Scheduler will attempt to run the process/program defined in this cell.
Process Program Parms Fail	Should a process/program fail the Scheduler will attempt to run the process/program defined in the previous cell and it will pass the parameters defined in this cell.
Pre Requsite File Flag	This flag is set to 'Y' when the command has a pre-requisite file associated with it. If this flag is not set to 'Y' then the remainder of the line is ignored.
Pre Req Type	This field is set to 'FileExists' or 'FileNotExists' depending on whether the user wants to test for the existence or non-existence of a file to determine if the command should be executed.
File Name	This is the name of the file that will be tested for existence or non-existence when this pre-requisite is set.
Short Description	This column contains a short description of the command that will be printed onto the

	reports for the batch processing. It is intended to assist the support personell understand the batches a little more easily.
Long Description	This column contains a long description of the command that will be printed onto the reports for the batch processing. It is intended to assist the support personell understand the batches a little more easily.
Process Commands	
This set of rows defines all the commands that will be run by the scheduler. The scheduler can run any command that can be typed into the 'system' prompt. Given that on windows and UNIX virtually all commands can be written as batch commands the scheduler allows virtually any command to be run as part of the scheduler. Commands will only run if the preceding command executed successfully. There is no intelligence return code manipulation or testing possible in the current version of the scheduler. Any non-zero return code will stop the portion of the batch dependent on the process where the error occurred with an error message. Processes not dependent on the process that failed will continue through to completion.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **pcom** – Process Command row. Each of these rows is written to a different table in the set of tables that manage the batch processing of the scheduler.
PK Integer key	An integer that will uniquely identify the Process Name defined in the next cell. It is simply a shortcut for identifying Process Commands in the table.
Process Name	The Process Name. This is the name that may be entered in the 'Process Program Name' in the Batch Commands portion of the worksheet. If the 'Process Program Name' is found then the Process Command in the next cell will be executed. If it is not found then the 'Process Program Name' will be submitted to the operating system. It is important to know that the Process Name in this workbook will be used in precedence to any operating system command. Therefore Process Names should not be defined to be the same name as any operating system commands.
Process Command	The Process Command is the actual command that will be submitted to the operating system. The scheduler also allows parameter substitution and this is indicated by the symbol '?' associated with a number such as ?1, ?2. ?3. A parameter may be used a number of times when it is needed to pass the same value to a number of positions in the Process Command. Note that the Architect is limited to 9 parameters per command, ?1 to ?9. For SeETL RunTime this is the only place that commands are now required to be stored and the only place that the parameters and program names need to be understood. Because there is only one place, and very few rows that need to be defined, it is recommended only the Architect and a backup be required to learn all the parameters available in SeETL RunTime. The Process Names should be published to developers with English (or local language) descriptions of what they do and developers of schedules should only be required to know which Process Command to invoke.
Short Description	This column contains a short description of the process command that will be printed onto the reports for the batch processing. It is intended to assist the support personell understand the batches a little more easily.
Long Description	This column contains a long description of the process command that will be printed onto the reports for the batch processing. It is intended to assist the support personell understand the batches a little more easily.

4.1.6. Last Key used

The Last Key Used tab is used to store the startup value for the last key used for each dimension table. It is generally not changed once the system is running.

The Last Key Used tab contains the following fields.

Field Name	Description
Use	This column tells SeETL whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL.
UID	This is the Update/Insert/Delete indicator. It allows the Architect to be able to edit individual rows in the database via UID statements generated by SeETL.
pk_int_key	This is the primary key of the table. It is an integer.
dim_table	The name of the 'dimension table' for which this record stores the last key used. This can be any valid database table or view name. It is recommended that it is the name of a view. Examples are 'time_dim', 'customer_dim' etc.
last_key_used	This will store the last key used for a particular dimension table. You only ever enter the key at startup time for the data warehouse and it is not recommended that it is ever changed. Integer keys will be allocated sequentially for each dimension table. We prefer not to use database specific sequence keys as they can often impede unload/load activities on tables by insisting on providing new keys for records being inserted.

Example rows for this tab are as follows:

Use	UID	pk_int_key	dim_table	last_key_used
Y	I	1	time_dim	1
Y	I	2	product_dim	1
Y	I	3	customer_dim	1

These records indicate that the three dimension tables will start numbering their new records from '2'.

4.1.7. Aggregation Control

One of the major features of SeETL[RT] is the ability to generate new aggregates without any program changes, merely a parameter change in this table. This aggregation control is maintained in the Aggregate Control worksheet.

The Aggregation Control tab contains the following fields.

Field Name	Description
Use	This column tells SeETL whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL.
UID	This is the Update/Insert/Delete indicator. It allows the Architect to be able to edit individual rows in the database via UID statements generated by SeETL.
pk_aggregate_number	This is an integer primary key to this table. It is actually passed through to the summary fact table to provide the ability to partition the summary fact table by the individual summary. It can also be specified in a view to define the fact table as a separate table to end user tools such as Business Objects and Microstrategy.
run_type	The type of run for this aggregation. Summaries which you want to occur for every aggregation run of the fact table should be marked as 'always'. Aggregation levels that you want to generate just once, perhaps for special analysis, you should mark as 'oneoff'.
fact_table_name	The name of the detailed level fact table that you want the summary to be generated from. An example is f_order_fact. The summary table will be assumed to be 'f_order_fact_summary'. Thus you should choose naming standards for fact tables that allow the suffix '_summary' to be added to the table name.
number_of_dimensions	The number of dimensions in this fact table. This field is used to select the correct number of columns from the level_dimension_n fields. For example if there are 4 dimensions, only the first 4 level_dimension fields will be read as significant.
level_dimension_1	The level of the dimension to be used in this summary level. For example 0 means 'detail' level, 1 means 'level1', 2 means 'level2' etc. You have one level field for each dimension for each aggregation for the fact table. This means that the level key for that level will be used in the summary fact table. You can have many levels of aggregation and hence for the time dimension you can construct levels at daily, weekly, monthly, quarterly, annual and total levels if you choose to.
level_dimension_2	The same explanations as level_dimension_1 except this is for the second dimension on the fact table.
etc.	
level_dimension_50	The same explanations as level_dimension_1 except this is for the fiftieth dimension on the fact table.

Please note there are a maximum of 50 dimensions allowed for any single fact table.

Example rows for this tab are as follows:

Use	UI D	pk aggregate number	run type	fact table name	number of dimension s	level dimension 1	level dimension 2
Y	I	1	always	f_order_fact	2	2	1
Y	I	2	always	f_order_fact	2	5	0
Y	I	3	always	f_order_fact	2	1	5

These rows mean that the f_order_fact_summary table will contain 3 aggregate levels. There are only 2 dimensions in the fact table.

The first aggregate will summarise the first dimension table at the key level 2 and the second dimension at the key level 2.

The second aggregate will summarise the first dimension table at the key level 5 and the second dimension at the detail level key.

The third aggregate will summarise the first dimension table at the key level 1 and the second dimension at the key level 5.

How these keys get allocated will be explained below in ctl_dim_table_key_definitions.

4.1.8. Dim Table Key Definitions

One of the major features of SeETL[RT] is the ability to generate multiple levels of aggregates depending on information fed into the dimension table. These levels in the dimension tables can then be used to generate levels in the summary fact tables. In all other solutions we have ever seen, this must be done in code. A new aggregate level being made available in the dimension table means someone has to make a change to some code somewhere. No more!! The definitions of the levels that are available in a dimension table are maintained in the Dim Table key Definitions tab.

The Dim Table Key Definitions tab contains the following fields.

Field Name	Description
Use	This column tells SeETL whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL.
UID	This is the Update/Insert/Delete indicator. It allows the Architect to be able to edit individual rows in the database via UID statements generated by SeETL.
pk_int_key	This is an integer key to the table that you must enter.
dim_table_name	This is the dimension table name. For example time_dim, customer_dim, product_dim etc.
dim_type	This is the 'type' of the dimension, type 1 or 2. Valid values are 1 or 2.
agg_level	This is the aggregate level that the set of concatenated columns will represent. Valid values are 0-9. 0 means 'detail' and 9 means 'total'. In between they are known as 'level1', 'level2', 'level3' etc. Note that there is no need for the levels to actually roll up into each other. For example level 0 could be day, level 2 could be week and level 3 could be month. And weeks do not roll up into months.
num_columns_joined	This is the number of columns that are to be joined together to form the string character key for this level. If you enter a lower number than fields you want to join SeETL[RT] will only join the first n columns. If you enter a higher number than the number of column names you provide it will only join on the number of column names available. However, it is highly recommended that the number of columns joined accurately reflects the number of column names you actually want to join. The maximum is 50 joined columns.
column_names	This is the set of column names you want to join together to form the character key to be used at this level. Every unique combination of these characters will be allocated an integer key and placed into the dimension table marked as being at that level. If you join non character columns (such as integers) they will be converted to character strings and then concatenated. Note that trailing blanks in an input field will also be truncated. The concatenated string of characters is limited to 255 characters. This seems more than enough since we have never used more than 150 characters previously. The concatenation character is the '+' character. Note that a column that is concatenated as part of the key that is null is allocated the internal character string 'NULL' and this string is placed onto the dimension record. This means that there is a restriction. It is recommended that columns used for dimension table keys that can be null should not have the real value of 'NULL' as a possible value.

Example rows for this tab are as follows:

Use	UID	pk int key	dim table name	dim type	agg level	num columns joined	column names
Y	I	0	customer_dim	1	0	1	cust_code
Y	I	1	customer_dim	1	1	1	cust_household
Y	I	2	customer_dim	1	2	2	cust_sex+cust_geocode
Y	I	3	customer_dim	1	3	1	directive_none
Y	I	4	customer_dim	1	4	1	directive_none
Y	I	5	customer_dim	1	5	1	directive_none
Y	I	6	customer_dim	1	6	1	directive_none
Y	I	7	customer_dim	1	7	1	directive_none
Y	I	8	customer_dim	1	8	1	directive_none
Y	I	9	customer_dim	1	9	1	Directive_total
Y	I	10	time_dim	1	0	1	day_field
Y	I	11	time_dim	1	1	2	week_number+year_field
Y	I	12	time_dim	1	2	2	month_number+year_field
Y		13	time_dim	1	3	1	directive_none
Y	I	14	time_dim	1	4	1	directive_none
Y	I	15	time_dim	1	5	1	directive_none
Y	I	16	time_dim	1	6	1	directive_none
Y	I	17	time_dim	1	7	1	directive_none
Y	I	18	time_dim	1	8	1	directive_none
Y	I	19	time_dim	1	9	1	directive_total

The data in these rows has the following meaning.

Both customer_dim and time_dim have 1 detail level and 9 aggregate levels. You do not need to have 9 aggregate levels. However, if you want to leave spaces for possibly having more levels in the future you must enter the rows for each level you might want to have at some point in the future. SeETL^{RT} assumes that the first time it reads this table it is reading the maximum number of levels you will ever want for this dimension. Increasing the number of levels at a later date is not tested and not supported.

Aggregate level '0' means the 'detail' level.

The text 'directive_none' has a special meaning. It means that this level of aggregate is reserved for future use but is not currently implemented.

The text 'directive_total' has a special meaning. It means that this level of aggregate is the highest level of aggregate and means total for that dimension. Hence, when you want to ignore a dimension in a summary you can set it to the total level and all rows in the summary level will have the same key.

The concatenation character '+' has special meaning in that it breaks the column names. The code looks at the '+' sign in this table as the delimiter of the field names. Naturally '+' is not allowed as a character in the table column name.

The concatenation of week_number+year_field means that when the dimension table is loaded these two fields will be joined together and the dimension table will be searched to see if this combination of week and year has ever been loaded. If not, it will allocate a key and insert the record thus creating a unique character string for week level and an integer key that can be used in the fact table to join to the dimension table record.

This processing is for type 1 dimensions and for the keys for type 2 dimensions. However, type 2 dimension tables have a further requirement to close out records and insert new records when a field that is not necessarily part of the key changes. Naturally, any field that changes in the key will insert a new record if it does not exist. To handle type 2 dimension tables another control table is required. It is discussed next.

4.1.9. Dim Table Type2 Col Defs

To support type 2 dimension table changes we must record the combination of fields that must be checked for change to determine if the existing dimension table record should be closed out and a new type 2 record should be inserted. The Dim Table Type 2 Col Defs tab supports this requirement.

The Dim Table Type 2 Col Defs tab contains the following fields.

Field Name	Description
Use	This column tells SeETL whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL.
UID	This is the Update/Insert/Delete indicator. It allows the Architect to be able to edit individual rows in the database via UID statements generated by SeETL.
pk_int_key	This is an integer key to the table that you must enter.
dim_table_name	This is the dimension table name. For example time_dim, customer_dim, product_dim etc.
num_columns_to_test	This is the number of columns that are to be tested to see if they have changed when compared to the current dimension table record. There is a limit of 50 fields.
column_names	This is the set of column names you want to test for change. Each column's data value on the incoming record will be compared with the value on the current record for the dimension. The comparison is null sensitive so if the values are both equal or they are both null the record will be considered not to have changed. The delimiter is again the '+' character.

An example row for this tab is as follows:

Use	UID	pk int key	dim table name	num columns joined	column names
Y	I	1	customer_dim2	3	cust_addr_1+cust_addr_2+cust_phone

The data in this row has the following meaning.

The customer dimension 2 is a type 2 dimension table. The fields to check (at the detailed level only) for change are cust_addr_1, cust_addr_2 and cust_phone. If any of these fields have changed then close out the current dimension record and open a new dimension record. The source of the dates for record closure and opening is discussed below.

4.1.10.Dim Table Load Control

Fact table attribution processes have a 'need for speed'. There are two Fact Table Attribution programs provided with SeETL[RT]. One is the standard processing program and one is for special processing required for the Sybase IWS data model. The attribution process can consume very large amounts of processing power and run for very long elapsed times.

One of the standard ways to speed it up is to trade memory for speed. That is, load the dimension tables into memory and search them there rather than read them through the database which is at least 10 times slower per read. This loading of dimension tables into memory is facilitated by the Dim Table Load Control tab.

The Dim table Load Control tab contains the following fields.

Field Name	Description
Use	This column tells SeETL whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL.
UID	This is the Update/Insert/Delete indicator. It allows the Architect to be able to edit individual rows in the database via UID statements generated by SeETL.
pk_int_key	This is the integer key to this record.
fact_table_name	This is the name of the fact table for which this record specifies the dimension table load option. **(2.1 Enhancement).** The fact table name can also be set to '*ALL*' to specify that the dimension table named in dim_table_name should be loaded as specified in the load_control_option field for all fact tables.
dim_table_name	This is the name of the dimension table for which this record specifies the dimension table load option. The reason that both the fact table and the dimension table are specified is so that you have the option of loading dimension tables into memory for specific fact tables. The larger fact tables will benefit more from loading dimension tables into memory.
load_control_option	Determines whether the dimension table will be loaded into memory and searched using a binary search or read from the database. Valid values are: 1. Read the table from the database. 2. Read the table from a 100 element array. If the row is not found then read it and put it into the array. (Note. This option operates exactly as Option 1 at this time. With memory mapped IO implemented this option may not be implemented in the future.) 3. Load the table into memory and read the dimension table from memory using a binary search. 4. Load the table into a memory mapped file backed by the system paging file for windows 2000 and by /tmp files on UNIX. If a dimension table is loaded into a memory mapped for by one attribution process (AT01/2) it is available for access by all attribution processes. Memory mapped IO has been added to provide 'unlimited scalability'. That is, large volumes of memory can be used to load one single copy of as many dimension tables as the user would like and these images are shared across as many processors as the user would like to use.
dimension_table_group	**(2.1 Enhancement).** This field can be defined to be any string, though we do recommend 'GROUP1', 'GROUP2', GROUPn' etc. With the introduction of memory mapped IO we also introduced the ability to run many instances of CTLU012 at the same time and to load many dimension tables at one time. This is made possible by specifying groups of dimension tables to load into memory mapped files. The dimension tables that are to be loaded in a single group are defined using this field. They are loaded by CTLU012 using the `LoadDimensionTableGroup` parameter. By using this field it is possible for the user to gain the maximum amount of control over the loading of dimension tables into memory mapped files enabling the most efficient use of the machine resources to achieve maximum throughput of the overall

	batch while using the minimum amount of real memory to maintain copies of the dimension tables in the memory of the machine.

Example rows for this table are as follows:

Use	UID	pk int key	fact table name	dim table name	load control option
Y	I	0	f_order_fact	customer_dim	1
Y	I	1	f_order_fact	time_dim	2
Y	I	2	f_order_fact	product_dim	3
Y	I	3	f_order_fact	product_dim	4

The data in these rows has the following meaning.

For the f_order_fact attribution process CTLAT01/02 will:

1. Read customer_dim and time_dim from the database

2. Load product_dim into in an in memory array and binary search the array when looking up product keys.

3. Load vendor_dim into a memory mapped in memory array and binary search that array. The vendor_dim table will also be available for any other instance of CTLAT01/02 to share while product_dim is private to the CTLAT01/02 process that loaded it.

4. Note that no record in this table for a fact table/dimension table combination is equivalent to specifying a load control option of 1, read the table from the database.

See the SeETL [RT] manual for the detailed implications of the memory allocations.

4.1.11. Dim Table Key Fields

The Dim Table Key Fields tab is used to store the start-up value for the last key used for each dimension table. It is generally not changed once the system is running.

The Dim Table Key Fields tab contains the following fields.

Field Name	Description
Use	This column tells SeETL whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL.
Dimension Table Name	This is the name of the dimension table for which the Architect would like to provide an alternate name for the primary key.
Key Field Name	SeETL recommends that the real primary key for a dimension table is called pk_dim_table_name. However, this is not always the case for our clients. If you have a primary key naming standard that is not pk_dim_table_name for your dimension tables you may use this worksheet to define the alternative primary key names for your dimension tables. You only require entries in this table for dimension tables that do not follow the standard recommended. This field contains the name of the primary key field name for the dimension table named in the previous cell.

4.1.12.Source File Definitions

The Source File Definitions tab is used to store the start-up value for the last key used for each dimension table. It is generally not changed once the system is running.

The Source File Definitions tab contains the following fields.

Field Name	Description
Use	This column tells SeETL whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL.
Pk Int Key	The PK Int Key is a sequence number used as the primary key to the table.
Source Table Name	If it is a table the source full table name
Source Table Prefix	If it is a table the source prefix (schema)
Source Table Suffix	If it is a table the source table name without prefix
File Name	If it is a file the name of the actual file
Status	This is the status of the file or table in the process of putting it into the target data warehouse. It is recommended that you select statuses that are ascending so that the spreadsheets that can be downloaded from the SeETL Reports can be 'drilled' on Status with a customer condition of Status > 'desired status'. It is also recommended that a status be included for files that do not make it into the scope of the project as they may make it into the scope of later projects. Proposed status might be: • 00 – Discovery. • 01 – Evaluation • 03 – Out of Scope • 04 – In Scope • 05 – Staging Table defined • 06 – ODS table defined • 07 – EDW table defined • 08 – SRC to STG ETL complete • 09 – STG to ODS ETL complete • 10 – ODS to EDW ETL complete • 11 – Integration testing • 12 – System testing • 13 – In production
Source System	The name of the source system from which the file/table comes.
Source Customer/Organisation	If the file is coming from an external customer/organisation then the name of the customer/organisation.
File Descriptive Name	The name of the file that is used in discussions or documentation. For example, 'Sell Through from Tesco'
Description	Enter a description of the file.
Sender/Creator/Contact	The name of the person who is the contact for sending/creating the file/table.
Our Company Data Owner	Name and Role of the person in our company who is responsible for the relationship with the source organisation for the provision of the file. This can be NA (not applicable) if it is from Internal Systems.
Our Company Tech Support	The name and role of our IT person who is responsible for the timely and effective processing of this file.
Granularity/Level	Some statement of the level of granularity of the data that is in the file. This is for descriptive purposes only. • SKU? • Store? • Reseller? • Account? • CDR? • Transaction • Combination of the above?
Frequency of Transfer	Is the file: • Near Real time? • Hourly? • Daily?

	• Weekly? • Monthly? • Ad-hoc? • Any other?
Approx Volume	Is it 1MB, 10MB, 100MB, 1GB, 10GB or more?
Transfer Mechanism	Is the file sent through: • EDI? • Ftp? • Email?
File Format	Is the file: • Excel? • Delimited text? • Fixed format? • EDI interchange format? • XML?
# of files	Number of files that are frequently transferred. In some cases a 'file' is actually a collection of files that must be processed together and if that number is more than one it should be recorded here.
Loaded to system today	If the data is loaded to a system today then the current system where the information is loaded to.
Reviewed by	The name of the person who reviewed this file to determine if it was really required in the EDW.
Required by	The name of the person who said that the file should be in scope for the EDW. Note that though a person says they believe a file should be in scope for the EDW that file should still be reviewed and checked to determine that: 1. The data is not already loaded into the EDW. 2. There is value in loading the data into the EDW.
Notes1..Notes5	There are five cells for notes for each file that can be completed at the Architect's discretion.

4.1.13. Sequence Numbers

The Sequence Numbers tab is used to store sequence names and numbers that are made available to SeETL^{RT} for such things as sequences for fact tables. Note that this table does not maintain the sequences for dimension tables.

In SeETL 3.0 the ability to generate sequence numbers at the front of ordinary fact tables was added as a new 'feature' and this table was introduced to support that feature.

The Sequence Numbers tab contains the following fields.

Field Name	Description
Use	This column tells SeETL whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL.
Primary Key Int	The PK Int Key is a sequence number used as the primary key to the table.
Sequence Name	This is the name of the sequence that is to be defined in this table. When using sequences to define sequences for fact tables it is recommended that the sequence is named the same as the view name that is processed to use the fact table. This helps the Architect remember what sequences are used for what processing.
Last Key Allocated	This is the initialisation number for the sequence. It should only ever be set once and loaded once. After loading of a sequence SeETL will maintain the sequence in the ctl_sequence_nums tables.
Sequence Group Size	This is the number of sequence numbers that will be reserved every time SeETL^{RT} requests a new set of sequence numbers. It is made available so that the Architect can decide how large the group of sequence numbers is to provide maximum parallelism for the processing of fact tables.
Sequence Locked	This field is used only at run time and should always be set to 'N' in this cell. It tells SeETL^{RT} that the process that is retrieving a new set of sequence numbers is in the middle of that process and that no other processes should try and allocate sequence numbers until the row is unlocked.

4.1.14.Mappings Documentation Infa

The Mappings Documentation Infa tab is used to store information specific to generating the Informatica Programmer Specifications documentation that a client using SeETL[DT] and Informatica might like to generate.

Because SeETL is free when used as a prototyping tool we have a number of clients who use SeETL to build their prototype EDW and when they are happy with it they use Informatica to write the 'real ETL' as they have made a long term commitment to building all production ETL in Informatica.

Some of these clients have asked us to assist in the process of converting the SeETL[DT] workbook to Informatica Programmer Program Specifications. Therefore, this sheet contains all the 'extra' information required to produce such Program Specifications.

The Mappings Documentation Infa tab contains the following fields.

Field Name	Description
Use	This column tells SeETL whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL.
PK Int Key	The PK Int Key is a sequence number used as the primary key to the table.
Mapping Name	This is the name of the Informatica mapping to be produced.
Mapping Dependencies	This lists the dependencies of this mapping. That is, this lists the mappings that must be run prior to this mapping running or, if there are many, the groups of mappings that must be run before this mapping.
View Name	The view name provided in this cell provides the 'link' to the Mappings01 workbook. The view name is used to look into the table storing the Mappings01 workbook and select out the rows that are mapped using this view name.
Target Table Name	This is the name of the target table for the mapping. It is generally expected that a mapping has one target table, and when it has more than one targer table it is expected there is a 'main' target table.
Description	This is a description of the target table.
Dev Order	This is a field which can be used to document the development order of mappings. It can be used to sort results of queries against the metadata database so that the Architect can define the development order and then see progress against that order.
Dev Priority	This is a field where the development priority of a mapping can be set.
Dev Status	This is a field where the development status of a mapping can be set. It is recommended that the description used be prefixed with numerics so that it can be queried using custom in the Excel spreadsheet reports provided with SeETL. It is recommended that some sequence of statuses be developed so that the reporting spreadsheet can be drilled by Status. A suggestion is: • 00 – Unallocated • 01 – Allocated to Programmer • 02 – In Development • 03 – Unit Testing • 04 – Walkthrough Passed • 05 – Integration Testing • 06 – System Testing • 07 – In Production
Dev Assigned	This is the name of the developer to whom the development of this mapping has been assigned.
Estimated Hours	This is the estimate number of hours to complete this mapping.
Audit Controls Reqd	This field defines whether audit control are required in this mapping. There is a separate report that documents Audit Fields and it uses this cell to constrain that report. If this field is 'Yes' then Audit controls are required.
Audit Fields	If the prior cell is set to 'Yes' then this field lists the fields/columns on the file/table that require auditing in the Informatica ETL processing.
Refresh methodology	This field documents the refresh strategy used by this mapping. In particular, is the target table to be 'insert/update' or 'truncate/load'.
Record Selection Criteria	This is an English Language description of how the developer should select rows from the source file/table.

Condition in SQ	This is the encoding that will be required in the Informatica SQ to select the source records/files from the input data.
Source Table(s)	This field contains the names of the tables to be read as input for this mapping.
Unique Key	This field contains the names of the fields which constitute the unique key that will be used to determine whether the row in the target table should be 'insert' or 'update'. It is not the primary key on the target table which is very likely a sequence.
Parameter01..05	These 5 fields contain documentation of parameter values that may be set in the mapping.
Variable01..05	These 5 fields contain documentation of variable values that may be set in the mapping.
Source Connection	This field contains the definition of the source connection that should be used by the mapping to connect to its input data.
Target Connection	This field contains the definition of the target connection that should be used by the mapping to connect to its output tables.
Error Log DB Connection	This field contains the definition of the target connection that should be used by the mapping to connect to its Error Log database.
Default Fields	If some fields are set to defaults then this cell documents the names and values of those defaults.
Session Names	This field documents the session name to be used for this mapping.
Workflow Names	This field documents the workflow name which will run this mapping.
Mapping Logic Comments	Where there is some complex logic in the mapping that needs to be further defined and documented this field is available to document such mapping logic comments.
Lookup Notes	Where there is some complexity in the lookups to be performed in the mapping the 'Lookup Notes' may be used to document these complexities for the programmer.
Data Flows	Where there is some complexity in the Data Flows to be performed in the mapping the 'Data Flows' may be used to document these complexities for the programmer.
Notes1...10	There is room for up to 10 notes to be placed into the Program Specification for the Informatics Programmer.

4.1.15.Other Views

The Other Views tab is used to store definitions of views that the Architect might like to define and which are not generated by SeETL[DT] in some other way.

In every implementation it has happened that there are some views which get created to make some portion of work that little bit easier. These views have normally just been created in text documents. However, in text documents they are not particularly 'visible' and this broke with the concept of 'You and see your ETL' which is a design point, and the basis for the name, of SeETL.

It was decided to make it possible to put all 'Other Views' into the SeETL Workbook and publish them via the portal just to 'close the gap' on having all portions available. The Architect can decide for him/her self whether he/she will use these other views for this style of documentation.

The Other Views tab contains the following fields.

Field Name	Description
Use Ind	This column tells SeETL whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL.
Type	The type of the field being defined. It may be: • 'M' – Move • 'F – From • 'W' – Where • 'G' – Group By • 'O' – Order by
Table owner name	This is the owner of the table over which the view is being placed.
Table name	This is the name of the table over which the view is being placed.
Col name	This is the name of the column in the table over which the view is being placed.
View owner name	This is the name of the owner of the view that is being created. It is not necessarily the same as the name of the owner of the table.
View name	This is the name of the view being created.
View col	This is the name of the column being created in the view.

4.1.16.Dim Table Current Flags

A client found a good use for being able to morph dimension tables between type 1 and type 2 processing. To enable the client to be able to do this we added this tab and we added the processing to SeETL to all the generation of a dimension maintenance view which would include a constraint of 'current_flag = 1'.

This is a very simple tab. If you would like to know more about how it works simply place and entry in this column and check the views that are generated.

The Dim Table Current Flags tab contains the following fields.

Field Name	Description
Use Ind	This column tells SeETL whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL.
Dimension Table Name	The name of the 'dimension table' for which a maintenance view will be created with a 'current_flag = 1' constraint on it.

4.1.17.Dim Table Levels

Those who have known Mr. Nolan for a while know that he has been an advocate of the multi-level dimensional modeling since he was taught the same at Metaphor back in 1993. Alas, this method of modeling has been largely ignored over more recent times.

However, in this release we have decided to include the ability to automatically generate the views requried for processing multi-level dimensional models. SeETL Runtime has always been inherently multi-level and so there were no changes required

The Dim Table Levels defines how many levels of aggregation will be defined for the dimension table. In years gone past we have always built dimension tables with 9 levels of aggregation plus the detailed level. With SeETL we have made it variable by dimension table to give the user a little more flexibility in how many levels are created. The maximum is still 9 levels of aggregation per dimension table.

The Dim Table Levels tab contains the following fields.

Field Name	Description
Use Ind	This column tells SeETL whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL.
Dimension Table Name	The name of the 'dimension table' for which a maintenance view will be created with a 'current_flag = 1' constraint on it.
Number of Levels	This is a numeric value between 1 and 9 inclusive to define the number of aggregate levels that are defined for a dimension table. If the user does not want to create levels then the row should be omitted.

4.1.18. Update BO Universe

The Update BO Universe tab is used to store definitions of fields that are exposed using a Business Objects Universe.

We do a lot of implementation where Business Objects is used to access the data warehouse. In these sites there is always a lot of work required to maintain the Business Objects Universes, especially the business names and help text of the fields in the Business Objects Universe.

So we decided to include a piece of code we wrote many years ago into SeETL[DT] which will assist Business Objects clients to maintain their Business Names and Help Text in an Excel spreadsheet and then directly update their Business Objects Universes.

The storage of this tab like this has an interesting implication.

If the Architect would like to maintain many Business Objects Universes in many Languages he/she simply needs to maintain N copies of this worksheet inside the workbook. For example Update BO Universe EN, Update BO Universe ES, Update BO Universe DE, Update BO Universe FR etc for English, Spanish, German and French. Then, when wanting to convert the base BO universe from the base language the Architect can simply copy (or rename) the language specific tab to the 'Update BO Universe' tab and perform the BO Universe update. The target universe will have its objects named accordingly.

By using this type of capability we have found that it is possible to maintain a single source BO Universe in some base language and then to generate the N universes in the N languages required.

Interestingly, we are also able to maintain just one base set of reports and convert them to the N languages they are required in without having to manually maintain the N sets of reports.

As far as we are aware we are the only people in the world able to do this. This might be a very nice 'freebie' to the Business Objects community.

The Update BO Universe tab contains the following fields.

Field Name	Description
Use Ind	This column tells SeETL whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL.
Table Name	This is the name of the table that the BO Object is defined over.
Column Name	This is the name of the column that the BO Object is defined over.
Object Name	This is the Business Name the Architect would like to give to the object.
Help text	This is the help text that the Architect would like to give to the object.

Note 1.
If a field is defined as a 'sum' of a column then the object name and help text will be applied.

Specifically, If the first 4 characters of a field are 'sum(' or 'SUM(' then the 4 characters will be removed along with the last character of the Business Objects internal field name and then what remains of the field will be compared with "Table Name.Column Name" and where they match the Object Name and Help Text may be updated.

Note 2.
The workbook does not contain an entry for the name of the file for the Business Objects Universe. Because Business Objects Universes are security controlled it is necessary to start the Designer as an application and allow the Architect to log onto the Business Objects Designer application and then specify the universe to be updated. In this way, the designer defines the target Business Objects Universe to be updated. When creating 'copies' of universes to then be altered to other languages the Architect must create the copy prior to using SeETL[DT] to alter the language of the objects in the universe.

4.1.19.Translate RS Reports

BI4ALL is a new poduct that we are working on. Currently there are 70+ MicroSoft Report Services Reports in the product. We have developed the ability to translate a base report (written in English) to up to 10 other languages. For example, we have already completely translated all reports to German and we are able to re-run the translation process at will.

This is a huge breakthrough because it means that the translation of reports between languages is automatic. It also means there is only one base set of reports to maintain.

We have already integrated some new features into SeETL DesignTime ready for the release of BI4ALL. This ability to translate reports into many languages is the first of a number of features we will be adding to SeETL Design Time in order to support BI4ALL.

The Translate RS Reports tab that supports the translation of BI4ALL Reports is relatively complex because of the amount of information that is required to be captured.

The Translate RS Reports tab contains the following fields.

Number of Languages	
The first row of the spreadsheet defines the number of languages the base report is to be translated into.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **nl** – Number of Languages.
Number of Languages	This field defines the number of languages that the base reports will be translated to. It must be a numeric value between 1 and 10 inclusive.
Input Output Definitions	
This is a single row that defines the source directory in which the source reports can be found. The remaining columns represent the target directories that the translated reports will be written to.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **dir** – Directory Name for source and target directories.
Source Directory Name	This is the name of the directory where the reports are stored as source rdl documents.
Target Directory Lang 01	This is the target directory where the translated reports will be written for language 01.
Target Directory Lang 02	This is the target directory where the translated reports will be written for language 02.
Target Directory Lang 03	This is the target directory where the translated reports will be written for language 03.
Target Directory Lang 04	This is the target directory where the translated reports will be written for language 04.
Target Directory Lang 05	This is the target directory where the translated reports will be written for language 05.
Target Directory Lang 06	This is the target directory where the translated reports will be written for language 06.
Target Directory Lang 07	This is the target directory where the translated reports will be written for language 07.
Target Directory Lang 08	This is the target directory where the translated reports will be written for language 08.
Target Directory Lang 09	This is the target directory where the translated reports will be written for language 09.
Target Directory Lang 10	This is the target directory where the translated reports will be written for language 10.
Report Name to File Definitions	

One of the more difficult things to achieve is the translation of the actual report names. Because the report names are used to present the report to the user the report name and file name must also be changed during the translation process.

In order for facilitate this translation we have implemented a definition of a 'report name' which is just an abstracted name that is not used anywhere else other than this worksheet.

We have then defined the source file related to that report name as well as the 10 target file names that the report name will be converted to when written out in that language. Note that the value used in the target file name fields may also be used inside the reports for links between reports.

Field Name	Description
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **rnfd** – Report Name File Definition
Report Name	The Report Name is any name up to 255 characters that is a unique name for the report within the workbook. It is only used inside the workbook to link the name of the physical files in which the RDL is stored with the actual translation rows. It is possible to apply a translation to a single report or to all reports documented in the workbook.
Source File Name	The Source File Name is the physical name of the file in which a reports RDL will be found. It is connected to the Source Directory Name as above. The source file will be read for rdl and it will be converted to the various languages and placed into the Target File Names below.
Target FileName Lang01	Each report is converted to between 1 and 10 languages inclusing. Therefore each report must have up to 10 output files defined so that the new reports can be written to these output files.
Target FileName Lang02	This is the file that the language 02 will be written to.
Target FileName Lang03	This is the file that the language 03 will be written to.
Target FileName Lang04	This is the file that the language 04 will be written to.
Target FileName Lang05	This is the file that the language 05 will be written to.
Target FileName Lang06	This is the file that the language 06 will be written to.
Target FileName Lang07	This is the file that the language 07 will be written to.
Target FileName Lang08	This is the file that the language 08 will be written to.
Target FileName Lang09	This is the file that the language 09 will be written to.
Target FileName Lang10	This is the file that the language 10 will be written to.

Translation Row

The translation row is the key row of the workbook. Each row of the translation rows defines a translation of a source string to a target string which is implemented as a change command inside the processing.

The way that this works is that the user defines a string in the source report RDL that is required to be translated, and then the n translations for that string.

It is recommended that the tags around the string are also specified in the Source String to reduce the possibility of translating fields un-intentionally. The string that is defined in the Source String will be translated and so the user must be careful to make it unique.

Field Name	Description
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **tr** – Translation row.

	Each of these rows is written to a different table in the set of tables that manage the batch processing of the scheduler.
Report Name	The Report Name is any name up to 255 characters that is a unique name for the report within the workbook. It is only used inside the workbook to link the name of the physical files in which the RDL is stored with the actual translation rows. It is possible to apply a translation to a single report or to all reports documented in the workbook.
Source String	This is the string that is to be converted from one language to another. It is recommended that the RDL tags are included in the source string.
Target String Lang01	This is the string that the Source String will be changed to for the language 01.
Target String Lang02	This is the string that the Source String will be changed to for the language 02.
Target String Lang03	This is the string that the Source String will be changed to for the language 03.
Target String Lang04	This is the string that the Source String will be changed to for the language 04.
Target String Lang05	This is the string that the Source String will be changed to for the language 05.
Target String Lang06	This is the string that the Source String will be changed to for the language 06.
Target String Lang07	This is the string that the Source String will be changed to for the language 07.
Target String Lang08	This is the string that the Source String will be changed to for the language 08.
Target String Lang09	This is the string that the Source String will be changed to for the language 09.
Target String Lang10	This is the string that the Source String will be changed to for the language 10.

4.1.20.Segmentation Type Code

The SeETL[RT] Segmentation Engine requires a number of tables to be created and made available to present the data to the end user correctly and with the correct descriptions while allowing those descriptions to be flexible. It was decided to take this approach to more easily allow for multi-language and highly customizable text to be presented onto web pages and reports. All efforts have been made to make the reporting engines and the segmentation engines as flexible and robust as possible.

The following tables were all created to support this goal of being more flexible and supporting more languages as much as is possible inside a relatively simply architecture and modest development budget.

The Segmentation Type Codes allow the user to define the text of the types of Segmentation that will be implemented in this instance of the Segmentation Engine.

The Segmentation Type Code tab contains the following fields.

Field Name	Description
Use Ind	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
UID	This is the Update/Insert/Delete indicator. It allows the Architect to be able to edit individual rows in the database via UID statements generated by SeETL[DT].
pk_int_key	This is an integer key to the table that you must enter.
Segmentation Type Short Description	This is a short description of the segmentation type that will be used on reports and screens. It is intended to be short enough to make a good column or row header. This is recommended to be no longer than 15 characters.
Segmentation Type Long Description	This is a long description of the segmentation type that will be used to explain what the Segmentation Type is relatively briefly. It is NOT used on reports.
Segmentation Type Help Description	This is the help description of the segmentation type that will be used to explain what the Segmentation Type is in as much detail as possible. It is presented to users for education and learning as they become familiar with the segmentation engine.
Start Date	This is the start date of the effectiveness of the row.
End Date	This is the end date of the effectiveness of the row.
Current Flag	This is a flag that indicates if the row is current or not.
Code name	This is the string 'vm_segmentation_type_code' for this table.
level col	This is set to 'detail' and must never be changed.
dim char ky fld	This is the dimension table character key and is set by the system architecture to be unique for this dimension table within level column.
Row Del From Source	This is set to 'Y' or 'N' depending whether the row is considered to be deleted from the source system or not.
Batch Number	This is set to the batch number in which the row was processed in the normal course of events. In the workbook it must be set to some value as determined to be the proper value for this installation by the Architect.
File Cycle Number	This is set to the file cycle number in which the row was processed in the normal course of events. In the workbook it must be set to some value as determined to be the proper value for this installation by the Architect.
View Number	This is set to the number allocated to the etl processing being performed in the normal course of events. In the workbook it must be set to some value as determined to be the proper value for this installation by the Architect.
Audit Timestamp	This is usually set to the system time of the row being extracted out of the staging area in the normal course of events. In the workbook it must be set to some value as determined to be the proper value for this installation by the Architect.

An example of the contents of this worksheet is as follows:

Use	UID	Key	Segmentation Type Short Description	Segmentation Type Long Description	Segmentation Type Help Description	Start Date	End Date
Y	I	1	Socio Economic	Social Group Segmentation	This is the type of socio economic coding that is made available by applications like Prizm.	02-Jan-2000	31-Dec-2999
Y	I	2	Demographic	Demographic Segmentation	This is the type of segmentation based purely on demographic factors.	02-Jan-2000	31-Dec-2999
Y	I	3	PCS	Product Catalogue Segmentation	This is the type of segmentation that is based on using the product usage or purchase patterns to infer the customer fits into one or more product catalogue segments.	02-Jan-2000	31-Dec-2999
Y	I	4	Strength	Strength Segmentation	This is the type of segmentation that is based on using strength and value of the relationships to place customers on a grid.	02-Jan-2000	31-Dec-2999

4.1.21.Segmentation Version

The Segmentation Version table allows the user to define the versions of the Segmentation Models that are implemented in this instance of the Segmentation Engine.

The Segmentation Version tab contains the following fields.

Field Name	Description
Use Ind	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
UID	This is the Update/Insert/Delete indicator. It allows the Architect to be able to edit individual rows in the database via UID statements generated by SeETL[DT].
pk_int_key	This is an integer key to the table that you must enter.
Segmentation Type Code	This is a code for the segmentation type that will be used on reports and screens. It is intended to be short enough to make a good column or row header. This is recommended to be no longer than 5 characters.
Segmentation Type Short Description	This is a short description of the segmentation type that will be used on reports and screens. It is intended to be short enough to make a good column or row header. This is recommended to be no longer than 15 characters.
Segmentation Type Long Description	This is a long description of the segmentation type that will be used to explain what the Segmentation Type is relatively briefly. It is NOT used on reports.
Segmentation Type Help Description	This is the help description of the segmentation type that will be used to explain what the Segmentation Type is in as much detail as possible. It is presented to users for education and learning as they become familiar with the segmentation engine.
Segmentation Version Number	This is the version number of the segmentation model that this record refers to. It is stored as an integer in this field.
Segmentation Version Number String	This is the version number of the segmentation model that this record refers to. It is stored as a character string in this field. The character string column is used to perform selects against inside stored procedures. Though it is possible to convert the integer to a char in the query for the select it was decided as a design point that ALL parameters passed to stored procedures would be character strings and that all constraints against databases would be character strings.
Start Date	This is the start date of the effectiveness of the row.
End Date	This is the end date of the effectiveness of the row.
Current Flag	This is a flag that indicates if the row is current or not.
Code name	This is the string 'vm_segmentation_type_code' for this table.
level col	This is set to 'detail' and must never be changed.
dim char ky fld	This is the dimension table character key and is set by the system architecture to be unique for this dimension table within level column.
Row Del From Source	This is set to 'Y' or 'N' depending whether the row is considered to be deleted from the source system or not.
Batch Number	This is set to the batch number in which the row was processed in the normal course of events. In the workbook it must be set to some value as determined to be the proper value for this installation by the Architect.
File Cycle Number	This is set to the file cycle number in which the row was processed in the normal course of events. In the workbook it must be set to some value as determined to be the proper value for this installation by the Architect.
View Number	This is set to the number allocated to the etl processing being performed in the normal course of events. In the workbook it must be set to some value as determined to be the proper value for this installation by the Architect.
Audit Timestamp	This is usually set to the system time of the row being extracted out of the staging area in the normal course of events. In the workbook it must be set to some value as determined to be the proper value for this installation by the Architect.

Refer to the sample spreadsheet for an example of the data in this worksheet.

4.1.22.Segment Codes

The SeETL[RT] Segmentation Engine allows the user to define the text that will appear on reports for the segment codes. This tab contains the segment codes and their proper descriptions to be placed onto reports. This set of rows is the text that will be used for segment codes where the text is placed as row headings in a column down the side of reports. Therefore the option was created to have different text for the same segment codes depending on whether the description was to be used in row headings or column headings.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
UID	This is the Update/Insert/Delete indicator. It allows the Architect to be able to edit individual rows in the database via UID statements generated by SeETL[DT].
Key	This is an integer key to the table that you must enter.
Segmentation Type Sdesc	This is the same segmentation type short description used in the above table to link the segment codes to the segmentation types.
Segment Code	This is a number from 0 to 64. There are a maximum of 64 different segments so that they can be placed onto a chess board. The 0 segment is taken to be the 'summary' segment or the 'totals' segment.
Segment Short Description	This is the short description of the segment name that will be used on reports where the segment description is used as a row header.
Segment Long Description	This is the long description of the segment name that will be used to explain briefly what the segment means.
Segment Help Description	This is the help description of the segment name that will be used to explain to used who are new to the segmentation engine the meanings of the segments.
Start Date	This is the start date of the effectiveness of the row.
End Date	This is the end date of the effectiveness of the row.
Current Flag	This is a flag that indicates if the row is current or not.
Level Column	This is set to 'detail' and must never be changed.
Dim Char Ky Fld	This is the dimension table character key and is set by the system architecture to be unique for this dimension table within level column.
Table Number	This is 163 for this table.
Row Del From Source	This is set to 'Y' or 'N' depending whether the row is considered to be deleted from the source system or not.
Batch Number	This is set to the batch number in which the row was processed in the normal course of events. In the workbook it must be set to some value as determined to be the proper value for this installation by the Architect.
File Cycle Number	This is set to the file cycle number in which the row was processed in the normal course of events. In the workbook it must be set to some value as determined to be the proper value for this installation by the Architect.
View Number	This is set to the number allocated to the etl processing being performed in the normal course of events. In the workbook it must be set to some value as determined to be the proper value for this installation by the Architect.
Audit Timestamp	This is usually set to the system time of the row being extracted out of the staging area in the normal course of events. In the workbook it must be set to some value as determined to be the proper value for this installation by the Architect.

An example row for this tab is as follows:

Use	UID	Key	Segmentation Type Sdesc	Segment Code	Segment Short Desc	Segment Long Desc	Segment Help Desc	Start Date	End Date
Y	I	1	PCS	0	Totals	Totals	Totals	1900-01-01	2999-12-32
Y	I	1	PCS	1	Sweaters	Sweaters	Sweaters	1900-01-01	2999-12-32
Y	I	1	PCS	2	Blouse	Blouse	Blouse	1900-01-01	2999-12-32
Y	I	1	PCS	3	Trousers	Trousers	Trousers	1900-01-01	2999-12-32

4.1.23.Segment Codes Cols

The SeETL[RT] Segmentation Engine allows the user to define the text that will appear on reports for the segment codes. This tab contains the segment codes and their proper descriptions to be placed onto reports. This set of rows is the text that will be used for segment codes where the text is placed across the top of reports. Therefore the option was created to have different text for the same segment codes depending on whether the description was to be used in row headings or column headings.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
UID	This is the Update/Insert/Delete indicator. It allows the Architect to be able to edit individual rows in the database via UID statements generated by SeETL[DT].
Key	This is an integer key to the table that you must enter.
Segmentation Type Sdesc	This is the same segmentation type short description used in the above table to link the segment codes to the segmentation types.
Segment Short Description 00	This is a number from 0 to 64. There are a maximum of 64 different segments so that they can be placed onto a chess board. The 0 segment is taken to be the 'summary' segment or the 'totals' segment.
Segment Short Description 01	This is the short description of the segment name that will be used on reports where the segment description is used as a row header for the 01 segment of 64
Segment Short Description 02 - 64	This is the short description of the segment name that will be used on reports where the segment description is used as a row header for the 02 segment of 64. We have not repeated all 64 fields in this documentation. The reader will understand that there are 65 descriptions of these segments in the 65 columns.
Start Date	This is the start date of the effectiveness of the row.
End Date	This is the end date of the effectiveness of the row.
Current Flag	This is a flag that indicates if the row is current or not.
Level Column	This is set to 'detail' and must never be changed.
Dim Char Ky Fld	This is the dimension table character key and is set by the system architecture to be unique for this dimension table within level column.
Table Number	This is 165 for this table.
Row Del From Source	This is set to 'Y' or 'N' depending whether the row is considered to be deleted from the source system or not.
Batch Number	This is set to the batch number in which the row was processed in the normal course of events. In the workbook it must be set to some value as determined to be the proper value for this installation by the Architect.
File Cycle Number	This is set to the file cycle number in which the row was processed in the normal course of events. In the workbook it must be set to some value as determined to be the proper value for this installation by the Architect.
View Number	This is set to the number allocated to the etl processing being performed in the normal course of events. In the workbook it must be set to some value as determined to be the proper value for this installation by the Architect.
Audit Timestamp	This is usually set to the system time of the row being extracted out of the staging area in the normal course of events. In the workbook it must be set to some value as determined to be the proper value for this installation by the Architect.

It is suggested the reader refer to the sample spreadsheet provided with SeETL[DT] to understand the contents of this worksheet if more detail is required.

4.1.24.Product Markers

The SeETL[RT] Segmentation Engine uses a points based system allocated to products to calculate the placement of customers into segments based on their product purchase or usage history. The Product Markers table is the table used to allocate points to sales or usage to place customers into those segments.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
UID	This is the Update/Insert/Delete indicator. It allows the Architect to be able to edit individual rows in the database via UID statements generated by SeETL[DT].
Key	This is an integer key to the table that you must enter.
Product Key	This is an integer key to the vm_product table that links this row to the current row of the vm_product table. There is, as yet, no automated maintenance between updates of the vm_product table and this product markets table.
Product Name	This is the product name off the vm_product table. It is provided as documentation for the setting of the product markers in the table.
Product Group	This is the product group off the vm_product table. It is provided as documentation for the setting of the product markers in the table.
Product Category	This is the product category off the vm_product table. It is provided as documentation for the setting of the product markers in the table.
Version	It is now possible to have a number of versions of a product markers table so as to be able to build multiple versions of segmentation models at the same time to compare the effectiveness of the construction of the segmentation model.
Seg01 Points	This is the number of points this product will score in the segment allocated to segment 01 for this version of the segmentation model.
Seg02 Points	This is the number of points this product will score in the segment allocated to segment 02 for this version of the segmentation model.
Seg03-64 Points	This is the number of points this product will score in the segment allocated to segment 03 through to 64 for this version of the segmentation model.
Current Flag	This is a flag that indicates if the row is current or not.
Level Column	This is set to 'detail' and must never be changed.
Dim Char Ky Fld	This is the dimension table character key and is set by the system architecture to be unique for this dimension table within level column.
Table Number	This is 161 for this table.
Row Del From Source	This is set to 'Y' or 'N' depending whether the row is considered to be deleted from the source system or not.
Batch Number	This is set to the batch number in which the row was processed in the normal course of events. In the workbook it must be set to some value as determined to be the proper value for this installation by the Architect.
File Cycle Number	This is set to the file cycle number in which the row was processed in the normal course of events. In the workbook it must be set to some value as determined to be the proper value for this installation by the Architect.
View Number	This is set to the number allocated to the etl processing being performed in the normal course of events. In the workbook it must be set to some value as determined to be the proper value for this installation by the Architect.
Audit Timestamp	This is usually set to the system time of the row being extracted out of the staging area in the normal course of events. In the workbook it must be set to some value as determined to be the proper value for this installation by the Architect.

4.1.25.Data Model Views

The SeETL[DT] Workbook has now been updated to be able to store the ***BI4ALL Data Models*** in the workbook as well. This is a major upgrade to the SeETL[DT] and further represents our commitment and ability to place as much as is possible into the workbook and to manage as much of the development process as possible out of the workbook with the complementary SQL Server database and reports.

The ***BI4ALL Data Models*** have previously been delivered in only text format and in only the English Language. This was a conscious decision and the time and effort to maintain data models in tools like Erwin and PowerDesigner is very great. And these tools do not lend themselves easily to managing data models in multiple languages.

We had determined early on that we would eventually move the data models to the SeETL[DT] Workbook and that we would manage them in up to 10 languages for a single data model.

The ***BI4ALL Data Models*** now come as a separate set of workbooks that the client can then copy/change/load and generate views from as is desired. We will continue to release upgrades to these models and it is up to the client to manage the upgrade process based on the dictionary provided.

The ***Data Model Views*** tab that supports the translation of definitions of views into scripts that can be executed against a variety of databases is relatively complex because of the amount of information that is required to be captured.

The ***Data Model Views*** tab contains the following fields.

Number of Languages	
The first row of the spreadsheet defines the number of languages the data models are available in.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **nl** – Number of Languages.
Number of Languages	This field defines the number of languages that the base reports will be translated to. It must be a numeric value between 1 and 10 inclusive.
Input Output Definitions	
This is a single row that defines the target directory to which the view statements will be written.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **dir** – Directory Name for source and target directories.
Target Directory Lang 01	This is the target directory where the translated reports will be written for language 01.
Target Directory Lang 02	This is the target directory where the translated reports will be written for language 02.
Target Directory Lang 03	This is the target directory where the translated reports will be written for language 03.
Target Directory Lang 04	This is the target directory where the translated reports will be written for language 04.
Target Directory Lang 05	This is the target directory where the translated reports will be written for language 05.
Target Directory Lang 06	This is the target directory where the translated reports will be written for language 06.
Target Directory Lang 07	This is the target directory where the translated reports will be written for language 07.
Target Directory Lang 08	This is the target directory where the translated reports will be written for language 08.
Target Directory Lang	This is the target directory where the translated reports will be written for language 09.

09	
Target Directory Lang 10	This is the target directory where the translated reports will be written for language 10.

View Definitions and Short Descriptions

The views that are documented and defined in the data models have short and long descriptions in up to 10 languages defined for them. This set of rows defines the short descriptions for the views.

Field Name	Description
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **vdsd** – View Description Short Description
View Owner	The owner of the view. This is documentation to be able to link this view across to the view in the detailed view definition.
View Name	The English name of the view.
Submodel	The name of the submodel that the view belongs to.
View Number	This is a unique number given to each view that can be used to describe the view in shorthand. It is used on where clauses to expose the view in many cases as well.
View Description Lan 01	This is the description of the view in the 01 language.
View Description Lan 02	This is the description of the view in the 02 language.
View Description Lan 03	This is the description of the view in the 03 language.
View Description Lan 04	This is the description of the view in the 04 language.
View Description Lan 05	This is the description of the view in the 05 language.
View Description Lan 06	This is the description of the view in the 06 language.
View Description Lan 07	This is the description of the view in the 07 language.
View Description Lan 08	This is the description of the view in the 08 language.
View Description Lan 09	This is the description of the view in the 09 language.
View Description Lan 10	This is the description of the view in the 10 language.

View Definitions and Long Descriptions

The views that are documented and defined in the data models have short and long descriptions in up to 10 languages defined for them. This set of rows defines the long descriptions for the views.

Field Name	Description
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **vdld** – View Description Short Description
View Owner	The owner of the view. This is documentation to be able to link this view across to the view in the detailed view definition.
View Name	The English name of the view.
View Description Lan 01	This is the long description of the view in the 01 language.
View Description Lan 02	This is the long description of the view in the 02 language.
View Description Lan 03	This is the long description of the view in the 03 language.
View Description Lan 04	This is the long description of the view in the 04 language.
View Description Lan 05	This is the long description of the view in the 05 language.
View Description Lan 06	This is the long description of the view in the 06 language.
View Description Lan 07	This is the long description of the view in the 07 language.
View Description Lan 08	This is the long description of the view in the 08 language.
View Description Lan 09	This is the long description of the view in the 09 language.
View Description Lan 10	This is the long description of the view in the 10 language.

View Column Definitions

The bulk of the information held for views is the view column definitions. These definitions are obviously used to generate the sql statements that are the create view statements over the top of the physical tables which are defined and created using whatever tool the client chooses to use.

These rows allow the view to be created into up to 10 languages and allow the help text for the fields to be defined into up to 10 languages as well. Every effort has been made to make the generation of the views as flexible and as easy as possible while also maintaining the maximum amount of documentation and detail.

Field Name	Description

Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **vcd** – View Column Definitions.
Type	This defines the type of the row that SeETL[DT] software will interpret the row as. There are some 'helper' values in here to make the code a little easier to write. The valid values are: • **b** – Begin. Each view should have a 'b' on the first line to warn the generator that it is the beginning of a new view. • **m** – Move. This is the 'normal' or default value of a row type. • **d** – derived. If the user wishes to create a 'derived' value as part of a view he is required to place 'd-derived' into the column and then place the derivation in the cell. • **f** – From. The from clause is required to be marked with an 'f' in this column. • **w** – Where. The where clause is required to be marked with a 'w' in this column. • **e** – End. Each view is required to have an 'end' line to tell the generator to generate the end portions of the create view statement. This just made life a little easier.
Table Owner Name	This is the owner of the table and this will be used in the create view statement.
Table Name	This is the name of the underlying table that the data is in. It can be the 'meaningless' names that are used by SeETL[DT] or the client may change these names to be tables he implements. The underlying tables are only available in one language and they views over the top are then available in 10 languages.
Column Name	This is the name of the column in the underlying table.
View Owner Name	This is the owner of the view and this will be used in the create view statement.
View Name Lang01	This is the name of the view in the English language.
View Column Lang01	This is the name of the column of the view in the English language.
View Column Short Help Text Lang01	This is the short help text for the column in the English Language.
View Column Long Help Text Lang01	This is the long help text for the column in the English Language.
View Name Lang02	This is the name of the view for language 02.
View Column Lang02	This is the name of the column for language 02.
View Column Short Help Text Lang02	This is the short help text for language 02.
View Column Long Help Text Lang02	This is the long help text for language 02.
View Name Lang03	This is the name of the view for language 03.
View Column Lang03	This is the name of the column for language 03.
View Column Short Help Text Lang03	This is the short help text for language 03.
View Column Long Help Text Lang03	This is the long help text for language 03.
View Name Lang04	This is the name of the view for language 04.
View Column Lang04	This is the name of the column for language 04.
View Column Short Help Text Lang04	This is the short help text for language 04.
View Column Long Help Text Lang04	This is the long help text for language 04.
View Name Lang05	This is the name of the view for language 05.
View Column Lang05	This is the name of the column for language 05.
View Column Short Help Text Lang05	This is the short help text for language 05.
View Column Long Help Text Lang05	This is the long help text for language 05.
View Name Lang06	This is the name of the view for language 06.
View Column Lang06	This is the name of the column for language 06.
View Column Short Help Text Lang06	This is the short help text for language 06.

View Column Long Help Text Lang06	This is the long help text for language 06.
View Name Lang07	This is the name of the view for language 07.
View Column Lang07	This is the name of the column for language 07.
View Column Short Help Text Lang07	This is the short help text for language 07.
View Column Long Help Text Lang07	This is the long help text for language 07.
View Name Lang08	This is the name of the view for language 08.
View Column Lang08	This is the name of the column for language 08.
View Column Short Help Text Lang08	This is the short help text for language 08.
View Column Long Help Text Lang08	This is the long help text for language 08.
View Name Lang09	This is the name of the view for language 09.
View Column Lang09	This is the name of the column for language 09.
View Column Short Help Text Lang09	This is the short help text for language 09.
View Column Long Help Text Lang09	This is the long help text for language 09.
View Name Lang10	This is the name of the view for language 10.
View Column Lang10	This is the name of the column for language 10.
View Column Short Help Text Lang10	This is the short help text for language 10.
View Column Long Help Text Lang10	This is the long help text for language 10.

4.1.26.Data Model Joins

The SeETL[DT] Workbook has been upgraded to support the definition of all joins in the ***BI4ALL Data Models*** into the workbook. This is used to be able to update business objects universes with all the joins needed to be placed into the universe. It also allows SeETL[DT] to be able to generate Referential Integrity Constraints into the ***BI4ALL Documentation Database*** that can be re-engineered into various data modeling tools.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
Database	This is the name of the database that contains the data model that these joins are being defined for.
Left Table Owner	This is the name of the owner of the table/view that is on the left hand side of the join.
Left Table	This is the name of the table/view that is on the left hand side of the join.
Left Column	This is the name of the column that is on the left hand side of the join.
Right Table Owner	This is the name of the owner of the table/view that is on the right hand side of the join.
Right Table	This is the name of the table/view that is on the right hand side of the join.
Right Column	This is the name of the column that is on the right hand side of the join.
Join Type	This defines the join type that is to be applied to the join.
Cardinality	This defines the cardinality of the join. Many to one, one to one, one to many.

4.1.27.Generate Stored Procedures

BI4ALL Analysis is a new product that we are working on. Currently there are 70+ Microsoft Report Services Reports in the product. These reports get data from the Data Warehouse via the ***'Data Access Layer'***. The ***Data Access Layer*** is a library of about 60,000 lines of Stored Procedures. We initially released the Stored Procedures for SQL Server only. We did not want to undertake the massive development effort of developing and deploying multiple sets of stored procedures to multiple databases using multiple languages until such time as it was necessary.

In order to deliver stored procedures in many database language dialects we elected to upgrade the ***BI4ALL Workbench*** to be able to read a set of stored procedures defined in a workbook and to apply filtering and directives to those pieces of text to produce separate files of output. Each separate file of output would be the stored procedure in the specific language for a specific database. The input file would contain all the different text required for all the different languages.

It is felt that such a solution, a single input text file, a generator/translator, and separate output files is a perfectly adequate way to manage different sets of stored procedures so as to allow IBI to maintain a single code base but to also facilitate the easy deployment of the ***Data Access Layer*** to any of the supported databases.

The Generate Stored Procedures tab that supports the translation of ***BI4ALL*** Analysis Stored Procedures into the different database stored procedure languages is relatively complex because of the amount of information that is required to be captured.

The ***Generate Stored Procedures Reports*** tab contains the following fields.

Number of Languages	
The first row of the spreadsheet defines the number of languages the Stored Procedure is stored in.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **nl** – Number of Languages.
Number of Languages	This field defines the number of languages that the input stored procedure file will emit as the generation process takes place. It must be a numeric value between 1 and 10 inclusive.
Input Output Definitions	
This is a single row that defines the source directory in which the source Stored Procedures can be found. The remaining columns represent the target directories that the translated Stored Procedures will be written to.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **dir** – Directory Name for source and target directories.
Source Directory Name	This is the name of the directory where the Stored Procedures are stored as source test files that are edited by any standard text editor.
Target Directory Lang 01	This is the target directory where the generated and translated stored procedures will be written for language 01.
Target Directory Lang 02	This is the target directory where the generated and translated stored procedures will be written for language 02.
Target Directory Lang 03	This is the target directory where the generated and translated stored procedures will be written for language 03.
Target Directory Lang 04	This is the target directory where the generated and translated stored procedures will be written for language 04.
Target Directory Lang 05	This is the target directory where the generated and translated stored procedures will be written for language 05.
Target Directory Lang 06	This is the target directory where the generated and translated stored procedures will be written for language 06.
Target Directory Lang	This is the target directory where the generated and translated stored procedures will

07	be written for language 07.
Target Directory Lang 08	This is the target directory where the generated and translated stored procedures will be written for language 08.
Target Directory Lang 09	This is the target directory where the generated and translated stored procedures will be written for language 09.
Target Directory Lang 10	This is the target directory where the generated and translated stored procedures will be written for language 10.

Stored Procedures Name of File Definitions

There is no requirement for the stored procedure to be stored in the same file name for the different language versions. So SeETL[DT] includes the ability to name the input file to be read and then associate that input file with different output file names in different directories. This seemed like a useful feature to add and it replicated the similar feature in the report translation process.

Field Name	Description
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **spfn** – Stored Procedure File Names Definition
Procedure Name	The Stored Procedure Name is any name up to 255 characters that is a unique name for the Stored Procedure. It is used inside the workbook to link the name of the stored procedures with the actual generation / translation directive rows. It can also be tied to the stored procedure documentation worksheet.
Source File Name	The Source File Name is the physical name of the file in which one or more stored procedures will be found. It is connected to the Source Directory Name as above. The source file will be read for the stored procedure languages it will be converted to the various languages and placed into the Target File Names for each language as specified below.
Target FileName Lang01	Each stored procedure is converted to between 1 and 10 languages inclusive. Therefore each stored procedure must have up to 10 output files defined so that the new reports can be written to these output files.
Target FileName Lang02	This is the file that the language 02 will be written to.
Target FileName Lang03	This is the file that the language 03 will be written to.
Target FileName Lang04	This is the file that the language 04 will be written to.
Target FileName Lang05	This is the file that the language 05 will be written to.
Target FileName Lang06	This is the file that the language 06 will be written to.
Target FileName Lang07	This is the file that the language 07 will be written to.
Target FileName Lang08	This is the file that the language 08 will be written to.
Target FileName Lang09	This is the file that the language 09 will be written to.
Target FileName Lang10	This is the file that the language 10 will be written to.

Language Directives Set

The languages set parameters tell the generator / translator which languages should be considered turned 'on' or 'off' as well as provide some language specific parameters to apply to the generation / translation process. All databases have slightly different aspects to them and these aspects needs to be provided for using the 'Language Set' directives.

Field Name	Description
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **lds** – Language Directives Set.
Language Number	This is an integer listing the language number. It is only required to provide a simply

	key to the documentation table that this set of directives is written to.
Language Status	This is a flag set to 'On' or 'Off' for a language. Please take care that this matches with the number of languages directive and that the number of languages the user wishes to generate/translate occur in the top 'n' rows of this set of directive entries. There is no allowance for turning directives on or off up and down the list of languages.
Language Table Owner	Inside the text the string ##languagetableowner## will be used to denote the language table owner parameter. The different databases use different default table owners and most companies also have standards about the name of the table owner for tables. Some companies insist that the table owner is explicitly named and we have adopted the idea that the table owner must be explicitly named. It is just safer that way. The generator/translator uses this input directive and it replaces all instances of ##languagetableowner## in the text with this parameter.
Language Parameter Prefix	Inside the text the string ##languageparameterprefix## will be used to denote the prefix to be used for parameters named in the stored procedure. In SQL Serve the @ symbol is used to denote a prefix. In Netezza no symbol is used but the alias for the variable name can not be the same name as the name of an existing field. So a prefix is required to make sure no clashes of field names and variable names occurs The generator/translator uses this input directive and it replaces all instances of ## languageparameterprefix ## in the text with this parameter.
Directives Set	
The generation/translation process allows for more directives that are applied across all languages. These directives work very much like C 'defines' statements. A directive is either 'On' or 'Off'. If the directive is 'on' those lines of text marked by the directives will be emitted to ALL languages. Similarly, if the directive 'All:on' is set to on then the text will be emitted to ALL output language files. It is expected that as time goes on we may find need for more directives than are currently seen as being required.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **ds** – Directives Set. This tells the generator that this is a 'directive setting' line.
Directive Number	This is an integer listing the directive number. It is only required to provide a simply key to the documentation table that this set of directives is written to.
Directive Name	Each of the directives have names. The list of current directive names and their meanings is given below: • RedundantJoins All template stored procedures have all the dimension tables they join to already pre-joined in the text of the stored procedure. This is to save the client time and effort when actually implementing the stored procedures. However, in many cases these joins are not needed in the stored procedures for reports or templates. They are there to make adding joins to new dimensions easier. As such they can be described as 'redundant'. However, even though the joins may be redundant they will cause processing to occur if they are submitted to the database. Hence, there is a need to be able to easily turn them on and off. This is the purpose of this directive. To easily turn redundant joins and the listing of the table name in the where clause on and off more easily. • SQLServerpartitioning In SQL Server 2005 Enterprise Edition there was still only ONE partitioning column. Therefore, IBI was required to come up with a better mechanism for partitioning for our clients on SQL Server. We did this by creating a two part key based on company reporting structure and time. The partitions were broken up into 'country by time' typically. This provided enough flexibility and made the partitions small enough to get decent performance out of the databases. However, the stored procedures needed to be told about this

	feature and it needed to be embedded in the SQL for the stored procedure. Obviously, on other databases, this feature needs to be turned off. This is the purpose of this directive. To turn on and off the feature that enables multi-part petitioning for SQL Server.
Directive Status	This is set to the literals “On” or “Off

4.1.28. Document Stored Procedures

BI4ALL Analysis is a new product that we are working on. Currently there are 70+ Microsoft Report Services Reports in the product. These reports get data from the Data Warehouse via the ***'Data Access Layer'***. The ***Data Access Layer*** is a library of about 60,000 lines of Stored Procedures. We initially released the Stored Procedures for SQL Server only. We did not want to undertake the massive development effort of developing and deploying multiple sets of stored procedures to multiple databases using multiple languages until such time as it was necessary.

One issue with having a large library of stored procedures is documenting them in such a way that the dictionary for the stored procedures is easily available and searchable for all those who would wish to understand what is in the stored procedure.

To the current point in time only IBI developers and developers at business partners have been required to learn these stored procedures. And they have had to do it the 'normal' (read hard) way. They have to sit down and read the source code and learn it. In most cases the people involved are then involved with the stored procedures for some years so the learning curve has not been significant compared to the time using the stored procedures.

This is about to change. We are about to make the stored procedures 'open' and 'available' to business users who wish to use them to do such things as call them from inside a query window and cut/paste the results into excel spreadsheets. We are expecting to be able to run stored procedures from inside excel spreadsheets in the not too distant future as Microsoft makes excel more open to it's own database as well as others.

Further, we expect that now that the stored procedures will be ported to Netezza that there will be a need to integrate the stored procedures with business objects. This will mean business objects report writers will need documentation for these stored procedures.

In order to deliver documentation for stored procedures we have developed a simple catalogue that will be extended over time. We are going to go about the process of filling up this catalogue for all the existing procedures as well as providing it to clients to document their own stored procedures. Tools to document stored procedures have been poor to date. This tool will make some significant contribution to our clients to document their stored procedures.

The Document Stored Procedures tab that supports the documentation of ***BI4ALL Analysis*** Stored Procedures is relatively complex because of the amount of information that is required to be captured.

The ***Document Stored Procedures Reports*** tab contains the following fields:

Number of Languages	
The first row of the spreadsheet defines the number of languages the base report is to be translated into.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **nl** – Number of Languages.
Number of Languages	This field defines the number of languages that the stored procedure documentation will exist in and be loaded in to the catalogue database. Usually it will only be English but the ability has been created to deploy in multiple languages for descriptions.
Stored Procedure Definitions and Short Descriptions	
The stored procedures that are documented and defined in this workbook have short and long descriptions in up to 10 languages defined for them. This set of rows defines the short descriptions for the Stored Procedures.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **spsd** – Stored Procedure Description Short Description
Procedure Owner	The owner of the Stored Procedure. This is documentation to be able to link this Stored Procedure across to the Stored Procedure Name in the detailed Stored Procedure definition.
Procedure Name	The English name of the Stored Procedure.

Submodel	The name of the submodel that the Stored Procedure belongs to.
Procedure Number	This is a unique number given to each Stored Procedure that can be used to describe the Stored Procedure in shorthand. It is used simply to make it easier to refer to Stored Procedures and to use as a key in the documentation database.
Procedure Description Lan 01	This is the description of the Stored Procedure in the 01 language.
Procedure Description Lan 02	This is the description of the Stored Procedure in the 02 language.
Procedure Description Lan 03	This is the description of the Stored Procedure in the 03 language.
Procedure Description Lan 04	This is the description of the Stored Procedure in the 04 language.
Procedure Description Lan 05	This is the description of the Stored Procedure in the 05 language.
Procedure Description Lan 06	This is the description of the Stored Procedure in the 06 language.
Procedure Description Lan 07	This is the description of the Stored Procedure in the 07 language.
Procedure Description Lan 08	This is the description of the Stored Procedure in the 08 language.
Procedure Description Lan 09	This is the description of the Stored Procedure in the 09 language.
Procedure Description Lan 10	This is the description of the Stored Procedure in the 10 language.
Stored Procedure Definitions and Long Descriptions	
The Stored Procedures that are documented and defined in this workbook have short and long descriptions in up to 10 languages defined for them. This set of rows defines the long descriptions for the Stored Procedures.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **spld** – Stored Procedure Description Long Description
Procedure Owner	The owner of the Stored Procedure. This is documentation to be able to link this Stored Procedure across to the Stored Procedure Name in the detailed Stored Procedure definition.
Procedure Name	The English name of the Stored Procedure.
Procedure Description Lan 01	This is the long description of the Stored Procedure in the 01 language.
Procedure Description Lan 02	This is the long description of the Stored Procedure in the 02 language.
Procedure Description Lan 03	This is the long description of the Stored Procedure in the 03 language.
Procedure Description Lan 04	This is the long description of the Stored Procedure in the 04 language.
Procedure Description Lan 05	This is the long description of the Stored Procedure in the 05 language.
Procedure Description Lan 06	This is the long description of the Stored Procedure in the 06 language.
Procedure Description Lan 07	This is the long description of the Stored Procedure in the 07 language.
Procedure Description Lan 08	This is the long description of the Stored Procedure in the 08 language.
Procedure Description Lan 09	This is the long description of the Stored Procedure in the 09 language.
Procedure Description Lan 10	This is the long description of the Stored Procedure in the 10 language.
Stored Procedure Input Definitions	
To be able to properly use a Stored Procedure the user calling the stored procedure must know the input fields that the stored procedure is expecting to receive. This section documents the input fields of the stored procedure in enough depth that the end user has a reasonable chance to be able to properly call the stored procedure.	

Field Name	Description
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **spid** – Stored Procedure Input Definitions.
Type	This defines the type of the row that the SeETL[DT] software will interpret the row as. There are some 'helper' values in here to make the code a little easier to write. The valid values are: • **b** – Begin. Each view should have a 'b' on the first line to warn the generator that it is the beginning of a new view. • **m** – Move. This is the 'normal' or default value of a row type. • **e** – End. Each view is required to have an 'end' line to tell the generator to generate the end portions of the create view statement. This just made life a little easier.
Procedure Owner	The owner of the Stored Procedure. This is documentation to be able to link this Stored Procedure across to the Stored Procedure Name in the detailed Stored Procedure definition.
Procedure Name	The English name of the Stored Procedure.
Input Column Name	This is the name of the input columns to the stored procedure.
Input Column Data Type	This is a text description of the input column data type.
Input Column Short help text Lang01	This is the short help text of the input column in language 01.
Input Column Long help text Lang01	This is the long help text of the input column in language 01.
Input Column Short help text Lang02	This is the short help text of the input column in language 02.
Input Column Long help text Lang02	This is the long help text of the input column in language 02.
Input Column Short help text Lang03	This is the short help text of the input column in language 03.
Input Column Long help text Lang03	This is the long help text of the input column in language 03.
Input Column Short help text Lang04	This is the short help text of the input column in language 04.
Input Column Long help text Lang04	This is the long help text of the input column in language 04.
Input Column Short help text Lang05	This is the short help text of the input column in language 05.
Input Column Long help text Lang05	This is the long help text of the input column in language 05.
Input Column Short help text Lang06	This is the short help text of the input column in language 06.
Input Column Long help text Lang06	This is the long help text of the input column in language 06.
Input Column Short help text Lang07	This is the short help text of the input column in language 07.
Input Column Long help text Lang07	This is the long help text of the input column in language 07.
Input Column Short help text Lang08	This is the short help text of the input column in language 08.
Input Column Long help text Lang08	This is the long help text of the input column in language 08.
Input Column Short help text Lang09	This is the short help text of the input column in language 09.
Input Column Long help text Lang09	This is the long help text of the input column in language 09.
Input Column Short help text Lang10	This is the short help text of the input column in language 10.
Input Column Long help text Lang10	This is the long help text of the input column in language 10.

Stored Procedure Output Definitions	
To be able to properly use a Stored Procedure the user calling the stored procedure must know the output fields that the stored procedure is expecting to receive. This section documents the output fields of the stored procedure in enough depth that the end user has a reasonable chance to be able to properly take the data from the stored procedure and place it onto a report or into some other front end reporting tool.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **spod** – Stored Procedure Output Definitions.
Type	This defines the type of the row that the SeETL[DT] software will interpret the row as. There are some 'helper' values in here to make the code a little easier to write. The valid values are: • **b** – Begin. Each view should have a 'b' on the first line to warn the generator that it is the beginning of a new view. • **m** – Move. This is the 'normal' or default value of a row type. • **e** – End. Each view is required to have an 'end' line to tell the generator to generate the end portions of the create view statement. This just made life a little easier.
Procedure Owner	The owner of the Stored Procedure. This is documentation to be able to link this Stored Procedure across to the Stored Procedure Name in the detailed Stored Procedure definition.
Procedure Name	The English name of the Stored Procedure.
Output Column Name	This is the name of the Output columns to the stored procedure.
Output Column Data Type	This is a text description of the Output column data type.
Output Column Short help text Lang01	This is the short help text of the Output column in language 01.
Output Column Long help text Lang01	This is the long help text of the Output column in language 01.
Output Column Short help text Lang02	This is the short help text of the Output column in language 02.
Output Column Long help text Lang02	This is the long help text of the Output column in language 02.
Output Column Short help text Lang03	This is the short help text of the Output column in language 03.
Output Column Long help text Lang03	This is the long help text of the Output column in language 03.
Output Column Short help text Lang04	This is the short help text of the Output column in language 04.
Output Column Long help text Lang04	This is the long help text of the Output column in language 04.
Output Column Short help text Lang05	This is the short help text of the Output column in language 05.
Output Column Long help text Lang05	This is the long help text of the Output column in language 05.
Output Column Short help text Lang06	This is the short help text of the Output column in language 06.
Output Column Long help text Lang06	This is the long help text of the Output column in language 06.
Output Column Short help text Lang07	This is the short help text of the Output column in language 07.
Output Column Long help text Lang07	This is the long help text of the Output column in language 07.
Output Column Short help text Lang08	This is the short help text of the Output column in language 08.
Output Column Long help text Lang08	This is the long help text of the Output column in language 08.
Output Column Short help text Lang09	This is the short help text of the Output column in language 09.

Output Column Long help text Lang09	This is the long help text of the Output column in language 09.
Output Column Short help text Lang10	This is the short help text of the Output column in language 10.
Output Column Long help text Lang10	This is the long help text of the Output column in language 10.

4.1.29.Compound Views

The Compound Views tab is used to store definitions of views that require some form of compound statement such as the 'union all' statement that the Architect might like to define. Such statements are not supported by other tabs such as the mappings or 'Other Views' tabs.

In occasional implementations it has happened that there are some 'union views' which get created to make some portion of work that little bit easier. These views have normally just been created in text documents. However, in text documents they are not particularly 'visible' and this broke with the concept of 'You can see your ETL' which is a design point of SeETL[DT].

It was decided to make it possible to put all 'Compound Views' into the SeETL[DT] Workbook and publish them via the portal just to 'close the gap' on having all portions available. The Architect can decide for him/her self whether he/she will use these other views for this style of documentation.

The Compound Views tab contains the following fields.

Field Name	Description
Use Ind	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Type	The type of the field being defined. It may be: • 'B' – Begin. Each new view must begin with a 'B' for begin line. • 'E' – End. Each view must have a null 'E' for end line. • 'M' – Move • 'D' – Derived. • 'C' – Compound. On a line denoting the 'compound' view the type is 'C'. • 'F – From • 'W' – Where • 'G' – Group By • 'O' – Order by
Table owner name	This is the owner of the table over which the view is being placed.
Table name	This is the name of the table over which the view is being placed. When the 'Type' is 'C' for 'Compound View' this field contains the compound view statement.
Col name	This is the name of the column in the table over which the view is being placed.
View owner name	This is the name of the owner of the view that is being created. It is not necessarily the same as the name of the owner of the table.
View name	This is the name of the view being created.
View col	This is the name of the column being created in the view.

4.1.30.Create Tables

The SeETL[DT] Workbook has now been updated to be able to store the ***BI4ALL Data Model Tables*** in the workbook as well. This is a major upgrade to SeETL[DT] and further represents our commitment and ability to place as much as is possible into the workbook and to manage as much of the development process as possible out of the workbook with the complementary SQL Server database and reports.

The ***BI4ALL Data Model Tables*** have previously been delivered in only text format and in only the English Language. This was a conscious decision as the time and effort to maintain data models in tools like Erwin and PowerDesigner is very great.

The ***BI4ALL Data Model Tables*** now come as a separate workbook that the client can then copy/change/load and generate tables from as is desired. We will continue to release upgrades to these models and it is up to the client to manage the upgrade process based on the dictionary provided.

The ***Create Tables*** tab that supports the translation of definitions of tables into scripts that can be executed against a variety of databases is relatively complex because of the amount of information that is required to be captured.

The ***Create Tables*** tab contains the following fields.

Target Directory	
The first row of the spreadsheet defines the Target Directory that the create table statements will be generated into.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **dir** – Indicating that the row will contain the name of the target directory for the table scripts.
Target Directory	• This field contains the name of the target directory for the create table scripts. It is recommended that it contain no quotes and no blanks as these have not been tested and will not be supported.
Table Definitions	
The bulk of the information held for tables is the table definitions rows. These definitions are obviously used to generate the sql statements that are the create table statements.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **tab** – Table Definitions.
Type	This defines the type of the row that the SeETL[DT] software will interpret the row as. There are some 'helper' values in here to make the code a little easier to write. The valid values are: • **b** – Begin. Each table should have a 'b' on the first line to warn the generator that it is the beginning of a new view. • **m** – Move. This is the 'normal' or default value of a row type. • **n** – eNd. This indicates the last row of the definitions of the columns for the table. The rows that follow will contain special clauses on a database specific basis. • **s** – Special. All databases have different clauses that can be used at the end of the create table statement to do things like create relationships for tables spaces or storage spaces and things like that. The way this have been implemented in the SeETL[DT] Workbook is to allow free form text in cells that will be directly replicated into the create table statements. • **e** – End. Each table is required to have an 'end' line to tell the generator to generate the end portions of the create table statement. This just made life a

	little easier.
Table Owner Name	This is the owner of the table and this will be used in the create table statement.
Table Name	This is the name of the table that the data is in. It can be the 'meaningless' names that are used by BI4ALL or the client may change these names to be tables he implements. The underlying tables are only available in one language and the views over the top are then available in 10 languages.
Column Name	This is the name of the column in the table.
Column Number	This is the ordinal number of the column in the table.
Data Type Deploy	This is the data type that will be accepted by the database that the table is being deployed to. The deployment database may be a different database to the documentation database. For example a client may deploy to Netezza or Oracle but document to SQL Server to gain the benefits of the already provided Report Services Reports.
Data Type Documentation	This is the data type that will be accepted by the database that the table is being documented to.
Deploy Null Clause	This is the null clause that will be accepted by the database that the table is being deployed to. It may be left empty of no null clause is desired. However, in most cases it is recommended that a null clause is specified.
Documentation Null Clause	This is the null clause that will be accepted by the database that the table is being documented to. It may be left empty of no null clause is desired. However, in most cases it is recommended that a null clause is specified.
Other Clauses	This is a field that is made available to provide any other clauses that need to be created for the table to be properly created onto the deployment or documentation database. In some cases it might be necessary to store multiple rows with multiple instances of other clauses and to switch on and off the 'Use' flags in the first column to get the SQL to generate propertly for all cases. When the documentation and deployment databases are different databases there will be these small issues.

4.1.31.Create Count SQL

The SeETL[DT] Workbook has now been updated to be able to generate the 'sql input rows counting statements' that were previously written by hand. These statements count the number of input rows in a view at the beginning of processing and are a part of the standard SeETL[RT] Audit processing.

A sample statement for such a row count is as follows.

```
-----------------------------------------------------------------------------------------------------------------------
--
--  spreadsheet name:       seetlwbmodeldemoworksheet01.xml spreadsheet version:        v3.1.00
--  spreadsheet file name:  c:\seetl\designtime\3.1.00\sourcecode\seetl\bi4allmodeldemoworksheet01v3.1.00.xml
--  file name:              c:\temp\bi4allbase\target01\rct_z01_vf_sale_txn_01.sql
--  windows domain:         emsbidev01
--  windows user:           administrator
--  windows workstation:    emsbidev01
--  seetl current directory: c:\seetl\designtime\3.1.00\sourcecode\seetl
--  windows version:        microsoft windows nt 5.2.3790 service pack 2
--  dot net version:        2.0.50727.1433
--  seetl workbench version: seetl 3.1.00 - 2010-09-01
--  timestamp:              17/06/2011 16:28:27
--  description:            description
--
-----------------------------------------------------------------------------------------------------------------------

 insert into dbo.ctl_file_cycle_control
 select 'z01_vf_sale_txn_01'  as pk_view_name
, (select coalesce(max(ctl_file_cycle_control.pk_file_cycle_number),0) + 1
from  dbo.ctl_file_cycle_control ctl_file_cycle_control
where 1=1 and ctl_file_cycle_control.pk_view_name = 'z01_vf_sale_txn_01'  ) as pk_file_cycle_number
, 'vf_sale_txn' as table_name
 ,(select ctl_batch_control.batch_date from dbo.ctl_batch_control ctl_batch_control where ctl_batch_control.batch_complete_flag = 0 ) as batch_date
 ,(select ctl_batch_control.pk_batch_number from dbo.ctl_batch_control ctl_batch_control where ctl_batch_control.batch_complete_flag = 0 ) as batch_number
 ,(select count(*) from dbo.z01_vf_sale_txn_01) as number_of_records

 ;
```

As can be seen from the statement. When it executes it will insert a row into the ctl_file_cycle_control table. The row will include the following elements.

1. The name of the mapping that the count statement is correlated to. Therefore the audit of the actual executed mapping statement should process the same number of rows.If it does not there is some sort of error.

2. The file cycle number. This number is the number of times this mapping has been run. A file may only be processed once per batch number. Two file cycle numbers in one batch is considered an error. A file is not necessarily processed in every batch. Batches of jobs may contain different sets of files to process.

3. The name of the underlying view or table that will be updated.

4. The batch date that the ctl_batch_control table for the current batch is saying is the batch processing date. The batch processing date can be different to the system date.

5. The batch number that the ctl_batch_control table for the current batch is saying is the batch number that is being processed.

From this information it is possible to record in the ctl_file_cycle_control table the number of rows to be processed for each input view, the file cycle in which they were processed, and the batch in which they were processed. The file cycle number and batch number are placed onto every record in the data warehouse as well as a number representing the mapping name. In prior versions the mapping name was placed onto each row. However this was seen as a little wasteful and it has been reduced to an integer. In the vast majority of cases only one mapping updates a target view. But for some processing like vm_party many mappings target and update the same underlying view or table.

The ***SeETL Create Count SQL*** tab contains the following fields.

Target Directory	
The first row of the spreadsheet defines directory that the sql statements will be written to.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **dir** – Indicating that the row will contain the name of the target directory for the table scripts.
Target Directory	• This field contains the name of the target directory for the create count sql scripts. It is recommended that it contain no quotes and no blanks as these have not been tested and will not be supported.
Create Count Definitions	
The create count definitions are of the workbook contains simple directives to generate the create count sql statements. Please note that the create count SQL is generated according to the database type specified for the staging area.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **count** – SQL Row Count generation directives.
Target File Name	This is the name of the file that will be created or overwritten when the generator creates the count sql. It is recommended that some form of prefix be used to identify it as the count sql for a mapping. The proposed standard prefix is rct for 'row count'.
View Owner	This is the name of the owner of the view that is to have it's row count counted.
View Name	This is the name of the view that is to have it's row count counted.
Table Name	This is the name of the target table/view that the rows will be sent to when the mapping actually runs to process the rows in the input view to the mapping.

4.1.32.Create Runstats SQL

The SeETL[DT] Workbook has now been updated to be able to store runstats commands. It is well understood that this is somewhat redundant and that the run stats commands can be just as easily written directly into files. However, a large client asked us to place as much information as possible into the workbook and also place that data into the SeETL[DT] dictionary so that it was published and public knowledge.

This is seen as valuable and this is why these tabs have been added.

Because all the different databases have very different formats for the run stats commands we did not try and generate runstats commands for each database type. We simply store the runstats commands and place them into tables in the dictionary.

The ***SeETL[DT] Create Runstats SQL*** tab contains the following fields.

Target Directory	
The first row of the spreadsheet defines directory that the sql statements will be written to.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **dir** – Indicating that the row will contain the name of the target directory for the table scripts.
Target Directory	• This field contains the name of the target directory for the create runstats sql scripts. It is recommended that it contain no quotes and no blanks as these have not been tested and will not be supported.
Create Runstats SQL Definitions	
The create run stats definitions are of the workbook contains simple fields to store the run stats sql and to abe able to query the dictionary to find the statements that have been stored without needing to query the text of the statements.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **runstats** – Indicates an SQL Runstats generation directive.
Target File Name	This is the name of the file that will be created or overwritten when the generator creates the runstats sql. It is recommended that some form of prefix be used to identify it as the runs stats sql for a line entry in this workbook. The proposed standard prefix is runstats for 'runs stats sql'.
Object Owner	This is the name of the owner of the object that is to have the runstats performed against it. This is for documentation and query.
Object Name	This is the name of the object that is to have the runstats performed against it. This is for documentation and query.
Drop Stats Command	Some databases require statistics to be dropped or over written and require this to be specified. Therefore this command could be the drop command for the runstats or it could be a null command that does nothing depending on what the next command is.
Create Stats Command	Some databases allow the rebuilding of existing stats or the creation of a new set of statistics. Depending on what commands the DBA wishes to submit to the database on a regular basis this command may be a create or rebuild stats command. Depending on what this command is the drop stats command may be different. The two commands are placed one after another in the generated file. The DBA is the authority as to exactly what should be placed into these two commands per row.

4.1.33.Create Table as Select

The SeETL[DT] Workbook has now been updated to be able to store create table as select commands. This sheet does not allow the creation of complex select statements. Should the create table as select be desired to be a complex statement then it is recommended that the client create a view in 'other views' to create the complex select statement and then create the create table as select based on that view.

Because all the different databases have different formats for the create table as select commands we did not try and generate create table as select commands for each database type. We simply store the create table as select commands and place them into tables in the dictionary.

The ***SeETL[DT] Create Tables As Select*** tab contains the following fields.

Target Directory	
The first row of the spreadsheet defines directory that the sql statements will be written to.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **dir** – Indicating that the row will contain the name of the target directory for the table scripts.
Target Directory	• This field contains the name of the target directory for the create table as sql scripts. It is recommended that it contain no quotes and no blanks as these have not been tested and will not be supported.
Create Table as Select SQL Definitions	
The create table as select definitions of the workbook contains simple fields to store the create table as select sql and to abe able to query the dictionary to find the statements that have been stored without needing to query the text of the statements.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **ctas** – Indicates a create table as select generation directive.
Target File Name	This is the name of the file that will be created or overwritten when the generator creates the create table as select sql. It is recommended that some form of prefix be used to identify it as the create table as select sql for a line entry in this workbook. The proposed standard prefix is ctas for 'create table as select sql'.
Table Owner	This is the name of the owner of the table that is to be the newly created table as a result of the create table as select statement being executed. This is for documentation and query.
Table Name	This is the name of the table that is to be the newly created table as a result of the create table as select statement being executed. This is for documentation and query.
Truncate or Drop Table Command	The DBA will have the option to either truncate the existing table having previously created it some other way or to drop the pre-existing table. Some DBAs have a great preference for truncate table over drop table. This is because in most databases the drop table also drops indexes and any statistics that are associated with the data in the table. Though, to be fair, some databases are allowing these other objects to be retained even when a table is dropped.
Create or Insert Table Command	Depending on whether the DBA has decided to allow drop tables or truncate tables for these tables the command that is then executed is the create table command or the insert rows into table command.

4.1.34.Create Index

The SeETL[DT] Workbook has now been updated to be able to store create index commands.

Because all the different databases have different formats for the create index commands we did not try and generate create index commands for each database type. We simply store the create index commands and place them into tables in the dictionary.

The ***SeETL[DT] Create Index*** tab contains the following fields.

Target Directory	
The first row of the spreadsheet defines directory that the sql statements will be written to.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **dir** – Indicating that the row will contain the name of the target directory for the table scripts.
Target Directory	• This field contains the name of the target directory for the create index sql scripts. It is recommended that it contain no quotes and no blanks as these have not been tested and will not be supported.
Create Index SQL Definitions	
The create index definitions of the workbook contains simple fields to store the create index sql and to abe able to query the dictionary to find the statements that have been stored without needing to query the text of the statements.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **index** – Indicates a create index generation directive.
Target File Name	This is the name of the file that will be created or overwritten when the generator creates the create index sql. It is recommended that some form of prefix be used to identify it as the create index sql for a line entry in this workbook. The proposed standard prefix is index for ‘create index sql’.
Table Owner	This is the name of the owner of the table that is to be indexed. This is for documentation and query.
Table Name	This is the name of the table that is to be indexed. This is for documentation and query.
Drop Index Command	In most databases indexes are dropped and recreatead rather than rebuilt. It is expected that this cell will contain drop index commands.
Create Index Command	In most databases indexes are dropped and recreatead rather than rebuilt. It is expected that this cell will contain create index commands.

4.1.35.Create Storage Object

The SeETL[DT] Workbook has now been updated to be able to store create storage objects commands.

A storage object are thinkgs such as tablespaces in DB2 and Oracle and File Groups in SQL Server. Most databases have some form of Storage Object that is required to be created to place tables and index data into.

Because all the different databases have different formats for the create storage objects commands we did not try and generate create storage objects commands for each database type. We simply store the create storage objects commands and place them into tables in the dictionary.

The ***SeETL[DT] Create Storage Object*** tab contains the following fields.

Target Directory	
The first row of the spreadsheet defines directory that the sql statements will be written to.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **dir** – Indicating that the row will contain the name of the target directory for the table scripts.
Target Directory	• This field contains the name of the target directory for the create storage object sql scripts. It is recommended that it contain no quotes and no blanks as these have not been tested and will not be supported.
Create Storage Object SQL Definitions	
The create storage object definitions of the workbook contains simple fields to store the create storage object sql and to abe able to query the dictionary to find the statements that have been stored without needing to query the text of the statements.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **storage** – Indicates a create storage objects generation directive.
Target File Name	This is the name of the file that will be created or overwritten when the generator creates the create storage objects sql. It is recommended that some form of prefix be used to identify it as the create storage objects sql for a line entry in this workbook. The proposed standard prefix is stg for ‘create storage objects sql’.
Table Owner	Storage Objects are most commonly used to store tables and indexes. Hence, to be able to find the commands used we have chosen to link the create storage object command to the table that it is most closely associated with. This is the name of the owner of the table that is to be indexed. This is for documentation and query.
Table Name	This is the name of the table that is to have a storage object created for it. This is for documentation and query.
Drop Storage Object Command	In most databases storage objects are dropped and recreatead rather than adjusted. It is expected that this cell will contain drop storage objects commands.
Create Storage Object Command	In most databases storage objects are dropped and recreatead rather than adjusted. It is expected that this cell will contain create storage objects commands.

4.1.36.Create Access Permission

The SeETL[DT] Workbook has now been updated to be able to store create access permission commands.

All databases have a variety of commands to grant various access permissions. All database managers have implemented robust security models. Some are more robust and intricate than others but all databases can protect the data held within it quite securely.

Since these permissions need to be explicitly created and submitted against the database it makes sense to store them and to publish them in reports outside the database itself. In this way the security models implemented in the databases can be made transparent and clear.

Because all the different databases have different formats for the create access permissions commands we did not try and generate access permissions commands for each database type. We simply store the access permissions commands and place them into tables in the dictionary.

The ***SeETL[DT] Create Access Permission*** tab contains the following fields.

Target Directory	
The first row of the spreadsheet defines directory that the sql statements will be written to.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **dir** – Indicating that the row will contain the name of the target directory for the table scripts.
Target Directory	• This field contains the name of the target directory for the create permissions sql scripts. It is recommended that it contain no quotes and no blanks as these have not been tested and will not be supported.
Create Access Permission SQL Definitions	
The create access permissions definitions of the workbook contains simple fields to store the create access permissions sql and to abe able to query the dictionary to find the statements that have been stored without needing to query the text of the statements.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **access** – Indicates a create access permissions generation directive.
Target File Name	This is the name of the file that will be created or overwritten when the generator creates the create access permissions sql. It is recommended that some form of prefix be used to identify it as the create access permissions sql for a line entry in this workbook. The proposed standard prefix is accs for ‘create access permissions sql’.
Table Owner	Access Permissions are most usually granted to a group or to a table or set of tables via an owner name. It is, therefore, not a simple matter to define some fields that give good query capability to determine how to find the access permission statements in the dictionary. We have decided to stay with table owner and table name for the query fields for access permissions as this is the primary type of object to which permissions are granted. This is the name of the owner of the table that access is to be granted to or some proxy for said table. This is for documentation and query.
Table Name	This is the name of the table that access is to be granted to or some proxy for said table. This is for documentation and query.
Drop Access Permission Command	In most databases access permissions are dropped and recreatead rather than adjusted. It is expected that this cell will contain drop access permissions commands.
Create Storage Object Command	In most access permissions are dropped and recreatead rather than adjusted. It is expected that this cell will contain create access permissions commands.

4.1.37.Create Generic Object

The SeETL[DT] Workbook has now been updated to be able to store create generic objects commands.

We have attempted to upgrade the workbook to store as many objects as are commonly found in the major relational databases. However, some databases have some other objects that are unique. Further? History tells us that some new forms of objects will come out in the future.

Hence we have chosen to create a sheet for those objects that are not explicitly covered in the list of other objects supported. These are just labeled 'generic objects'.

Because all the different databases have different formats for the create generic objects commands we did not try and generate create generic objects commands for each database type. We simply store the create generic objects commands and place them into tables in the dictionary.

The ***SeETL[DT] Create Generic Object*** tab contains the following fields.

Target Directory	
The first row of the spreadsheet defines directory that the sql statements will be written to.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **dir** – Indicating that the row will contain the name of the target directory for the table scripts.
Target Directory	• This field contains the name of the target directory for the create generic object sql scripts. It is recommended that it contain no quotes and no blanks as these have not been tested and will not be supported.
Create Generic Object SQL Definitions	
The create generic object definitions of the workbook contains simple fields to store the create generic object sql and to abe able to query the dictionary to find the statements that have been stored without needing to query the text of the statements.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **generic** – Indicates a create generic object generation directive.
Target File Name	This is the name of the file that will be created or overwritten when the generator creates the create generic objects sql. It is recommended that some form of prefix be used to identify it as the create generic objects sql for a line entry in this workbook. The proposed standard prefix is geno for 'create generic objects sql'.
Object Type	At the time of writing this manual we can have no idea of what generic object might be created. Hence we are creating a text field to store the type of object that is being created with this command so that it may be later retrieved for reporting of the various commands that exist in the dictionary to create generic objects.
Object Owner	We are assuming that most objects created by such statements will have an object owner or, as a minimum, a surrogate owner. This is for documentation and query.
Object Name	We are assuming that most objects created by such statements will have an object name or, as a minimum, a surrogate name that the object can be linked to for reporting purposes. This is for documentation and query.
Drop Generic Object Command	It is expected that most generic objects are dropped and recreatead rather than adjusted. It is expected that this cell will contain drop generic objects commands.
Create Generic Object Command	It is expected that most generic objects are dropped and recreatead rather than adjusted. It is expected that this cell will contain create generic objects commands.

4.1.38.SQL Statement Load

The SeETL[DT] Workbook has now been updated to be able to load SQL Statements into a target table to be processed at a later date by the scheduler. This is a ***MAJOR*** improvement in security and auditability.

The idea behind this feature is this. We upgraded the workbook to be able to generate SQL as the ETL subsystem. This left us with HUNDREDS of SQL fragments that need to be managed. Some clients have repositories to maintain these snippets of SQL and some do not.

Those that do not have a repository to manage all these snippets of code are then faced with the normal security and auditability problems of text source code. We placed many comments such as userid, machine name, timestamps etc at the front of each snippet of code. This helps to identify the person who generated the SQL. However it does not version the SQL and give the client a chance to 'back out to a prior version' or things like this.

To provide versioning, increased security and auditability we are now placing the SQL generated into a table. SeETL[DT] will always turn off the 'current flag' for the existing statement and then insert the new SQL statement into the database.

Since SQL Server can only support fields of 8000 bytes the SQL Statement is broken into up to 4 pieces of 7,900 bytes each. This allows for long statements.

Each time the statement is loaded from a file the old version will be closed off and the new version will be written into the database with a timestamp as well as details of who made the change to the statement. Later, when the statement is executed the first 8,000 bytes of the statement is logged along with the timestamp. In this way it is possible to know exactly what statement was executed at any given time save for malicious efforts to do otherwise by the DBA of the production database.

The ***SeETL[DT] SQL Statement Load*** tab contains the following fields.

SeETL[DT] SQL Statement Loading Directives	
The worksheet for loading SQL Statements is very simple. It is merely a list of directory names and file names and a flag indicating if it should be loaded or not.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Directory Name	The name of the directory visible to SeETL[DT] that the SQL file resides in. It is recommended to not contain blanks in the directory name. The directory name is to be enclosed in double quotes as follows: "C:\temp\dmltype1"
File Name	The name of the file visible to SeETL[DT] that the SQL file resides in. It is recommended to not contain blanks in the file name. The file name is to be enclosed in double quotes as follows: "dim1p_z01_vm_address_01_dim1s_01.sql"
Load Flag	This is a flag set to "Y" or "N" to indicate if SeETL[DT] is being directed to load the piece of SQL in the associated File.

The table that the SQL Statements are loaded into is as follows:

```
create table dbo.ctl_dw_sql_stmts (
        pk_spreadsheet_name          varchar (255)                default 'unknown'
       ,pk_spreadsheet_version       varchar (255)                default 'unknown'
       ,pk_row_number                integer                      not null default 0
       ,pk_load_timestamp            datetime                     not null   default current_timestamp
       ,current_flag                 integer                      not null default 0
       ,directory_name               varchar (4000)               not null default 'unknown'
       ,sql_file_name                varchar (4000)               not null default 'unknown'
       ,full_sql_file_name           varchar (4000)               not null default 'unknown'
       ,sql_file_win_user            varchar (255)                default 'unknown'
       ,sql_file_win_workstation     varchar (255)                default 'unknown'
       ,sql_file_win_domain          varchar (255)                default 'unknown'
       ,sql_file_os_version          varchar (255)                default 'unknown'
       ,sql_file_seetl_current_directory varchar (4000)           default 'unknown'
       ,sql_file_dot_net_version     varchar (255)                default 'unknown'
       ,sql_file_seetl_version_date         varchar (255)                default 'unknown'
       ,sql_statement_01             varchar (8000)               not null default 'unknown'
       ,sql_statement_02             varchar (8000)               not null default 'unknown'
       ,sql_statement_03             varchar (8000)               not null default 'unknown'
       ,sql_statement_04             varchar (8000)               not null default 'unknown'
)
```

The table shows the fields that are obviously filled in when the statement itself is loaded into the database. The full sql file name is redundantly stored to make finding the row in the execution program that much simpler.

The table that logs the execution of the SQL Statement is as follows:

```
create table dbo.ctl_dw_sql_stmt_log (
        pk_spreadsheet_name          varchar (255)                default 'unknown'
       ,pk_spreadsheet_version       varchar (255)                default 'unknown'
       ,pk_row_number                integer                      not null default 0
       ,pk_load_timestamp           datetime                     not null   default current_timestamp
       ,pk_batch_number             integer                      not null
       ,pk_start_stmt_timestamp     datetime                     not null   default current_timestamp
       ,end_stmt_timestamp          datetime                     not null   default current_timestamp
       ,directory_name               varchar (4000)               not null default 'unknown'
       ,sql_file_name                varchar (4000)               not null default 'unknown'
       ,full_sql_file_name           varchar (4000)               not null default 'unknown'
       ,number_rows_affected         integer                      not null default 0
       ,sql_code                     integer                      not null default 0
       ,sql_statement                varchar (8000)               not null default 'unknown'
)
```

It can be clearly seen that the start timestamp, stop timestamp and number of rows processed is recorded in the log. The key to the source statement is also stored so that the link back to the piece of SQL that was loaded is stored inside the database.

This is ***VERY IMPORTANT.*** What this means is that the client can build their entire ETL subsystem out of SQL fragments. This can include such things as dropping and recreating indexes, altering permissions, dropping and creating objects etc and AUDIT AND LOG all activity into the execution database which may be different to the staging database and the target database.

The reason this is so important is that this means that the clients now have a viable alternative to ETL tools that reside on different servers. The new SeETL^{DT} provides the opportunity to push all processing into the database and the only thing that needs to be outside the database is the C++ Scheduler and the C++ program that executes the SQL statements.

ALL other work can be done inside the database. Because it is done inside the database all the skills of the DBAs and all the investment by the vendor in database technology can be leveraged. The database vendors spend many times more money and effort on the database than the ETL vendors spend on their tools because the revenues possible are vastly different.

Further. There are MANY more people who are able to run databases than there are who can maintain ETL tool developed software. This will remain true forever.

This option will be very attractive for clients wishing to minimize their expenditure on ETL software.

4.1.39.Data Profiling Worksheet

The SeETL[DT] Workbook has now been updated to be able to generate Data Profiling SQL that can be submitted against any ASNI SQL compliant database. Data Profiling is an area where the vendors have done a relatively poor job. The engines simply don't do that much that is of value over and above what a bit of experience will bring to the table. We decided to create some simple data profiling algorithms and we will add more and more complex algorithms as we see how people start to use the simple ones.

The ***SeETL[DT] Data Profiling*** tab contains the following fields.

Tables to Create Data Profiling Statements For	
The first set of rows indicates the tables on which to create data profiling SQL Statements.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **tab** – A table row for the definition of Data Profiling SQL Statements.
Owner	This is the name of the table/view owner that is to have Data Profiling SQL generated by the Data Profiling Generation Facility.
Table	This is the name of the table/view that is to have Data Profiling SQL generated by the Data Profiling Generation Facility.
Row Count Flag	This is a field set to "Y" or "N" that indicates whether this table should have SQL generated to run and store the number of rows counted in the table in the dictionary.
Row Stats Flag	This is a field set to "Y" or "N" that indicates whether this table should have the statistical Data Profiling SQL Generated
Columns to Create Data Profiling Statements for	
Some of the Data Profiling SQL is generated at the table level and some is generated at the column level. Because the client will need to be able to specify which columns to perform data profiling on it will be necessary to specify them. It is wasteful to perform Data Profiling on columns that are known to be empty or where the contents of the columns is such that the performance of Data Profiling algorithms is not beneficial. For example text fields are not of value to perform data profiling on.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **col** – A column row for the definition of Data Profiling SQL Statements.
Table Owner	This is the name of the owner of the Table in which the column to be profiled exists.
Table Name	This is the name of the Table in which the column to be profiled exists.
Column Name	This is the name of the column to be profiled.
Column Number	This is the ordinal number of the column on the table.
Data Type	This is the data type of the column to be profiled. It is documentation only. It is not used.
Length	This is the length of the column to be profiled. It is documentation only. It is not used.
Precision	This is the precision of the column to be profiled. It is documentation only. It is not used.
Scale	This is the scale of the column to be profiled. It is documentation only. It is not used.
Null	This is a field set to "Y" or "N" that indicates whether this column can contain Nulls.
Count Unique Values Flag	This is a field set to "Y" or "N" that indicates whether SeETL[DT] should generate the Data Profiling SQL to count the number of unique values in the column.
Store Unique Values Flag	This is a field set to "Y" or "N" that indicates whether SeETL[DT] should generate the Data Profiling SQL to store the actual unique values of this field in the dictionary.
Run Stats Flag	This is a field set to "Y" or "N" that indicates whether SeETL[DT] should generate the Data Profiling SQL to perform statistical analysis on the column.
Run Xref Flag	This is a field set to "Y" or "N" that indicates whether SeETL[DT] should generate the Data Profiling SQL to generate the cross reference rows that may be found in the data that has been placed into the dictionary.

The tables that the results of data profiling are loaded into are as follows. Notice that all the rows have the name of the spreadsheet that generated them, the timestamp of the inserted row and the current flag. This allows all Data Profiling to be versioned.

The first table maintains the number of rows in tables. It is an easy way of maintaining row counts over time. This is useful in the data profiling of data for development of a project. It is also useful information to have for the staging area and data warehouse over time.

```
create table dbo.ctl_dw_dp_row_counts (
        pk_spreadsheet_name          varchar (255)               default 'unknown'
       ,pk_spreadsheet_version       varchar (255)               default 'unknown'
       ,pk_row_number                integer                     not null default 0
       ,pk_run_timestamp             datetime                    not null   default current_timestamp
       ,current_flag                 integer                     not null default 0
       ,table_owner                  varchar (255)               not null default 'unknown'
       ,table_name                   varchar (255)               not null default 'unknown'
       ,number_rows                  integer                     not null default 0
)
```

This table stored the number of unique values found in a column and not the values themselves. It is always useful to count the number of unique values and divide by the number of rows to see how much repetition there basically is in the data.

```
create table dbo.ctl_dw_dp_col_counts_01 (
        pk_spreadsheet_name          varchar (255)               default 'unknown'
       ,pk_spreadsheet_version       varchar (255)               default 'unknown'
       ,pk_row_number                integer                     not null default 0
       ,pk_run_timestamp             datetime                    not null   default current_timestamp
       ,current_flag                 integer                     not null default 0
       ,table_owner                  varchar (255)               not null default 'unknown'
       ,table_name                   varchar (255)               not null default 'unknown'
       ,column_name                  varchar (255)               not null default 'unknown'
       ,column_number                integer                     not null default 0
       ,num_unique_values            integer                     not null default 0
)
```

This table stores the ACTUAL unique values in the column. Note that the value itself is stored in a string field of 8,000 bytes. If the values in the column are unique the field ‘candidate key flag’ is set to 1 to notify the database designer that this might actually be useful as a candidate key.

```
create table dbo.ctl_dw_dp_col_counts_02 (
        pk_spreadsheet_name          varchar (255)               default 'unknown'
       ,pk_spreadsheet_version       varchar (255)               default 'unknown'
       ,pk_row_number                integer                     not null default 0
       ,pk_run_timestamp             datetime                    not null   default current_timestamp
       ,current_flag                 integer                     not null default 0
       ,table_owner                  varchar (255)               not null default 'unknown'
       ,table_name                   varchar (255)               not null default 'unknown'
       ,column_name                  varchar (255)               not null default 'unknown'
       ,column_number                integer                     not null default 0
       ,unique_value                 varchar (8000)              not null default 'unknown'
       ,num_occurrences              integer                     not null default 0
       ,candidate_key_flag           integer                     not null default 0
)
```

This table stores statistics like min, max, average, mode etc for columns. Obviously this only makes sense with numeric columns.

```
create table dbo.ctl_dw_dp_col_counts_03 (
        pk_spreadsheet_name          varchar (255)               default 'unknown'
       ,pk_spreadsheet_version       varchar (255)               default 'unknown'
       ,pk_row_number                integer                     not null default 0
       ,pk_run_timestamp             datetime                    not null   default current_timestamp
       ,current_flag                 integer                     not null default 0
       ,table_owner                  varchar (255)               not null default 'unknown'
       ,table_name                   varchar (255)               not null default 'unknown'
       ,column_name                  varchar (255)               not null default 'unknown'
       ,column_number                integer                     not null default 0
       ,col_average                  decimal (38,12)             not null default 0
       ,col_mode                     decimal (38,12)             not null default 0
       ,col_min                      decimal (38,12)             not null default 0
       ,col_max                      decimal (38,12)             not null default 0
       ,col_std_deviation            decimal (38,12)             not null default 0
)
```

The final table is trying to match different columns in different tables to guess at joins at the data exploration time. The query to populate this table is, of course, dependend on actually storing all the unqiue values in the dictionary. Such information on possible joins is some of the most useful information that can be gleaned from the data itself.

```
create table dbo.ctl_dw_dp_col_counts_04 (
        pk_spreadsheet_name          varchar (255)               default 'unknown'
       ,pk_spreadsheet_version       varchar (255)               default 'unknown'
       ,pk_row_number                integer                     not null default 0
       ,pk_run_timestamp             datetime                    not null   default current_timestamp
       ,current_flag                 integer                     not null default 0
       ,left_table_owner             varchar (255)               not null default 'unknown'
       ,left_table_name              varchar (255)               not null default 'unknown'
       ,left_column_name             varchar (255)               not null default 'unknown'
       ,left_column_number           integer                     not null default 0
       ,left_unique_value            varchar (255)               not null default 'unknown'
       ,left_num_occurrences         integer                     not null default 0
       ,left_candidate_key_flag      integer                     not null default 0
       ,right_table_owner            varchar (255)               not null default 'unknown'
       ,right_table_name             varchar (255)               not null default 'unknown'
       ,right_column_name            varchar (255)               not null default 'unknown'
       ,right_column_number          integer                     not null default 0
       ,right_num_occurrences        integer                     not null default 0
       ,right_candidate_key_flag     integer                     not null default 0
)
```

The SQL that gets generated by the SeETL[DT] is as follows.

This is the update of stats on an SQL Server database.

```
update statistics dbo.in_customer_dim with fullscan , all
;
```

Each insert statement is preceeded by the appropriate update statement to turn off the current flag. We will not copy all these into this documentation as it is simple to generate such examples.

```
update ctl_dw_dp_row_counts set current_flag = 0
where  1=1
and pk_spreadsheet_name        = 'SeETLDataProfilingV3.1.00.XML'
and pk_spreadsheet_version     = 'V3.1.00'
and current_flag               = 1
and table_owner                = 'dbo'
and table_name                 = 'in_customer_dim'
;
```

SQL to count the number of rows in a table.

```
insert into ctl_dw_dp_row_counts
(
 pk_spreadsheet_name
,pk_spreadsheet_version
,pk_row_number
,pk_run_timestamp
,current_flag
,table_owner
,table_name
,number_rows
)
select
 'SeETLDataProfilingV3.1.00.XML'
,'V3.1.00'
, 0
, current_timestamp
, 1
,'dbo'
,'in_customer_dim'
, count(*)
from dbo.in_customer_dim
;
```

SQL to count the number of specific occurrences of a piece of data in a table.

```
insert into ctl_dw_dp_col_counts_01
(
 pk_spreadsheet_name
,pk_spreadsheet_version
,pk_row_number
,pk_run_timestamp
,current_flag
,table_owner
,table_name
,column_name
,column_number
,num_unique_values
)
select
 'SeETLDataProfilingV3.1.00.XML'
,'V3.1.00'
, 0
, current_timestamp
, 1
,'dbo'
,'in_customer_dim'
,'cust_code'
, 1
, count (distinct cust_code)
from dbo.in_customer_dim
;
```

SQL to actually store the unique values in a column.

```
insert into ctl_dw_dp_col_counts_02
(
 pk_spreadsheet_name
,pk_spreadsheet_version
,pk_row_number
,pk_run_timestamp
,current_flag
,table_owner
,table_name
,column_name
,column_number
,unique_value
,num_occurrences
,candidate_key_flag
)
select
 'SeETLDataProfilingV3.1.00.XML'
,'V3.1.00'
, 0
, current_timestamp
, 1
,'dbo'
,'in_customer_dim'
,'cust_code'
, 1
, cust_code
, count (*)
, 0
from dbo.in_customer_dim
group by cust_code
```

SQL to set the candidate key flag if the value only occurs once in the column.

```
update ctl_dw_dp_col_counts_02 set candidate_key_flag = 1
where  1=1
and pk_spreadsheet_name        = 'SeETLDataProfilingV3.1.00.XML'
and pk_spreadsheet_version     = 'V3.1.00'
and current_flag               = 1
and table_owner                = 'dbo'
and table_name                 = 'in_customer_dim'
and column_name                = 'cust_code'
and candidate_key_flag         = 0
and num_occurrences            = 1
```

SQL to perform the generation of statistics for a column.

```
insert into ctl_dw_dp_col_counts_03
(
 pk_spreadsheet_name
,pk_spreadsheet_version
,pk_row_number
,pk_run_timestamp
,current_flag
,table_owner
,table_name
,column_name
,column_number
,col_average
,col_mode
,col_min
,col_max
,col_std_deviation
)
select
 'SeETLDataProfilingV3.1.00.XML'
,'V3.1.00'
, 6
, current_timestamp
, 1
,'dbo'
,'in_customer_dim'
,'cust_household'
, 7
, avg(cust_household)
, 0
, min(cust_household)
, max(cust_household)
, stdevp(cust_household)
from dbo.in_customer_dim
```

SQL to generate the cross matching of columns to attempt to guess at possible joins from the data that was captured into the dictionary earlier in the Data Profiling run.

```
insert into ctl_dw_dp_col_counts_04
(
 pk_spreadsheet_name
,pk_spreadsheet_version
,pk_row_number
,pk_run_timestamp
,current_flag
,left_table_owner
,left_table_name
,left_column_name
,left_column_number
,left_unique_value
,left_num_occurrences
,left_candidate_key_flag
,right_table_owner
,right_table_name
,right_column_name
,right_column_number
,right_num_occurrences
,right_candidate_key_flag
)
select
 a.pk_spreadsheet_name
,a.pk_spreadsheet_version
,a.pk_row_number
, current_timestamp
,1
,a.table_owner
,a.table_name
,a.column_name
,a.column_number
,a.unique_value
,a.num_occurrences
,a.candidate_key_flag
,b.table_owner
,b.table_name
,b.column_name
,b.column_number
,b.num_occurrences
,b.candidate_key_flag
from ctl_dw_dp_col_counts_02 a , ctl_dw_dp_col_counts_02 b
where 1=1
and a.pk_spreadsheet_name                   = b.pk_spreadsheet_name
and a.pk_spreadsheet_version                = b.pk_spreadsheet_version
and a.pk_row_number                         = b.pk_row_number
and a.table_owner                           = b.table_owner
and not (a.table_name                       = b.table_name)
and a.unique_value                          = b.unique_value
and a.current_flag                          = 1
and b.current_flag                          = 1
and a.pk_spreadsheet_name                   = 'SeETLDataProfilingV3.1.00.XML'
and a.pk_spreadsheet_version                = 'V3.1.00'
and a.table_owner                           = 'dbo'
and a.table_name                            = 'in_customer_dim'
and a.column_name                           = 'cust_code'
;
```

4.1.40. Document Reports

The SeETL[DT] Workbook has now been updated to be able to store documentation about reports that are deployed across the enterprise.

One of the biggest issues in any large enterprise is that there are hundreds if not thousands of electronic reports that are produced. These reports are often produced on different reporting technologies. They are documented poorly, if at all, and usually only the source code used to create the report in the reporting tool is documented.

The concepts behind 'self documenting reports' that were used in www.meta5.com have not been widely adopted by the leading reporting vendors Cognos, Business Objects, MicroStrategy and Microsoft. Further, these four vendors, and you can add to them Oracle, have failed to co-operate on integration of reporting information for many years now. There seems no possibility that they will co-operate any time soon.

As a result of this there is a dire need to be able to document reports across an enterprise.

We have created a small set of dictionary tables to provide such a tool should our clients wish to use it. The idea is simply that all the reports and the data on the reports can be captured via one or more workbooks and then placed into the dictionary. Capturing this information into Excel sheets is as easy as anything else that might be done. Reports will be developed in Report Services as time permits. However, we felt it was best to get this tool out into the public domain as soon as possible even though we have not performed report development yet.

The ***SeETL[DT] Document Reports*** tab contains the following fields.

Number of Languages	
The first row of the spreadsheet defines the number of languages that are being used to document the reports.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **nl** – Number of Languages.
Number of Languages	This field defines the number of languages that are being used to document the reports.
Report Definitions and Short Descriptions	
The report descriptions that are documented and defined in this workbook have short and long descriptions in up to 10 languages defined for them. This set of rows defines the base information and short descriptions for the reports.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **rptsd** – Report Description Short Description
Report Short Name	This is a short name that is assigned to a report. Usually it is about an 8 to 12 character string that contains some indication of the system/application, purpose and number of the report. It is usual to assign some form of report short name so that the technology people can use that name for quick and simple communication of which report they are talking about.
Report Long Name	This is a long name that is assigned to a report. This is the name by which the report is generally known to the business community. It often contains the report short name for clarity.
Submodel	The ***BI4ALL Data Models*** are implemented in submodels. Very frequently reports are associated with a single sub model. This sub model gives the opportunity to link the report to the submodel should that be the actual case.
Report Number	It is often the case that a report is also given a number. Should the client choose to give reports numbers as well as short names that number can be stored in this cell. It is considered a string field even if it contains purely numeric characters.
Frequency	This is the typical frequency with which the report will be executed.
Report Owner	This is the Business Owner of the report. It is stored as a text field.

Tech Support	This is the person or department that is responsible for maintaining this report as 'Tech Support'.
Reporting Tool	This is a text field that describeds the tool that was used to create this report. Examples are "Gognos", "ReportServices", "BusinessObjects" etc.
Report Description Lan 01	This is the description of the Report in the 01 language.
Report Description Lan 02	This is the description of the Report in the 02 language.
Report Description Lan 03	This is the description of the Report in the 03 language.
Report Description Lan 04	This is the description of the Report in the 04 language.
Report Description Lan 05	This is the description of the Report in the 05 language.
Report Description Lan 06	This is the description of the Report in the 06 language.
Report Description Lan 07	This is the description of the Report in the 07 language.
Report Description Lan 08	This is the description of the Report in the 08 language.
Report Description Lan 09	This is the description of the Report in the 09 language.
Report Description Lan 10	This is the description of the Report in the 10 language.

Report Definitions and Long Descriptions

The Reports that are documented and defined in this workbook have short and long descriptions in up to 10 languages defined for them. This set of rows defines the long descriptions for the Reports.

Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **rptId** – Report Description Long Description
Report Short Name	This is a short name that is assigned to a report. Usually it is about an 8 to 12 character string that contains some indication of the system/application, purpose and number of the report. It is usual to assign some form of report short name so that the technology people can use that name for quick and simple communication of which report they are talking about.
Report Long Name	This is a long name that is assigned to a report. This is the name by which the report is generally known to the business community. It often contains the report short name for clarity.
Report Description Lan 01	This is the long description of the Report in the 01 language.
Report Description Lan 02	This is the long description of the Report in the 02 language.
Report Description Lan 03	This is the long description of the Report in the 03 language.
Report Description Lan 04	This is the long description of the Report in the 04 language.
Report Description Lan 05	This is the long description of the Report in the 05 language.
Report Description Lan 06	This is the long description of the Report in the 06 language.
Report Description Lan 07	This is the long description of the Report in the 07 language.
Report Description Lan 08	This is the long description of the Report in the 08 language.
Report Description Lan 09	This is the long description of the Report in the 09 language.
Report Description Lan	This is the long description of the Report in the 10 language.

10	

Report Input Parameters

To be able to properly use a Report the user using the report must usually set some input parameters. There may be defaults but in general there are pull downs or drills for input parameters. This section documents the input parameters of the report in enough depth that the end user has a reasonable chance to be able to properly invoke the report when new to the report.

Field Name	Description
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **rptip** – Report Input Parameters.
Type	This defines the type of the row that the SeETL[DT] software will interpret the row as. There are some 'helper' values in here to make the code a little easier to write. The valid values are: • **b** – Begin. Each report should have a 'b' on the first line to warn the generator that it is the beginning of a new report definition. • **m** – Move. This is the 'normal' or default value of a row type. • **e** – End. Each report is required to have an 'end' line. This is just to be consistent with all the other worksheets like this.
Report Short Name	This is a short name that is assigned to a report. Usually it is about an 8 to 12 character string that contains some indication of the system/application, purpose and number of the report. It is usual to assign some form of report short name so that the technology people can use that name for quick and simple communication of which report they are talking about.
Report Long Name	This is a long name that is assigned to a report. This is the name by which the report is generally known to the business community. It often contains the report short name for clarity.
Parameter Column Name	This is the name of the parameter that is being used to filter the report. It may be the name on the screen/report that is visible to the user. It might just be a column name that is used that is not visible to the user.
Parameter Column Data Type	This is documentation of the Parameter Column Data Type. It explains to the user the data tpe of the parameter. In most cases parameters are strings. On rare occasions they may be numerics. In some cases dates are used and can be selected from calendars that pop up to constrain the reports. It all depends on how the reporting tools will be implemented for the client.
Parameter Column Short help text Lang01	This is the short help text of the parameter column in language 01.
Parameter Column Long help text Lang01	This is the long help text of the parameter column in language 01.
Parameter Column Short help text Lang02	This is the short help text of the parameter column in language 02.
Parameter Column Long help text Lang02	This is the long help text of the parameter column in language 02.
Parameter Column Short help text Lang03	This is the short help text of the parameter column in language 03.
Parameter Column Long help text Lang03	This is the long help text of the parameter column in language 03.
Parameter Column Short help text Lang04	This is the short help text of the parameter column in language 04.
Parameter Column Long help text Lang04	This is the long help text of the parameter column in language 04.
Parameter Column Short help text Lang05	This is the short help text of the parameter column in language 05.
Parameter Column Long help text Lang05	This is the long help text of the parameter column in language 05.
Parameter Column Short help text Lang06	This is the short help text of the parameter column in language 06.
Parameter Column Long help text Lang06	This is the long help text of the parameter column in language 06.

Parameter Column Short help text Lang07	This is the short help text of the parameter column in language 07.
Parameter Column Long help text Lang07	This is the long help text of the parameter column in language 07.
Parameter Column Short help text Lang08	This is the short help text of the parameter column in language 08.
Parameter Column Long help text Lang08	This is the long help text of the parameter column in language 08.
Parameter Column Short help text Lang09	This is the short help text of the parameter column in language 09.
Parameter Column Long help text Lang09	This is the long help text of the parameter column in language 09.
Parameter Column Short help text Lang10	This is the short help text of the parameter column in language 10.
Parameter Column Long help text Lang10	This is the long help text of the parameter column in language 10.

Report Output Fields

To be able to properly document reports we must document all output fields. The most important part of the report documentation is the attempt to be able to link the field on the report all the way back to the source field that the report field came from. And this attempt includes even being able to perform that linkage for more complex fields and derivations.

Field Name	Description
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **rptof** – Report Output Field.
Type	This defines the type of the row that the SeETL[DT] software will interpret the row as. There are some 'helper' values in here to make the code a little easier to write. The valid values are: • **b** – Begin. Each report should have a 'b' on the first line to warn the generator that it is the beginning of a new report. • **m** – Move. This is the 'normal' or default value of a row type. • **e** – End. Each report is required to have an 'end' line to tell the generator this is the end of the report. This is done to make it consistent with other worksheets.
Report Short Name	This is a short name that is assigned to a report. Usually it is about an 8 to 12 character string that contains some indication of the system/application, purpose and number of the report. It is usual to assign some form of report short name so that the technology people can use that name for quick and simple communication of which report they are talking about.
Report Long Name	This is a long name that is assigned to a report. This is the name by which the report is generally known to the business community. It often contains the report short name for clarity.
Output Column Name	This is the name of the output field that is placed onto the report. It may be something like the field name from a stored procedure or the object name in a business objects universe. It must be remembered that this is documentation only and that what should be placed into this cell is the name that is most beneficial to the client.
Output Column Data Type	This is documentation of the Output Column Data Type. It explains to the user the data tpe of the output field. In most cases output columns are strings on reports. They may be able to be converted to things like dates and numerics by an 'export to' function. For example many reporting tools can export reports to Microsofts Excel.
Data Model Table	If the field is the direct placement of a field from the data base onto the report then the data model table is named in this cell.
Data Model Field	If the field is the direct placement of a field from the data base onto the report then the data model field is named in this cell.
Output Column Short help text Lang01	This is the short help text of the Output column in language 01.
Output Column Long help text Lang01	This is the long help text of the Output column in language 01.
Output Column Short	This is the short help text of the Output column in language 02.

help text Lang02	
Output Column Long help text Lang02	This is the long help text of the Output column in language 02.
Output Column Short help text Lang03	This is the short help text of the Output column in language 03.
Output Column Long help text Lang03	This is the long help text of the Output column in language 03.
Output Column Short help text Lang04	This is the short help text of the Output column in language 04.
Output Column Long help text Lang04	This is the long help text of the Output column in language 04.
Output Column Short help text Lang05	This is the short help text of the Output column in language 05.
Output Column Long help text Lang05	This is the long help text of the Output column in language 05.
Output Column Short help text Lang06	This is the short help text of the Output column in language 06.
Output Column Long help text Lang06	This is the long help text of the Output column in language 06.
Output Column Short help text Lang07	This is the short help text of the Output column in language 07.
Output Column Long help text Lang07	This is the long help text of the Output column in language 07.
Output Column Short help text Lang08	This is the short help text of the Output column in language 08.
Output Column Long help text Lang08	This is the long help text of the Output column in language 08.
Output Column Short help text Lang09	This is the short help text of the Output column in language 09.
Output Column Long help text Lang09	This is the long help text of the Output column in language 09.
Output Column Short help text Lang10	This is the short help text of the Output column in language 10.
Output Column Long help text Lang10	This is the long help text of the Output column in language 10.

4.1.41.Project Plan

The SeETL[DT] Workbook has now been updated to be able to store project plans for implementation projects that can be linked to the forthcoming ***IBI Methodology, BI4ALL Data Models*** and SeETL tools.

All development projects require some form of Project Plan to be completed. MS Project has proven to be a very poor tool to use over a very long period. Most clients have some form of Project Management tool that they use and that they ask us to also use. Because of this there has been little incentive for us to build any sort of Project Planning tools.

However. Now seems to be an appropriate time to define an IBI Project Plan and to build a relatively simple project planning tool.

In the early releases this will be little more than the ability to store the Project Plan in a spreadsheet and load it up into the dictionary for reporting purposes. The reports will continue to be built in MS Report Services. Other tools may be used for reporting as well.

The ***IBI Project Plan*** tab contains the following fields.

Number of Persons	
The first row of the spreadsheet defines the number of Persons allocated to the project.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **np** – Number of Persons.
Number of Persons	This field defines the number of persons that will be assigned to the project.
Persons Assigned to Project Descriptions	
The project plan must link the Persons and the Projects. This first set of fields defines the Persons assigned to the Project.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **patp** – Person Assigned to Project Description
Person Key	This is a string key that will be used to link the Person to other rows in the dictionary.
Person Initials	These are the initials of the person to be used on reports.
Person Name	This is the name of the person to be used on reports.
Start Date	This is the start date of the persons availability for the project. It is required to be in Quotes and is in the format “dd/mm/yyyy”.
End Date	This is the end date of the persons availability for the project. It is required to be in Quotes and is in the format “dd/mm/yyyy”.
Percentage Available	This is the percentage of time in the available period that the person is available. It is usually availability on a weekly basis that is measured on this field but could be weekly or monthly.
Project Role Descriptions	
The Persons working on the Project work in various roles. These are described in this section.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **pr** – Project Role Description
Role Key	This is a string key that will be used to link the Project Role to other rows in the dictionary.
Role Name	This is the name of the role that will appear on reports.
Role Short Description	This is the short description of the Project Role.
Role Long Description	This is the long description of the Project Role.

Start Date	This is the start date of the role on the project. It is required to be in Quotes and is in the format “dd/mm/yyyy”.
End Date	This is the end date of the role on the project. It is required to be in Quotes and is in the format “dd/mm/yyyy”.
Percentage Time Required	This is the percentage of time in the role is required to be worked for this project. Some roles are part time such as DBA resource. Some are full time but split between people.
Project Task Groups Description	
The Project Plan is made up of tasks. However these tasks can be grouped up into Project Task Groups so that the groups can be managed and closed off as a group.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **ptg** – Project Task Group Description.
Group Number	This is the Project Task Group Number which will be used as a string key to link it to other rows in the dictionary.
Group Number String	This is the Project Task Group Number which will be used as a string key to link it to other rows in the dictionary. It is intended as a string that can contain periods in the number so that the number is more easily read. It is a reformatting of the Group Number.
Group Name	This is the Project Task Group Name. It is a descriptive field that will be used on reports.
Plan Start Date	This is the planned start date of the Project Task Group. It is required to be in Quotes and is in the format “dd/mm/yyyy”.
Plan End Date	This is the planned end date of the Project Task Group. It is required to be in Quotes and is in the format “dd/mm/yyyy”.
Actual Start Date	This is the actual start date of the Project Task Group. It is required to be in Quotes and is in the format “dd/mm/yyyy”.
Actual End Date	This is the actual end date of the Project Task Group. It is required to be in Quotes and is in the format “dd/mm/yyyy”.
Group Status	This is the current status of the Project Task Group. It is a descriptive field that will be used on reports.
Group Percentage Complete	This is the current Project Task Group Percentage Complete. It is used to track progress and completion of Project Task Groups.
Group Owner Person Key	This is the link to the Person Assigned to Project Table to retrieve the details of the Person who is the owner of this Project Task Group.
Group Owner Initials	These are the initials of the Person who is the Task Group Owner for easier identification on the workbook. It is redundantly stored to make it easier to know who the Group Owner Person Key links to.
Group Effort in Hours	This is the number of hours that it is expected to take for the Task Group to be completely.
Group Level	Task Groups can be made into Hierarchies to more effectively manage the work that needs to be done. The Group Level field is an integer field that allows the group to participate in a Hierarchy. It is numeric.
Signatories as Text	Some Task Groups will need to be signed off by signatories. This field contains the initials of the signatories to the task group should the task group require signatories. If no signatories are required then “NA” is the recommended value.
Group Detailed Description	This is space for a detailed description of the Task Group. It is for reporting purposes only.
Link Key	This is a string field to enable the Task Group Row to be linked to some other table via this field. It is user definable and the client is free to place any string key into this field he chooses.
Estimated Internal Cost	This is the estimated cost in the clients currency units that it is estimated this task group is going to consume. It is a whole number.
Estimated External Revenue	This is the estimated external revenue in the clients currency units that will be generated for this task group if the client is something like a consulting firm that is performing tasks for their client. It is a whole number.
Actual Internal Cost	This is the actual cost in the clients currency units that were consumed for the performance of this task group. It is a whole number.
Actual External	This is the actual external revenue in the clients currency units that was generated for

Revenue	this task group if the client is something like a consulting firm that is performing tasks for their client. It is a whole number.
Invoiced Flag	This is a flag that can be set to “Y” or “N” to indicate whether this Task Group has been invoiced to the end client.
Paid Flag	This is a flag that can be set to “Y” or “N” to indicate whether this Task Group has been invoiced to the end client and the end client has paid this amount to the bank of the client.
Project Task Description	
The Project Plan is made up of tasks. This section documents each task that will be managed at the task level for the project.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **pt** – Project Task Description.
Task Number	This is the Project Task Number which will be used as a string key to link it to other rows in the dictionary.
Task Number String	This is the Project Task Number which will be used as a string key to link it to other rows in the dictionary. It is intended as a string that can contain periods in the number so that the number is more easily read. It is a reformatting of the Task Number.
Group Number	This is the Project Task Group Number which will be used as a string key to link it to other rows in the dictionary.
Task Name	This is the Project Task Name. It is a descriptive field that will be used on reports.
Plan Start Date	This is the planned start date of the Project Task. It is required to be in Quotes and is in the format “dd/mm/yyyy”.
Plan End Date	This is the planned end date of the Project Task. It is required to be in Quotes and is in the format “dd/mm/yyyy”.
Actual Start Date	This is the actual start date of the Project Task. It is required to be in Quotes and is in the format “dd/mm/yyyy”.
Actual End Date	This is the actual end date of the Project Task. It is required to be in Quotes and is in the format “dd/mm/yyyy”.
Task Status	This is the current status of the Project Task. It is a descriptive field that will be used on reports.
Task Percentage Complete	This is the current Project Task Percentage Complete. It is used to track progress and completion of Project Tasks.
Task Owner Person Key	This is the link to the Person Assigned to Project Table to retrieve the details of the Person who is the owner of this Project Task.
Task Owner Initials	These are the initials of the Person who is the Task Owner for easier identification on the workbook. It is redundantly stored to make it easier to know who the Task Owner Person Key links to.
Task Effort in Hours	This is the number of hours that it is expected to take for the Task to be completely.
Deliverable from Task Flag	This is a flag that can be set to “Y” or “N” to indicate whether this Task is expected to produce a specific deliverable.
Deliverable Number	If this task has a deliverable to be produced as a result of performing the task then this is the deliverable number that will be linked to the Deliverables section of the workbook below.
Deliverable Name	If this task has a deliverable to be produced as a result of performing the task then this is the deliverable name that will be linked to the Deliverables section of the workbook below.
Deliverable Sign Off Flag	This is a flag that can be set to “Y” or “N” to indicate whether this Task is expected to produce a specific deliverable and whether that deliverable is required to be signed off by specific people.
Signatories as Text	Some Tasks will need to be signed off by signatories. This field contains the initials of the signatories to the task should the task require signatories. If no signatories are required then “NA” is the recommended value.
Task Detailed Description	This is space for a detailed description of the Task. It is for reporting purposes only.
Link Key	This is a string field to enable the Task Row to be linked to some other table via this field. It is user definable and the client is free to place any string key into this field he chooses.

Estimated Internal Cost	This is the estimated cost in the clients currency units that it is estimated this task is going to consume. It is a whole number.
Estimated External Revenue	This is the estimated external revenue in the clients currency units that will be generated for this task if the client is something like a consulting firm that is performing tasks for their client. It is a whole number.
Actual Internal Cost	This is the actual cost in the clients currency units that were consumed for the performance of this task. It is a whole number.
Actual External Revenue	This is the actual external revenue in the clients currency units that was generated for this task if the client is something like a consulting firm that is performing tasks for their client. It is a whole number.
Invoiced Flag	This is a flag that can be set to "Y" or "N" to indicate whether this Task has been invoiced to the end client.
Paid Flag	This is a flag that can be set to "Y" or "N" to indicate whether this Task has been invoiced to the end client and the end client has paid this amount to the bank of the client.

Person Assigned To Tasks

In prior sheet the Persons have been defined and the Tasks have been defined. Now we need to link the Persons to the Tasks. There may be many Persons assigned to one Task and many tasks may be assigned to one Person. There is an obvious many to many relationship between Persons and Tasks.

Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **patt** – Person Assigned To Task Description.
Person Key	This is a string key that will be used to link the Person to other rows in the dictionary.
Person Initials	These are the initials of the person to be used on reports.
Person Name	This is the name of the person to be used on reports.
Task Number	This is the Project Task Number which will be used as a string key to link it to other rows in the dictionary.
Role Key	This is a string key that will be used to link the Project Role to other rows in the dictionary.
Role Short Description	This is the short description of the Project Role.
Start Date	This is the start date of the Person on the Task. It is required to be in Quotes and is in the format "dd/mm/yyyy".
End Date	This is the end date of the Person on the Task. It is required to be in Quotes and is in the format "dd/mm/yyyy".
Percentage Assigned	This is the percentage of time in the Person is assigned to the Role for this task. It is a whole number.
Notes for the Assignment	This is a free format text field for documentation purposes should it be needed.

Task Dependencies

Tasks very frequently have sets of dependencies. This tool is not a project management tool in that it does not have an active engine to build project plans based on dependencies. There are a multitude of such project planning tools out there. The trouble is that none of them do a very good job of highly iterative projects and they are more trouble than they are worth in that area. The task dependencies in this sheet are for documentation and communication purposes only. The dates and other items are required to be maintained by hand in this simple project management tool.

Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **td** – Task Dependency Description.
Task Number Pre Dependency	This is the string key of the Task that is the "pre" or left hand side dependency. Since the dependency may be 'co-dependency' it is not strictly true that this is always a pre-dependency.
Pre Dependency Task Name	This is the name of the task that is the "pre" or left hand side dependency. It is redundantly included so as to make the workbook more meaningful to read.
Task Number Post Dependency	This is the string key of the Task that is the "post" or right hand side dependency. Since the dependency may be 'co-dependency' it is not strictly true that this is always a post-

	dependency.
Post Dependency Task Name	This is the name of the task that is the “post” or right hand side dependency. It is redundantly included so as to make the workbook more meaningful to read.
Dependency Type	This is the type of the dependency. It might be such things as ‘Start-to-Finish’ or ‘Start-to-Start’ etc. It is documentation and not used in any processing.
Lag in Days	This is the number of days lag between the pre and post dependent tasks.
Project Deliverables	
During Projects a number of Deliverables will be handed over to the client. Some of these deliverables go into production at the end clients site. The client may choose to document deliverables at any level he wishes to. It might not make sense to document deliverables at the level of sql file produced or even at mapping or report level. There will be tasks that deliver reports etc and the client may decide the fact that the report is delivered and also documented in the Reports Documentation Workbook might be enough documentation on the report. The client may decide that there is little need to also document the deliverable of the report in this section of the worksheet. The usage of the worksheet and level of detail for deliverables is to be decided by the client project manager.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **pd** – Project Deliverable Description.
Deliverable Number	This is the string key for the deliverable. It can be linked to tasks.
Deliverable Name	This is the name of the deliverable.
Deliverable Sign Off Flag	This is a flag that can be set to “Y” or “N” to indicate whether this Deliverable is required to be signed off.
Signatories as Text	Some Deliverables will need to be signed off by signatories. This field contains the initials of the signatories to the Deliverable should the task require signatories. If no signatories are required then “NA” is the recommended value.

4.1.42.Business Requirements

The SeETL[DT] Workbook has been upgraded to store documentation of Business Requirements for any particular project though in this case the emphasis is on Business Intelligence Projects. The Business Requirements entered are made available to all users via a suite of reports.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
BR Number	This is the Business Requirement Number that will be used to identify the Business Requirement on reports and in conversations.
BR Name	This is the Business Requirement Name that will be used to identify the Business Requirement on reports and in conversations.
BR Short Description	This is the Business Requirement Short Description that will be used to describe the Business Requirement on reports and in conversations.
BR Long Description	This is the Business Requirement Long Description that will be used to describe the Business Requirement on reports and in conversations.
Owner Name	This is the Business Requirement Owner Name that will be used to document the Business Requirement on reports.
Owner Department	This is the Business Requirement Owner Department that will be used to document the Business Requirement on reports.
Importance	This is the Business Requirement Importance that will be used to document the Business Requirement Importance on reports.
Priority	This is the Business Requirement Priority that will be used to document the Business Requirement Priority on reports.
Delivered In Iteration	This is the Business Requirement Delivered In Iteration information that will be used to document the Business Requirement Delivered In Iteration information on reports.
Status	This is the Business Requirement Status information that will be used to document the Business Requirement Status information on reports.
Link to Documentation	This is the Business Requirement Link to Documentation information that will be used to document the Business Requirement Link to Documentation information on reports.
Last Review Date	This is the Business Requirement Last Review Date information that will be used to document the Business Requirement Last Review Date information on reports.
Notes 1	This is the Business Requirement Notes1 information that will be used to document the Business Requirement Notes1 information on reports.
Notes 2	This is the Business Requirement Notes2 information that will be used to document the Business Requirement Notes2 information on reports.
Notes 3	This is the Business Requirement Notes3 information that will be used to document the Business Requirement Notes3 information on reports.
Notes 4	This is the Business Requirement Notes4 information that will be used to document the Business Requirement Notes4 information on reports.
Notes 5	This is the Business Requirement Notes5 information that will be used to document the Business Requirement Notes5 information on reports.
Notes 6	This is the Business Requirement Notes6 information that will be used to document the Business Requirement Notes6 information on reports.
Notes 7	This is the Business Requirement Notes7 information that will be used to document the Business Requirement Notes7 information on reports.
Notes 8	This is the Business Requirement Notes8 information that will be used to document the Business Requirement Notes8 information on reports.
Notes 9	This is the Business Requirement Notes9 information that will be used to document the Business Requirement Notes9 information on reports.
Notes 10	This is the Business Requirement Notes10 information that will be used to document the Business Requirement Notes10 information on reports.
UserCol 01	This is the Business Requirement UserCol01 information that will be used to document the Business Requirement UserCol01 information on reports.
UserCol 02	This is the Business Requirement UserCol02 information that will be used to document the Business Requirement UserCol02 information on reports.

UserCol 03	This is the Business Requirement UserCol03 information that will be used to document the Business Requirement UserCol03 information on reports.
UserCol 04	This is the Business Requirement UserCol04 information that will be used to document the Business Requirement UserCol04 information on reports.
UserCol 05	This is the Business Requirement UserCol05 information that will be used to document the Business Requirement UserCol05 information on reports.
UserCol 06	This is the Business Requirement UserCol06 information that will be used to document the Business Requirement UserCol06 information on reports.
UserCol 07	This is the Business Requirement UserCol07 information that will be used to document the Business Requirement UserCol07 information on reports.
UserCol 08	This is the Business Requirement UserCol08 information that will be used to document the Business Requirement UserCol08 information on reports.
UserCol 09	This is the Business Requirement UserCol09 information that will be used to document the Business Requirement UserCol09 information on reports.
UserCol 10	This is the Business Requirement UserCol10 information that will be used to document the Business Requirement UserCol10 information on reports.

4.1.43. Project Objectives

The SeETL[DT] Workbook has been upgraded to store documentation of Project Objectives for any particular project though in this case the emphasis is on Business Intelligence Projects. The Project Objectives entered are made available to all users via a suite of reports.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
PO Number	This is the Project Objective Number that will be used to identify the Project Objective on reports and in conversations.
PO Name	This is the Project Objective Name that will be used to identify the Project Objective on reports and in conversations.
PO Short Description	This is the Project Objective Short Description that will be used to describe the Project Objective on reports and in conversations.
PO Long Description	This is the Project Objective Long Description that will be used to describe the Project Objective on reports and in conversations.
Owner Name	This is the Project Objective Owner Name that will be used to document the Project Objective on reports.
Owner Department	This is the Project Objective Owner Department that will be used to document the Project Objective on reports.
Importance	This is the Project Objective Importance that will be used to document the Project Objective Importance on reports.
Priority	This is the Project Objective Priority that will be used to document the Project Objective Priority on reports.
Delivered In Iteration	This is the Project Objective Delivered In Iteration information that will be used to document the Project Objective Delivered In Iteration information on reports.
Status	This is the Project Objective Status information that will be used to document the Project Objective Status information on reports.
Link to Documentation	This is the Project Objective Link to Documentation information that will be used to document the Project Objective Link to Documentation information on reports.
Status	This is the Project Objective Status information that will be used to document the Project Objective Status information on reports.
Priority	This is the Project Objective Priority that will be used to document the Project Objective Priority on reports.
Last Review Date	This is the Project Objective Last Review Date information that will be used to document the Project Objective Last Review Date information on reports.
Notes 1	This is the Project Objective Notes1 information that will be used to document the Project Objective Notes1 information on reports.
Notes 2	This is the Project Objective Notes2 information that will be used to document the Project Objective Notes2 information on reports.
Notes 3	This is the Project Objective Notes3 information that will be used to document the Project Objective Notes3 information on reports.
Notes 4	This is the Project Objective Notes4 information that will be used to document the Project Objective Notes4 information on reports.
Notes 5	This is the Project Objective Notes5 information that will be used to document the Project Objective Notes5 information on reports.
Notes 6	This is the Project Objective Notes6 information that will be used to document the Project Objective Notes6 information on reports.
Notes 7	This is the Project Objective Notes7 information that will be used to document the Project Objective Notes7 information on reports.
Notes 8	This is the Project Objective Notes8 information that will be used to document the Project Objective Notes8 information on reports.
Notes 9	This is the Project Objective Notes9 information that will be used to document the Project Objective Notes9 information on reports.
Notes 10	This is the Project Objective Notes10 information that will be used to document the Project Objective Notes10 information on reports.

UserCol 01	This is the Project Objective UserCol01 information that will be used to document the Project Objective UserCol01 information on reports.
UserCol 02	This is the Project Objective UserCol02 information that will be used to document the Project Objective UserCol02 information on reports.
UserCol 03	This is the Project Objective UserCol03 information that will be used to document the Project Objective UserCol03 information on reports.
UserCol 04	This is the Project Objective UserCol04 information that will be used to document the Project Objective UserCol04 information on reports.
UserCol 05	This is the Project Objective UserCol05 information that will be used to document the Project Objective UserCol05 information on reports.
UserCol 06	This is the Project Objective UserCol06 information that will be used to document the Project Objective UserCol06 information on reports.
UserCol 07	This is the Project Objective UserCol07 information that will be used to document the Project Objective UserCol07 information on reports.
UserCol 08	This is the Project Objective UserCol08 information that will be used to document the Project Objective UserCol08 information on reports.
UserCol 09	This is the Project Objective UserCol09 information that will be used to document the Project Objective UserCol09 information on reports.
UserCol 10	This is the Project Objective UserCol10 information that will be used to document the Project Objective UserCol10 information on reports.

4.1.44.Critical Success Factors

The SeETL[DT] Workbook has been upgraded to store documentation of Critical Success Factors for any particular project though in this case the emphasis is on Business Intelligence Projects. The Critical Success Factors entered are made available to all users via a suite of reports.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
CSF Number	This is the Critical Success Factor Number that will be used to identify the Critical Success Factor on reports and in conversations.
CSF Name	This is the Critical Success Factor Name that will be used to identify the Critical Success Factor on reports and in conversations.
CSF Short Description	This is the Critical Success Factor Short Description that will be used to describe the Critical Success Factor on reports and in conversations.
CSF Long Description	This is the Critical Success Factor Long Description that will be used to describe the Critical Success Factor on reports and in conversations.
Owner Name	This is the Critical Success Factor Owner Name that will be used to document the Critical Success Factor on reports.
Owner Department	This is the Critical Success Factor Owner Department that will be used to document the Critical Success Factor on reports.
Importance	This is the Critical Success Factor Importance that will be used to document the Critical Success Factor Importance on reports.
Priority	This is the Critical Success Factor Priority that will be used to document the Critical Success Factor Priority on reports.
Potential Impact	This is the Critical Success Factor Potential Impact that will be used to document the Critical Success Factor Potential Impact on reports.
Delivered In Iteration	This is the Critical Success Factor Delivered In Iteration information that will be used to document the Critical Success Factor Delivered In Iteration information on reports.
Status	This is the Critical Success Factor Status information that will be used to document the Critical Success Factor Status information on reports.
Link to Documentation	This is the Critical Success Factor Link to Documentation information that will be used to document the Critical Success Factor Link to Documentation information on reports.
Status	This is the Critical Success Factor Status information that will be used to document the Critical Success Factor Status information on reports.
Priority	This is the Critical Success Factor Priority that will be used to document the Critical Success Factor Priority on reports.
Last Review Date	This is the Critical Success Factor Last Review Date information that will be used to document the Critical Success Factor Last Review Date information on reports.
Notes 1	This is the Critical Success Factor Notes1 information that will be used to document the Critical Success Factor Notes1 information on reports.
Notes 2	This is the Critical Success Factor Notes2 information that will be used to document the Critical Success Factor Notes2 information on reports.
Notes 3	This is the Critical Success Factor Notes3 information that will be used to document the Critical Success Factor Notes3 information on reports.
Notes 4	This is the Critical Success Factor Notes4 information that will be used to document the Critical Success Factor Notes4 information on reports.
Notes 5	This is the Critical Success Factor Notes5 information that will be used to document the Critical Success Factor Notes5 information on reports.
Notes 6	This is the Critical Success Factor Notes6 information that will be used to document the Critical Success Factor Notes6 information on reports.
Notes 7	This is the Critical Success Factor Notes7 information that will be used to document the Critical Success Factor Notes7 information on reports.
Notes 8	This is the Critical Success Factor Notes8 information that will be used to document the Critical Success Factor Notes8 information on reports.
Notes 9	This is the Critical Success Factor Notes9 information that will be used to document the Critical Success Factor Notes9 information on reports.

Notes 10	This is the Critical Success Factor Notes10 information that will be used to document the Critical Success Factor Notes10 information on reports.
UserCol 01	This is the Critical Success Factor UserCol01 information that will be used to document the Critical Success Factor UserCol01 information on reports.
UserCol 02	This is the Critical Success Factor UserCol02 information that will be used to document the Critical Success Factor UserCol02 information on reports.
UserCol 03	This is the Critical Success Factor UserCol03 information that will be used to document the Critical Success Factor UserCol03 information on reports.
UserCol 04	This is the Critical Success Factor UserCol04 information that will be used to document the Critical Success Factor UserCol04 information on reports.
UserCol 05	This is the Critical Success Factor UserCol05 information that will be used to document the Critical Success Factor UserCol05 information on reports.
UserCol 06	This is the Critical Success Factor UserCol06 information that will be used to document the Critical Success Factor UserCol06 information on reports.
UserCol 07	This is the Critical Success Factor UserCol07 information that will be used to document the Critical Success Factor UserCol07 information on reports.
UserCol 08	This is the Critical Success Factor UserCol08 information that will be used to document the Critical Success Factor UserCol08 information on reports.
UserCol 09	This is the Critical Success Factor UserCol09 information that will be used to document the Critical Success Factor UserCol09 information on reports.
UserCol 10	This is the Critical Success Factor UserCol10 information that will be used to document the Critical Success Factor UserCol10 information on reports.

4.1.45.Major Dimensions

The SeETL[DT] Workbook has been upgraded to store documentation of Major Dimensions for any Business Intelligence Projects. The purpose of this WorkBook is to serve as a documentation step early in the project rather than to actually start designing the data model. This is a mandatory step for companies that are not buying a data model into the project.

The Major Dimensions entered are made available to all users via a suite of reports.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
Dimension Number	This is the Dimension Number that will be used to identify the Dimension on reports and in conversations.
Dimension Name	This is the Dimension Name that will be used to identify the Dimension on reports and in conversations.
Dimension Short Description	This is the Dimension Short Description that will be used to describe the Dimension on reports and in conversations.
Dimension Long Description	This is the Dimension Long Description that will be used to describe the Dimension on reports and in conversations.
Owner Name	This is the Dimension Owner Name that will be used to document the Dimension on reports.
Owner Department	This is the Dimension Owner Department that will be used to document the Dimension on reports.
Importance	This is the Dimension Importance that will be used to document the Dimension Importance on reports.
Priority	This is the Dimension Priority that will be used to document the Dimension Priority on reports.
Delivered In Iteration	This is the Dimension Delivered In Iteration information that will be used to document the Dimension Delivered In Iteration information on reports.
Status	This is the Dimension Status information that will be used to document the Dimension Status information on reports.
Link to Documentation	This is the Dimension Link to Documentation information that will be used to document the Dimension Link to Documentation information on reports.
Status	This is the Dimension Status information that will be used to document the Dimension Status information on reports.
Priority	This is the Dimension Priority that will be used to document the Dimension Priority on reports.
Last Review Date	This is the Dimension Last Review Date information that will be used to document the Dimension Last Review Date information on reports.
Notes 1	This is the Dimension Notes1 information that will be used to document the Dimension Notes1 information on reports.
Notes 2	This is the Dimension Notes2 information that will be used to document the Dimension Notes2 information on reports.
Notes 3	This is the Dimension Notes3 information that will be used to document the Dimension Notes3 information on reports.
Notes 4	This is the Dimension Notes4 information that will be used to document the Dimension Notes4 information on reports.
Notes 5	This is the Dimension Notes5 information that will be used to document the Dimension Notes5 information on reports.
Notes 6	This is the Dimension Notes6 information that will be used to document the Dimension Notes6 information on reports.
Notes 7	This is the Dimension Notes7 information that will be used to document the Dimension Notes7 information on reports.
Notes 8	This is the Dimension Notes8 information that will be used to document the Dimension Notes8 information on reports.
Notes 9	This is the Dimension Notes9 information that will be used to document the Dimension

	Notes9 information on reports.
Notes 10	This is the Dimension Notes10 information that will be used to document the Dimension Notes10 information on reports.
UserCol 01	This is the Dimension UserCol01 information that will be used to document the Dimension UserCol01 information on reports.
UserCol 02	This is the Dimension UserCol02 information that will be used to document the Dimension UserCol02 information on reports.
UserCol 03	This is the Dimension UserCol03 information that will be used to document the Dimension UserCol03 information on reports.
UserCol 04	This is the Dimension UserCol04 information that will be used to document the Dimension UserCol04 information on reports.
UserCol 05	This is the Dimension UserCol05 information that will be used to document the Dimension UserCol05 information on reports.
UserCol 06	This is the Dimension UserCol06 information that will be used to document the Dimension UserCol06 information on reports.
UserCol 07	This is the Dimension UserCol07 information that will be used to document the Dimension UserCol07 information on reports.
UserCol 08	This is the Dimension UserCol08 information that will be used to document the Dimension UserCol08 information on reports.
UserCol 09	This is the Dimension UserCol09 information that will be used to document the Dimension UserCol09 information on reports.
UserCol 10	This is the Dimension UserCol10 information that will be used to document the Dimension UserCol10 information on reports.

4.1.46.Major Facts

The SeETL[DT] Workbook has been upgraded to store documentation of Major Facts for any Business Intelligence Projects. The purpose of this WorkBook is to serve as a documentation step early in the project rather than to actually start designing the data model. This is a mandatory step for companies that are not buying a data model into the project.

The Major Facts entered are made available to all users via a suite of reports.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
Fact Number	This is the Fact Number that will be used to identify the Fact on reports and in conversations.
Fact Name	This is the Fact Name that will be used to identify the Fact on reports and in conversations.
Fact Short Description	This is the Fact Short Description that will be used to describe the Fact on reports and in conversations.
Fact Long Description	This is the Fact Long Description that will be used to describe the Fact on reports and in conversations.
Owner Name	This is the Fact Owner Name that will be used to document the Fact on reports.
Owner Department	This is the Fact Owner Department that will be used to document the Fact on reports.
Importance	This is the Fact Importance that will be used to document the Fact Importance on reports.
Priority	This is the Fact Priority that will be used to document the Fact Priority on reports.
Delivered In Iteration	This is the Fact Delivered In Iteration information that will be used to document the Fact Delivered In Iteration information on reports.
Status	This is the Fact Status information that will be used to document the Fact Status information on reports.
Link to Documentation	This is the Fact Link to Documentation information that will be used to document the Fact Link to Documentation information on reports.
Status	This is the Fact Status information that will be used to document the Fact Status information on reports.
Priority	This is the Fact Priority that will be used to document the Fact Priority on reports.
Last Review Date	This is the Fact Last Review Date information that will be used to document the Fact Last Review Date information on reports.
Notes 1	This is the Fact Notes1 information that will be used to document the Fact Notes1 information on reports.
Notes 2	This is the Fact Notes2 information that will be used to document the Fact Notes2 information on reports.
Notes 3	This is the Fact Notes3 information that will be used to document the Fact Notes3 information on reports.
Notes 4	This is the Fact Notes4 information that will be used to document the Fact Notes4 information on reports.
Notes 5	This is the Fact Notes5 information that will be used to document the Fact Notes5 information on reports.
Notes 6	This is the Fact Notes6 information that will be used to document the Fact Notes6 information on reports.
Notes 7	This is the Fact Notes7 information that will be used to document the Fact Notes7 information on reports.
Notes 8	This is the Fact Notes8 information that will be used to document the Fact Notes8 information on reports.
Notes 9	This is the Fact Notes9 information that will be used to document the Fact Notes9 information on reports.
Notes 10	This is the Fact Notes10 information that will be used to document the Fact Notes10 information on reports.
UserCol 01	This is the Fact UserCol01 information that will be used to document the Fact

	UserCol01 information on reports.
UserCol 02	This is the Fact UserCol02 information that will be used to document the Fact UserCol02 information on reports.
UserCol 03	This is the Fact UserCol03 information that will be used to document the Fact UserCol03 information on reports.
UserCol 04	This is the Fact UserCol04 information that will be used to document the Fact UserCol04 information on reports.
UserCol 05	This is the Fact UserCol05 information that will be used to document the Fact UserCol05 information on reports.
UserCol 06	This is the Fact UserCol06 information that will be used to document the Fact UserCol06 information on reports.
UserCol 07	This is the Fact UserCol07 information that will be used to document the Fact UserCol07 information on reports.
UserCol 08	This is the Fact UserCol08 information that will be used to document the Fact UserCol08 information on reports.
UserCol 09	This is the Fact UserCol09 information that will be used to document the Fact UserCol09 information on reports.
UserCol 10	This is the Fact UserCol10 information that will be used to document the Fact UserCol10 information on reports.

4.1.47.Defined Measures

The SeETL[DT] Workbook has been upgraded to store documentation of Defined Measures for any Business Intelligence Projects. The purpose of this WorkBook is to serve as a documentation step early in the project rather than to actually start designing the data model. This is a mandatory step for companies that are not buying a data model into the project.

The Defined Measures entered are made available to all users via a suite of reports.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
Measure Number	This is the Measure Number that will be used to identify the Measure on reports and in conversations.
Measure Name	This is the Measure Name that will be used to identify the Measure on reports and in conversations.
Measure Short Description	This is the Measure Short Description that will be used to describe the Measure on reports and in conversations.
Measure Long Description	This is the Measure Long Description that will be used to describe the Measure on reports and in conversations.
Owner Name	This is the Measure Owner Name that will be used to document the Measure on reports.
Owner Department	This is the Measure Owner Department that will be used to document the Measure on reports.
Importance	This is the Measure Importance that will be used to document the Measure Importance on reports.
Priority	This is the Measure Priority that will be used to document the Measure Priority on reports.
Delivered In Iteration	This is the Measure Delivered In Iteration information that will be used to document the Measure Delivered In Iteration information on reports.
Status	This is the Measure Status information that will be used to document the Measure Status information on reports.
Link to Documentation	This is the Measure Link to Documentation information that will be used to document the Measure Link to Documentation information on reports.
Status	This is the Measure Status information that will be used to document the Measure Status information on reports.
Priority	This is the Measure Priority that will be used to document the Measure Priority on reports.
Last Review Date	This is the Measure Last Review Date information that will be used to document the Measure Last Review Date information on reports.
Notes 1	This is the Measure Notes1 information that will be used to document the Measure Notes1 information on reports.
Notes 2	This is the Measure Notes2 information that will be used to document the Measure Notes2 information on reports.
Notes 3	This is the Measure Notes3 information that will be used to document the Measure Notes3 information on reports.
Notes 4	This is the Measure Notes4 information that will be used to document the Measure Notes4 information on reports.
Notes 5	This is the Measure Notes5 information that will be used to document the Measure Notes5 information on reports.
Notes 6	This is the Measure Notes6 information that will be used to document the Measure Notes6 information on reports.
Notes 7	This is the Measure Notes7 information that will be used to document the Measure Notes7 information on reports.
Notes 8	This is the Measure Notes8 information that will be used to document the Measure Notes8 information on reports.
Notes 9	This is the Measure Notes9 information that will be used to document the Measure

	Notes9 information on reports.
Notes 10	This is the Measure Notes10 information that will be used to document the Measure Notes10 information on reports.
UserCol 01	This is the Measure UserCol01 information that will be used to document the Measure UserCol01 information on reports.
UserCol 02	This is the Measure UserCol02 information that will be used to document the Measure UserCol02 information on reports.
UserCol 03	This is the Measure UserCol03 information that will be used to document the Measure UserCol03 information on reports.
UserCol 04	This is the Measure UserCol04 information that will be used to document the Measure UserCol04 information on reports.
UserCol 05	This is the Measure UserCol05 information that will be used to document the Measure UserCol05 information on reports.
UserCol 06	This is the Measure UserCol06 information that will be used to document the Measure UserCol06 information on reports.
UserCol 07	This is the Measure UserCol07 information that will be used to document the Measure UserCol07 information on reports.
UserCol 08	This is the Measure UserCol08 information that will be used to document the Measure UserCol08 information on reports.
UserCol 09	This is the Measure UserCol09 information that will be used to document the Measure UserCol09 information on reports.
UserCol 10	This is the Measure UserCol10 information that will be used to document the Measure UserCol10 information on reports.

4.1.48.Subject Areas

The SeETL[DT] Workbook has been upgraded to store documentation of Subject Areas for any Business Intelligence Projects. The purpose of this WorkBook is to serve as a documentation step early in the project rather than to actually start designing the data model. This is a mandatory step for companies that are not buying a data model into the project.

The Subject Areas entered are made available to all users via a suite of reports.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
SA Number	This is the Subject Area Number that will be used to identify the Subject Area on reports and in conversations.
SA Name	This is the Subject Area Name that will be used to identify the Subject Area on reports and in conversations.
SA Short Description	This is the Subject Area Short Description that will be used to describe the Subject Area on reports and in conversations.
SA Long Description	This is the Subject Area Long Description that will be used to describe the Subject Area on reports and in conversations.
Owner Name	This is the Subject Area Owner Name that will be used to document the Subject Area on reports.
Owner Department	This is the Subject Area Owner Department that will be used to document the Subject Area on reports.
Importance	This is the Subject Area Importance that will be used to document the Subject Area Importance on reports.
Priority	This is the Subject Area Priority that will be used to document the Subject Area Priority on reports.
Delivered In Iteration	This is the Subject Area Delivered In Iteration information that will be used to document the Subject Area Delivered In Iteration information on reports.
Status	This is the Subject Area Status information that will be used to document the Subject Area Status information on reports.
Link to Documentation	This is the Subject Area Link to Documentation information that will be used to document the Subject Area Link to Documentation information on reports.
Status	This is the Subject Area Status information that will be used to document the Subject Area Status information on reports.
Priority	This is the Subject Area Priority that will be used to document the Subject Area Priority on reports.
Last Review Date	This is the Subject Area Last Review Date information that will be used to document the Subject Area Last Review Date information on reports.
Notes 1	This is the Subject Area Notes1 information that will be used to document the Subject Area Notes1 information on reports.
Notes 2	This is the Subject Area Notes2 information that will be used to document the Subject Area Notes2 information on reports.
Notes 3	This is the Subject Area Notes3 information that will be used to document the Subject Area Notes3 information on reports.
Notes 4	This is the Subject Area Notes4 information that will be used to document the Subject Area Notes4 information on reports.
Notes 5	This is the Subject Area Notes5 information that will be used to document the Subject Area Notes5 information on reports.
Notes 6	This is the Subject Area Notes6 information that will be used to document the Subject Area Notes6 information on reports.
Notes 7	This is the Subject Area Notes7 information that will be used to document the Subject Area Notes7 information on reports.
Notes 8	This is the Subject Area Notes8 information that will be used to document the Subject Area Notes8 information on reports.
Notes 9	This is the Subject Area Notes9 information that will be used to document the Subject

	Area Notes9 information on reports.
Notes 10	This is the Subject Area Notes10 information that will be used to document the Subject Area Notes10 information on reports.
UserCol 01	This is the Subject Area UserCol01 information that will be used to document the Subject Area UserCol01 information on reports.
UserCol 02	This is the Subject Area UserCol02 information that will be used to document the Subject Area UserCol02 information on reports.
UserCol 03	This is the Subject Area UserCol03 information that will be used to document the Subject Area UserCol03 information on reports.
UserCol 04	This is the Subject Area UserCol04 information that will be used to document the Subject Area UserCol04 information on reports.
UserCol 05	This is the Subject Area UserCol05 information that will be used to document the Subject Area UserCol05 information on reports.
UserCol 06	This is the Subject Area UserCol06 information that will be used to document the Subject Area UserCol06 information on reports.
UserCol 07	This is the Subject Area UserCol07 information that will be used to document the Subject Area UserCol07 information on reports.
UserCol 08	This is the Subject Area UserCol08 information that will be used to document the Subject Area UserCol08 information on reports.
UserCol 09	This is the Subject Area UserCol09 information that will be used to document the Subject Area UserCol09 information on reports.
UserCol 10	This is the Subject Area UserCol10 information that will be used to document the Subject Area UserCol10 information on reports.

4.1.49.Cross Reference 01

This Sheet contains the SeETL Cross References for Project Objectives and Business Requirements.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
CREF Number	This is the Cross Reference Number that will be used to identify the Cross Reference on reports and in conversations.
PO Number	This is the Project Objective Number that will be used to identify the Project Objective on reports and in conversations.
PO Name	This is the Project Objective Name that will be used to identify the Project Objective on reports and in conversations.
BR Number	This is the Business Requirement Number that will be used to identify the Business Requirement on reports and in conversations.
BR Name	This is the Business Requirement Name that will be used to identify the Business Requirement on reports and in conversations.
Importance	This is the Cross Reference Importance that will be used to document the Cross Reference Importance on reports.
Priority	This is the Cross Reference Priority that will be used to document the Cross Reference Priority on reports.
Status	This is the Cross Reference Status information that will be used to document the Cross Reference Status information on reports.
Link to Documentation	This is the Cross Reference Link to Documentation information that will be used to document the Cross Reference Link to Documentation information on reports.
Last Review Date	This is the Cross Reference Last Review Date information that will be used to document the Cross Reference Last Review Date information on reports.
Notes 1	This is the Cross Reference Notes1 information that will be used to document the Cross Reference Notes1 information on reports.
Notes 2	This is the Cross Reference Notes2 information that will be used to document the Cross Reference Notes2 information on reports.
Notes 3	This is the Cross Reference Notes3 information that will be used to document the Cross Reference Notes3 information on reports.
Notes 4	This is the Cross Reference Notes4 information that will be used to document the Cross Reference Notes4 information on reports.
Notes 5	This is the Cross Reference Notes5 information that will be used to document the Cross Reference Notes5 information on reports.
Notes 6	This is the Cross Reference Notes6 information that will be used to document the Cross Reference Notes6 information on reports.
Notes 7	This is the Cross Reference Notes7 information that will be used to document the Cross Reference Notes7 information on reports.
Notes 8	This is the Cross Reference Notes8 information that will be used to document the Cross Reference Notes8 information on reports.
Notes 9	This is the Cross Reference Notes9 information that will be used to document the Cross Reference Notes9 information on reports.
Notes 10	This is the Cross Reference Notes10 information that will be used to document the Cross Reference Notes10 information on reports.
UserCol 01	This is the Cross Reference UserCol01 information that will be used to document the Cross Reference UserCol01 information on reports.
UserCol 02	This is the Cross Reference UserCol02 information that will be used to document the Cross Reference UserCol02 information on reports.
UserCol 03	This is the Cross Reference UserCol03 information that will be used to document the Cross Reference UserCol03 information on reports.
UserCol 04	This is the Cross Reference UserCol04 information that will be used to document the Cross Reference UserCol04 information on reports.
UserCol 05	This is the Cross Reference UserCol05 information that will be used to document the Cross Reference UserCol05 information on reports.

UserCol 06	This is the Cross Reference UserCol06 information that will be used to document the Cross Reference UserCol06 information on reports.
UserCol 07	This is the Cross Reference UserCol07 information that will be used to document the Cross Reference UserCol07 information on reports.
UserCol 08	This is the Cross Reference UserCol08 information that will be used to document the Cross Reference UserCol08 information on reports.
UserCol 09	This is the Cross Reference UserCol09 information that will be used to document the Cross Reference UserCol09 information on reports.
UserCol 10	This is the Cross Reference UserCol10 information that will be used to document the Cross Reference UserCol10 information on reports.

4.1.50.Cross Reference 02

This Sheet contains the SeETL Cross References for Project Objectives, Business Requirements and Major Dimensions.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
CREF Number	This is the Cross Reference Number that will be used to identify the Cross Reference on reports and in conversations.
PO Number	This is the Project Objective Number that will be used to identify the Project Objective on reports and in conversations.
PO Name	This is the Project Objective Name that will be used to identify the Project Objective on reports and in conversations.
BR Number	This is the Business Requirement Number that will be used to identify the Business Requirement on reports and in conversations.
BR Name	This is the Business Requirement Name that will be used to identify the Business Requirement on reports and in conversations.
Dimension Number	This is the Dimension Number that will be used to identify the Dimension on reports and in conversations.
Dimension Name	This is the Dimension Name that will be used to identify the Dimension on reports and in conversations.
Importance	This is the Cross Reference Importance that will be used to document the Cross Reference Importance on reports.
Priority	This is the Cross Reference Priority that will be used to document the Cross Reference Priority on reports.
Status	This is the Cross Reference Status information that will be used to document the Cross Reference Status information on reports.
Link to Documentation	This is the Cross Reference Link to Documentation information that will be used to document the Cross Reference Link to Documentation information on reports.
Last Review Date	This is the Cross Reference Last Review Date information that will be used to document the Cross Reference Last Review Date information on reports.
Notes 1	This is the Cross Reference Notes1 information that will be used to document the Cross Reference Notes1 information on reports.
Notes 2	This is the Cross Reference Notes2 information that will be used to document the Cross Reference Notes2 information on reports.
Notes 3	This is the Cross Reference Notes3 information that will be used to document the Cross Reference Notes3 information on reports.
Notes 4	This is the Cross Reference Notes4 information that will be used to document the Cross Reference Notes4 information on reports.
Notes 5	This is the Cross Reference Notes5 information that will be used to document the Cross Reference Notes5 information on reports.
Notes 6	This is the Cross Reference Notes6 information that will be used to document the Cross Reference Notes6 information on reports.
Notes 7	This is the Cross Reference Notes7 information that will be used to document the Cross Reference Notes7 information on reports.
Notes 8	This is the Cross Reference Notes8 information that will be used to document the Cross Reference Notes8 information on reports.
Notes 9	This is the Cross Reference Notes9 information that will be used to document the Cross Reference Notes9 information on reports.
Notes 10	This is the Cross Reference Notes10 information that will be used to document the Cross Reference Notes10 information on reports.
UserCol 01	This is the Cross Reference UserCol01 information that will be used to document the Cross Reference UserCol01 information on reports.
UserCol 02	This is the Cross Reference UserCol02 information that will be used to document the Cross Reference UserCol02 information on reports.
UserCol 03	This is the Cross Reference UserCol03 information that will be used to document the

	Cross Reference UserCol03 information on reports.
UserCol 04	This is the Cross Reference UserCol04 information that will be used to document the Cross Reference UserCol04 information on reports.
UserCol 05	This is the Cross Reference UserCol05 information that will be used to document the Cross Reference UserCol05 information on reports.
UserCol 06	This is the Cross Reference UserCol06 information that will be used to document the Cross Reference UserCol06 information on reports.
UserCol 07	This is the Cross Reference UserCol07 information that will be used to document the Cross Reference UserCol07 information on reports.
UserCol 08	This is the Cross Reference UserCol08 information that will be used to document the Cross Reference UserCol08 information on reports.
UserCol 09	This is the Cross Reference UserCol09 information that will be used to document the Cross Reference UserCol09 information on reports.
UserCol 10	This is the Cross Reference UserCol10 information that will be used to document the Cross Reference UserCol10 information on reports.

4.1.51.Cross Reference 03

This Sheet contains the SeETL Cross References for Project Objectives, Business Requirements and Major Facts.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
CREF Number	This is the Cross Reference Number that will be used to identify the Cross Reference on reports and in conversations.
PO Number	This is the Project Objective Number that will be used to identify the Project Objective on reports and in conversations.
PO Name	This is the Project Objective Name that will be used to identify the Project Objective on reports and in conversations.
BR Number	This is the Business Requirement Number that will be used to identify the Business Requirement on reports and in conversations.
BR Name	This is the Business Requirement Name that will be used to identify the Business Requirement on reports and in conversations.
Fact Number	This is the Fact Number that will be used to identify the Fact on reports and in conversations.
Fact Name	This is the Fact Name that will be used to identify the Fact on reports and in conversations.
Importance	This is the Cross Reference Importance that will be used to document the Cross Reference Importance on reports.
Priority	This is the Cross Reference Priority that will be used to document the Cross Reference Priority on reports.
Status	This is the Cross Reference Status information that will be used to document the Cross Reference Status information on reports.
Link to Documentation	This is the Cross Reference Link to Documentation information that will be used to document the Cross Reference Link to Documentation information on reports.
Last Review Date	This is the Cross Reference Last Review Date information that will be used to document the Cross Reference Last Review Date information on reports.
Notes 1	This is the Cross Reference Notes1 information that will be used to document the Cross Reference Notes1 information on reports.
Notes 2	This is the Cross Reference Notes2 information that will be used to document the Cross Reference Notes2 information on reports.
Notes 3	This is the Cross Reference Notes3 information that will be used to document the Cross Reference Notes3 information on reports.
Notes 4	This is the Cross Reference Notes4 information that will be used to document the Cross Reference Notes4 information on reports.
Notes 5	This is the Cross Reference Notes5 information that will be used to document the Cross Reference Notes5 information on reports.
Notes 6	This is the Cross Reference Notes6 information that will be used to document the Cross Reference Notes6 information on reports.
Notes 7	This is the Cross Reference Notes7 information that will be used to document the Cross Reference Notes7 information on reports.
Notes 8	This is the Cross Reference Notes8 information that will be used to document the Cross Reference Notes8 information on reports.
Notes 9	This is the Cross Reference Notes9 information that will be used to document the Cross Reference Notes9 information on reports.
Notes 10	This is the Cross Reference Notes10 information that will be used to document the Cross Reference Notes10 information on reports.
UserCol 01	This is the Cross Reference UserCol01 information that will be used to document the Cross Reference UserCol01 information on reports.
UserCol 02	This is the Cross Reference UserCol02 information that will be used to document the Cross Reference UserCol02 information on reports.
UserCol 03	This is the Cross Reference UserCol03 information that will be used to document the Cross Reference UserCol03 information on reports.

UserCol 04	This is the Cross Reference UserCol04 information that will be used to document the Cross Reference UserCol04 information on reports.
UserCol 05	This is the Cross Reference UserCol05 information that will be used to document the Cross Reference UserCol05 information on reports.
UserCol 06	This is the Cross Reference UserCol06 information that will be used to document the Cross Reference UserCol06 information on reports.
UserCol 07	This is the Cross Reference UserCol07 information that will be used to document the Cross Reference UserCol07 information on reports.
UserCol 08	This is the Cross Reference UserCol08 information that will be used to document the Cross Reference UserCol08 information on reports.
UserCol 09	This is the Cross Reference UserCol09 information that will be used to document the Cross Reference UserCol09 information on reports.
UserCol 10	This is the Cross Reference UserCol10 information that will be used to document the Cross Reference UserCol10 information on reports.

4.1.52.Cross Reference 04

This Sheet contains the SeETL Cross References for Project Objectives, Business Requirements and Measures.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
CREF Number	This is the Cross Reference Number that will be used to identify the Cross Reference on reports and in conversations.
PO Number	This is the Project Objective Number that will be used to identify the Project Objective on reports and in conversations.
PO Name	This is the Project Objective Name that will be used to identify the Project Objective on reports and in conversations.
BR Number	This is the Business Requirement Number that will be used to identify the Business Requirement on reports and in conversations.
BR Name	This is the Business Requirement Name that will be used to identify the Business Requirement on reports and in conversations.
Measure Number	This is the Measure Number that will be used to identify the Measure on reports and in conversations.
Measure Name	This is the Measure Name that will be used to identify the Measure on reports and in conversations.
Importance	This is the Cross Reference Importance that will be used to document the Cross Reference Importance on reports.
Priority	This is the Cross Reference Priority that will be used to document the Cross Reference Priority on reports.
Status	This is the Cross Reference Status information that will be used to document the Cross Reference Status information on reports.
Link to Documentation	This is the Cross Reference Link to Documentation information that will be used to document the Cross Reference Link to Documentation information on reports.
Last Review Date	This is the Cross Reference Last Review Date information that will be used to document the Cross Reference Last Review Date information on reports.
Notes 1	This is the Cross Reference Notes1 information that will be used to document the Cross Reference Notes1 information on reports.
Notes 2	This is the Cross Reference Notes2 information that will be used to document the Cross Reference Notes2 information on reports.
Notes 3	This is the Cross Reference Notes3 information that will be used to document the Cross Reference Notes3 information on reports.
Notes 4	This is the Cross Reference Notes4 information that will be used to document the Cross Reference Notes4 information on reports.
Notes 5	This is the Cross Reference Notes5 information that will be used to document the Cross Reference Notes5 information on reports.
Notes 6	This is the Cross Reference Notes6 information that will be used to document the Cross Reference Notes6 information on reports.
Notes 7	This is the Cross Reference Notes7 information that will be used to document the Cross Reference Notes7 information on reports.
Notes 8	This is the Cross Reference Notes8 information that will be used to document the Cross Reference Notes8 information on reports.
Notes 9	This is the Cross Reference Notes9 information that will be used to document the Cross Reference Notes9 information on reports.
Notes 10	This is the Cross Reference Notes10 information that will be used to document the Cross Reference Notes10 information on reports.
UserCol 01	This is the Cross Reference UserCol01 information that will be used to document the Cross Reference UserCol01 information on reports.
UserCol 02	This is the Cross Reference UserCol02 information that will be used to document the Cross Reference UserCol02 information on reports.
UserCol 03	This is the Cross Reference UserCol03 information that will be used to document the Cross Reference UserCol03 information on reports.

UserCol 04	This is the Cross Reference UserCol04 information that will be used to document the Cross Reference UserCol04 information on reports.
UserCol 05	This is the Cross Reference UserCol05 information that will be used to document the Cross Reference UserCol05 information on reports.
UserCol 06	This is the Cross Reference UserCol06 information that will be used to document the Cross Reference UserCol06 information on reports.
UserCol 07	This is the Cross Reference UserCol07 information that will be used to document the Cross Reference UserCol07 information on reports.
UserCol 08	This is the Cross Reference UserCol08 information that will be used to document the Cross Reference UserCol08 information on reports.
UserCol 09	This is the Cross Reference UserCol09 information that will be used to document the Cross Reference UserCol09 information on reports.
UserCol 10	This is the Cross Reference UserCol10 information that will be used to document the Cross Reference UserCol10 information on reports.

4.1.53.Cross Reference 05

This Sheet contains the SeETL Cross References for Tables to Subject Areas.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
CREF Number	This is the Cross Reference Number that will be used to identify the Cross Reference on reports and in conversations.
SA Number	This is the Subject Area Number that will be used to identify the Subject Area on reports and in conversations.
SA Name	This is the Subject Area Name that will be used to identify the Subject Area on reports and in conversations.
Table Number	This is the Table Number that will be used to identify the Table on reports and in conversations.
Table Name	This is the Table Name that will be used to identify the Table on reports and in conversations.
Importance	This is the Cross Reference Importance that will be used to document the Cross Reference Importance on reports.
Priority	This is the Cross Reference Priority that will be used to document the Cross Reference Priority on reports.
Status	This is the Cross Reference Status information that will be used to document the Cross Reference Status information on reports.
Link to Documentation	This is the Cross Reference Link to Documentation information that will be used to document the Cross Reference Link to Documentation information on reports.
Last Review Date	This is the Cross Reference Last Review Date information that will be used to document the Cross Reference Last Review Date information on reports.
Notes 1	This is the Cross Reference Notes1 information that will be used to document the Cross Reference Notes1 information on reports.
Notes 2	This is the Cross Reference Notes2 information that will be used to document the Cross Reference Notes2 information on reports.
Notes 3	This is the Cross Reference Notes3 information that will be used to document the Cross Reference Notes3 information on reports.
Notes 4	This is the Cross Reference Notes4 information that will be used to document the Cross Reference Notes4 information on reports.
Notes 5	This is the Cross Reference Notes5 information that will be used to document the Cross Reference Notes5 information on reports.
Notes 6	This is the Cross Reference Notes6 information that will be used to document the Cross Reference Notes6 information on reports.
Notes 7	This is the Cross Reference Notes7 information that will be used to document the Cross Reference Notes7 information on reports.
Notes 8	This is the Cross Reference Notes8 information that will be used to document the Cross Reference Notes8 information on reports.
Notes 9	This is the Cross Reference Notes9 information that will be used to document the Cross Reference Notes9 information on reports.
Notes 10	This is the Cross Reference Notes10 information that will be used to document the Cross Reference Notes10 information on reports.
UserCol 01	This is the Cross Reference UserCol01 information that will be used to document the Cross Reference UserCol01 information on reports.
UserCol 02	This is the Cross Reference UserCol02 information that will be used to document the Cross Reference UserCol02 information on reports.
UserCol 03	This is the Cross Reference UserCol03 information that will be used to document the Cross Reference UserCol03 information on reports.
UserCol 04	This is the Cross Reference UserCol04 information that will be used to document the Cross Reference UserCol04 information on reports.
UserCol 05	This is the Cross Reference UserCol05 information that will be used to document the Cross Reference UserCol05 information on reports.

UserCol 06	This is the Cross Reference UserCol06 information that will be used to document the Cross Reference UserCol06 information on reports.
UserCol 07	This is the Cross Reference UserCol07 information that will be used to document the Cross Reference UserCol07 information on reports.
UserCol 08	This is the Cross Reference UserCol08 information that will be used to document the Cross Reference UserCol08 information on reports.
UserCol 09	This is the Cross Reference UserCol09 information that will be used to document the Cross Reference UserCol09 information on reports.
UserCol 10	This is the Cross Reference UserCol10 information that will be used to document the Cross Reference UserCol10 information on reports.

4.1.54.Solicitation Transaction Email

The SeETL[DT] Workbook has been upgraded to store transaction requests for outbound Emails. Outbound emails are a part of any companies marketing needs today.

The purpose of this WorkBook is to serve as a data entry device for email solicitation transactions. Email transactions can be created via other tools and integrated into the data warehouse. This is just a simple tool that allows SeETL to also generate the email solicitation transactions if desired.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
Solicitation Date	This is the date on which the Solicitation is planned to be made.
Pk_vm_party	If the user is using BI4ALL then this number should be the primary key for the vm_party record for the party to be solicited. It is used as documentation only. However it is placed onto the solicitation record for extra documentation.
Prefix	This is the normal prefix used for the party as defined in the BI4ALL data models.
First Name	This is the normal first name used for the party as defined in the BI4ALL data models.
Second Name	This is the normal second name used for the party as defined in the BI4ALL data models.
Third Name	This is the normal third name used for the party as defined in the BI4ALL data models.
Surname	This is the normal surname used for the party as defined in the BI4ALL data models.
Suffix	This is the normal suffix used for the party as defined in the BI4ALL data models.
Concatenated Name	This is the normal concatenated name used for the party as defined in the BI4ALL data models.
Address to Name	This is the normal concatenated name used for the party as defined in the BI4ALL data models.
Formal Salutation	This is the normal formal salutation used for the party as defined in the BI4ALL data models.
Informal Salutation	This is the normal informal salutation used for the party as defined in the BI4ALL data models.
Written Salutation	This is the normal written salutation used for the party as defined in the BI4ALL data models.
Email Address Receiver	This is the email address that the solicitation will be sent to.
Email Receiver Display Name	This is the display name that will appear in the "to" box of the email client.
Web Address	This is the web address that is associated with the receiver. Usually it will be the receivers LinkedIn address. However it may be a personal email address or a company email address. It is whatever the party has given you to be considered their email web address.
Company	This is the normal company information used for the party as defined in the BI4ALL data models.
Department	This is the normal department information used for the party as defined in the BI4ALL data models.
Job Title	This is the normal job title information used for the party as defined in the BI4ALL data models.
Solicitation Short Description	This is a short description of the solicitation that will be placed onto the BI4ALL data models and can be retrieved by queries or reports.
Solicitation Long Description	This is a long description of the solicitation that will be placed onto the BI4ALL data models and can be retrieved by queries or reports.
Email File	This is a file that is visible to SeETL. It is recommended that it is enclosed in double quotes. The file can be in html or plain text. It is the template for the email that will be sent out to the receiver.
Email File Format	The file may be in html or text format. The valid values of this cell are "html" or "text". If an incorrect value is provided the email file will be assumed to be html.
Email Subject	This will be used as the subject line on each email that is sent out.
Email Address CC	This is an email address that can be used as the CC Email Address.
Email Address BCC	This is an email address that can be used as the BCC Email Address.

Email Address Sender	This is an email address that will appear as the senders email address.
Email Sender Display Name	This is the display name that will appear in the “from” box of the email client.
Campaign Group Name Short	This is the short name of the Campaign Group that this solicitation is for if the user is using BI4ALL Campaign Groups.
Campaign Name Short	This is the short name of the Campaign that this solicitation is for if the user is using BI4ALL Campaigns.
Sub Campaign Name Short	This is the short name of the Sub Campaign that this solicitation is for if the user is using BI4ALL Sub Campaigns.
Smtp Host Name	This is the name of the smtp host that SeETL will attempt to connect to in order to send the email.
Smtp Port	This is the smtp port that SeETL will attempt to connect to in order to send the email.
Smtp Log In Email	This is the smtp log in email that SeETL will attempt to use in order to connect to the mail server to send the email.
Smtp Password	This is the smtp password that SeETL will attempt to use in order to connect to the mail server to send the email.
Email Variable Name 01	This is a literal value that is the name of a variable that can be placed into the text of the email which is able to be replaced by the actual variable at run time. For example “####EmailSalutation####”
Email Variable Value 01	This is a actual value the variable that can be placed into the text of the email which exactly replaces the variable it is substituting for. For example “Dear John Smith”
Email Variable Name 02	This is a literal value that is the name of a variable that can be placed into the text of the email which is able to be replaced by the actual variable at run time.
Email Variable Value 02	This is a actual value the variable that can be placed into the text of the email which exactly replaces the variable it is substituting for.
Email Variable Name 03	This is a literal value that is the name of a variable that can be placed into the text of the email which is able to be replaced by the actual variable at run time.
Email Variable Value 03	This is a actual value the variable that can be placed into the text of the email which exactly replaces the variable it is substituting for.
Email Variable Name 04	This is a literal value that is the name of a variable that can be placed into the text of the email which is able to be replaced by the actual variable at run time.
Email Variable Value 04	This is a actual value the variable that can be placed into the text of the email which exactly replaces the variable it is substituting for.
Email Variable Name 05	This is a literal value that is the name of a variable that can be placed into the text of the email which is able to be replaced by the actual variable at run time.
Email Variable Value 05	This is a actual value the variable that can be placed into the text of the email which exactly replaces the variable it is substituting for.
Email Variable Name 06	This is a literal value that is the name of a variable that can be placed into the text of the email which is able to be replaced by the actual variable at run time.
Email Variable Value 06	This is a actual value the variable that can be placed into the text of the email which exactly replaces the variable it is substituting for.
Email Variable Name 07	This is a literal value that is the name of a variable that can be placed into the text of the email which is able to be replaced by the actual variable at run time.
Email Variable Value 07	This is a actual value the variable that can be placed into the text of the email which exactly replaces the variable it is substituting for.
Email Variable Name 08	This is a literal value that is the name of a variable that can be placed into the text of the email which is able to be replaced by the actual variable at run time.
Email Variable Value 08	This is a actual value the variable that can be placed into the text of the email which exactly replaces the variable it is substituting for.
Email Variable Name 09	This is a literal value that is the name of a variable that can be placed into the text of the email which is able to be replaced by the actual variable at run time.
Email Variable Value 09	This is a actual value the variable that can be placed into the text of the email which exactly replaces the variable it is substituting for.
Email Variable Name 10	This is a literal value that is the name of a variable that can be placed into the text of the email which is able to be replaced by the actual variable at run time.
Email Variable Value 10	This is a actual value the variable that can be placed into the text of the email which exactly replaces the variable it is substituting for.
User Data 01	User data column that is passed to the data warehouse.
User Data 02	User data column that is passed to the data warehouse.
User Data 03	User data column that is passed to the data warehouse.

User Data 04	User data column that is passed to the data warehouse.
User Data 05	User data column that is passed to the data warehouse.
User Data 06	User data column that is passed to the data warehouse.
User Data 07	User data column that is passed to the data warehouse.
User Data 08	User data column that is passed to the data warehouse.
User Data 09	User data column that is passed to the data warehouse.
User Data 10	User data column that is passed to the data warehouse.
Row Deleted from Source Indicator	Set to 0
Row Valid Indicator	Set to 1
Row Sent to EDW Indicator	Set to 0
Row Sent to ODS Indicator	Set to 0

4.1.55.Response Transactions Email

The SeETL[DT] Workbook has been upgraded to store response transactions from outbound Emails. Email marketing is a part of any companies marketing needs today.

The purpose of this WorkBook is to serve as a data entry device for email response transactions. Email response transactions can be created via other tools and integrated into the data warehouse. This is just a simple tool that allows SeETL to also capture the responses from email solicitation transctions if desired.

The Email response transactions entered are processed into the BI4ALL data models and made available to all users via a suite of reports.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
Solicitation Date	This is the date on which the Solicitation is planned to be made.
Response Date	This is the date on which the response was received from the solicitation.
Pk_vm_party	If the user is using BI4ALL then this number should be the primary key for the vm_party record for the party to be solicited. It is used as documentation only. However it is placed onto the solicitation record for extra documentation.
Prefix	This is the normal prefix used for the party as defined in the BI4ALL data models.
First Name	This is the normal first name used for the party as defined in the BI4ALL data models.
Second Name	This is the normal second name used for the party as defined in the BI4ALL data models.
Third Name	This is the normal third name used for the party as defined in the BI4ALL data models.
Surname	This is the normal surname used for the party as defined in the BI4ALL data models.
Suffix	This is the normal suffix used for the party as defined in the BI4ALL data models.
Concatenated Name	This is the normal concatenated name used for the party as defined in the BI4ALL data models.
Address to Name	This is the normal concatenated name used for the party as defined in the BI4ALL data models.
Formal Salutation	This is the normal formal salutation used for the party as defined in the BI4ALL data models.
Informal Salutation	This is the normal informal salutation used for the party as defined in the BI4ALL data models.
Written Salutation	This is the normal written salutation used for the party as defined in the BI4ALL data models.
Email Address Receiver	This is the email address that the solicitation will be sent to.
Web Address	This is the web address that is associated with the receiver. Usually it will be the receivers LinkedIn address. However it may be a personal email address or a company email address. It is whatever the party has given you to be considered their email web address.
Company	This is the normal company information used for the party as defined in the BI4ALL data models.
Department	This is the normal department information used for the party as defined in the BI4ALL data models.
Job Title	This is the normal job title information used for the party as defined in the BI4ALL data models.
Response Short Description	This is a short description of the response that will be placed onto the BI4ALL data models and can be retrieved by queries or reports.
Response Long Description	This is a long description of the response that will be placed onto the BI4ALL data models and can be retrieved by queries or reports.
Campaign Group Name Short	This is the short name of the Campaign Group that this solicitation is for if the user is using BI4ALL Campaign Groups.
Campaign Name Short	This is the short name of the Campaign that this solicitation is for if the user is using BI4ALL Campaigns.
Sub Campaign Name Short	This is the short name of the Sub Campaign that this solicitation is for if the user is using BI4ALL Sub Campaigns.

User Data 01	User data column that is passed to the data warehouse.
User Data 02	User data column that is passed to the data warehouse.
User Data 03	User data column that is passed to the data warehouse.
User Data 04	User data column that is passed to the data warehouse.
User Data 05	User data column that is passed to the data warehouse.
User Data 06	User data column that is passed to the data warehouse.
User Data 07	User data column that is passed to the data warehouse.
User Data 08	User data column that is passed to the data warehouse.
User Data 09	User data column that is passed to the data warehouse.
User Data 10	User data column that is passed to the data warehouse.
Row Deleted from Source Indicator	Set to 0
Row Valid Indicator	Set to 1
Row Sent to EDW Indicator	Set to 0
Row Sent to ODS Indicator	Set to 0

4.1.56.Email Address Bounce

The SeETL[DT] Workbook has been upgraded to store email bounce indicators from outbound Emails.

The Email address bounce transactions entered are processed into the BI4ALL data models and made available to all users via a suite of reports.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
Email Bounce Date	This is the date on which the email bounce happened.
Pk_vm_party	If the user is using BI4ALL then this number should be the primary key for the vm_party record for the party to be solicited. It is used as documentation only. However it is placed onto the solicitation record for extra documentation.
Prefix	This is the normal prefix used for the party as defined in the BI4ALL data models.
First Name	This is the normal first name used for the party as defined in the BI4ALL data models.
Second Name	This is the normal second name used for the party as defined in the BI4ALL data models.
Third Name	This is the normal third name used for the party as defined in the BI4ALL data models.
Surname	This is the normal surname used for the party as defined in the BI4ALL data models.
Suffix	This is the normal suffix used for the party as defined in the BI4ALL data models.
Concatenated Name	This is the normal concatenated name used for the party as defined in the BI4ALL data models.
Address to Name	This is the normal concatenated name used for the party as defined in the BI4ALL data models.
Formal Salutation	This is the normal formal salutation used for the party as defined in the BI4ALL data models.
Informal Salutation	This is the normal informal salutation used for the party as defined in the BI4ALL data models.
Written Salutation	This is the normal written salutation used for the party as defined in the BI4ALL data models.
Email Address Receiver	This is the email address that the solicitation will be sent to.
Web Address	This is the web address that is associated with the receiver. Usually it will be the receivers LinkedIn address. However it may be a personal email address or a company email address. It is whatever the party has given you to be considered their email web address.
User Data 01	User data column that is passed to the data warehouse.
User Data 02	User data column that is passed to the data warehouse.
User Data 03	User data column that is passed to the data warehouse.
User Data 04	User data column that is passed to the data warehouse.
User Data 05	User data column that is passed to the data warehouse.
User Data 06	User data column that is passed to the data warehouse.
User Data 07	User data column that is passed to the data warehouse.
User Data 08	User data column that is passed to the data warehouse.
User Data 09	User data column that is passed to the data warehouse.
User Data 10	User data column that is passed to the data warehouse.
Row Deleted from Source Indicator	Set to 0
Row Valid Indicator	Set to 1
Row Sent to EDW Indicator	Set to 0
Row Sent to ODS Indicator	Set to 0

4.1.57.Do Not Email Request

The SeETL[DT] Workbook has been upgraded to store Do Not Email requests that have been received via whatever mechanism this information is captured.

The Do Not Email requests transactions entered are processed into the BI4ALL data models and made available to all users via a suite of reports.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
Do Not Email Request Date	This is the date on which the Do Not Email request was received.
Pk_vm_party	If the user is using BI4ALL then this number should be the primary key for the vm_party record for the party to be solicited. It is used as documentation only. However it is placed onto the solicitation record for extra documentation.
Prefix	This is the normal prefix used for the party as defined in the BI4ALL data models.
First Name	This is the normal first name used for the party as defined in the BI4ALL data models.
Second Name	This is the normal second name used for the party as defined in the BI4ALL data models.
Third Name	This is the normal third name used for the party as defined in the BI4ALL data models.
Surname	This is the normal surname used for the party as defined in the BI4ALL data models.
Suffix	This is the normal suffix used for the party as defined in the BI4ALL data models.
Concatenated Name	This is the normal concatenated name used for the party as defined in the BI4ALL data models.
Address to Name	This is the normal concatenated name used for the party as defined in the BI4ALL data models.
Formal Salutation	This is the normal formal salutation used for the party as defined in the BI4ALL data models.
Informal Salutation	This is the normal informal salutation used for the party as defined in the BI4ALL data models.
Written Salutation	This is the normal written salutation used for the party as defined in the BI4ALL data models.
Email Address Receiver	This is the email address that the solicitation will be sent to.
Web Address	This is the web address that is associated with the receiver. Usually it will be the receivers LinkedIn address. However it may be a personal email address or a company email address. It is whatever the party has given you to be considered their email web address.
User Data 01	User data column that is passed to the data warehouse.
User Data 02	User data column that is passed to the data warehouse.
User Data 03	User data column that is passed to the data warehouse.
User Data 04	User data column that is passed to the data warehouse.
User Data 05	User data column that is passed to the data warehouse.
User Data 06	User data column that is passed to the data warehouse.
User Data 07	User data column that is passed to the data warehouse.
User Data 08	User data column that is passed to the data warehouse.
User Data 09	User data column that is passed to the data warehouse.
User Data 10	User data column that is passed to the data warehouse.
Row Deleted from Source Indicator	Set to 0
Row Valid Indicator	Set to 1
Row Sent to EDW Indicator	Set to 0
Row Sent to ODS Indicator	Set to 0

4.1.58.Campaign Groups

The SeETL[DT] Workbook has been upgraded to store Campaign Groups as part of the campaign definition data model. This covers all sorts of Campaigns which may be created and run to optimise company performance. This is not limited to marketing campaigns. Campaign management is a part of any companies needs today.

The purpose of this WorkBook is to serve as a data entry device for Campaign Groups. Campaign Groups can be created via other tools and integrated into the data warehouse. This is just a simple tool that allows SeETL to also capture Campaign Groups if desired.

The Campaign Groups entered are processed into the BI4ALL data models and made available to all users via a suite of reports.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
Table Number	This should be set to 112 in all cases.
Source System Number	If the user feeds data into the campaign sub model from different source systems and wishes to differentiate those source systems then this number should be used to provide a unique identifier for those source systems.
Campaign Group Name Short	This is a short name given to the campaign Group. It is recommended that some form of number is used at the front of the short name to ensure easy uniqueness.
Campaign Group Name Long	This is a long name given to the campaign Group.
Short Description	This is a field for the user to document short descriptions for Campaign Groups.
Long Description	This is a field for the user to document long descriptions for Campaign Groups.
Purpose Short Description	This is a field for the user to document short descriptions for the purpose of the Campaign Group.
Purpose Long Description	This is a field for the user to document long descriptions for the purpose of the Campaign Group.
Customer Data 01	Customer data column that is passed to the data warehouse.
Customer Data 02	Customer data column that is passed to the data warehouse.
Customer Data 03	Customer data column that is passed to the data warehouse.
Customer Data 04	Customer data column that is passed to the data warehouse.
Customer Data 05	Customer data column that is passed to the data warehouse.
Customer Data 06	Customer data column that is passed to the data warehouse.
Customer Data 07	Customer data column that is passed to the data warehouse.
Customer Data 08	Customer data column that is passed to the data warehouse.
Customer Data 09	Customer data column that is passed to the data warehouse.
Customer Data 10	Customer data column that is passed to the data warehouse.
Customer Data 11	Customer data column that is passed to the data warehouse.
Customer Data 12	Customer data column that is passed to the data warehouse.
Customer Data 13	Customer data column that is passed to the data warehouse.
Customer Data 14	Customer data column that is passed to the data warehouse.
Customer Data 15	Customer data column that is passed to the data warehouse.
Customer Data 16	Customer data column that is passed to the data warehouse.
Customer Data 17	Customer data column that is passed to the data warehouse.
Customer Data 18	Customer data column that is passed to the data warehouse.
Customer Data 19	Customer data column that is passed to the data warehouse.
Customer Data 20	Customer data column that is passed to the data warehouse.
Row Deleted from Source Indicator	Set to 0
Row Valid Indicator	Set to 1
Row Sent to EDW Indicator	Set to 0
Row Sent to ODS	Set to 0

Indicator	

4.1.59.Campaigns

The SeETL[DT] Workbook has been upgraded to store Campaigns as part of the campaign definition data model. This covers all sorts of Campaigns which may be created and run to optimise company performance. This is not limited to marketing campaigns. Campaign management is a part of any companies needs today.

The purpose of this WorkBook is to serve as a data entry device for Campaigns. Campaigns can be created via other tools and integrated into the data warehouse. This is just a simple tool that allows SeETL to also capture Campaigns if desired.

The Campaigns entered are processed into the BI4ALL data models and made available to all users via a suite of reports.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
Table Number	This should be set to 111 in all cases.
Source System Number	If the user feeds data into the campaign sub model from different source systems and wishes to differentiate those source systems then this number should be used to provide a unique identifier for those source systems.
Campaign Group Name Short	This is a short name given to the campaign Group. It is recommended that some form of number is used at the front of the short name to ensure easy uniqueness.
Short Description	This is a field for the user to document short descriptions for the Campaigns.
Long Description	This is a field for the user to document long descriptions for the Campaigns.
Campaign Name Short	This is a short name given to the Campaign. It is recommended that some form of number is used at the front of the short name to ensure easy uniqueness.
Campaign Name Long	This is a long name given to the Campaign.
Campaign Priority	This is a field for the user to document the priority of the Campaign. Eg. 01-High, 02-Medium, 03-Low.
Campaign Type	This is a field for the user to document the type of the Campaign. Eg. 01-Lift Sales, 02-Cut Costs, 03-Increase Close Rates etc.
Plan Total Solicits.	How many solicitations are planned to be made in this Campaign across all sub campaigns.
Actual Total Solicits.	The actual number of solicitations that were believed to be made in this Campaign across all sub campaigns. It may be incrementally updated over time as results come in.
Plan Total Responses	How many responses are planned to be made in this Campaign across all sub campaigns.
Actual Total Responses	The actual number of responses that were believed to be made to this Campaign across all sub campaigns. It may be incrementally updated over time as results come in.
Plan Start Date	The date on which the campaign is planned to be started.
Plan End Date	The date on which the campaign is planned to be ended.
Actual Start Date	The date on which the campaign was actually started.
Actual End Date	The date on which the campaign was actually ended.
Plan Total Revenue	The planned amount of revenue that should be achieved from this campaign across all sub campaigns.
Actual Total Revenue	The actual amount of revenue that is believed to have been achieved from this campaign across all sub campaigns.
Plan Total Expense	The planned amount of expenses that should be invested into this campaign across all sub campaigns.
Actual Total Expense	The actual amount of expenses that is believed to have been expended into this campaign across all sub campaigns.
Target Group	This is a field for the user to document the Target Group for the Campaign.
Plan Total Accepts	The planned number of accepts that should be achieved from this campaign across all sub campaigns.

Actual Total Accepts	The actual number of accepts that is believed to have been achieved from this campaign across all sub campaigns.
Plan Total Declines	The planned number of declines that should be achieved from this campaign across all sub campaigns.
Actual Total Declines	The actual number of declines that is believed to have been achieved from this campaign across all sub campaigns.
Customer Data 01	Customer data column that is passed to the data warehouse.
Customer Data 02	Customer data column that is passed to the data warehouse.
Customer Data 03	Customer data column that is passed to the data warehouse.
Customer Data 04	Customer data column that is passed to the data warehouse.
Customer Data 05	Customer data column that is passed to the data warehouse.
Customer Data 06	Customer data column that is passed to the data warehouse.
Customer Data 07	Customer data column that is passed to the data warehouse.
Customer Data 08	Customer data column that is passed to the data warehouse.
Customer Data 09	Customer data column that is passed to the data warehouse.
Customer Data 10	Customer data column that is passed to the data warehouse.
Customer Data 11	Customer data column that is passed to the data warehouse.
Customer Data 12	Customer data column that is passed to the data warehouse.
Customer Data 13	Customer data column that is passed to the data warehouse.
Customer Data 14	Customer data column that is passed to the data warehouse.
Customer Data 15	Customer data column that is passed to the data warehouse.
Customer Data 16	Customer data column that is passed to the data warehouse.
Customer Data 17	Customer data column that is passed to the data warehouse.
Customer Data 18	Customer data column that is passed to the data warehouse.
Customer Data 19	Customer data column that is passed to the data warehouse.
Customer Data 20	Customer data column that is passed to the data warehouse.
Row Deleted from Source Indicator	Set to 0
Row Valid Indicator	Set to 1
Row Sent to EDW Indicator	Set to 0
Row Sent to ODS Indicator	Set to 0

4.1.60.Sub Campaigns

The SeETL[DT] Workbook has been upgraded to store Sub Campaigns as part of the campaign definition data model. This covers all sorts of Campaigns which may be created and run to optimise company performance. This is not limited to marketing campaigns. Campaign management is a part of any companies needs today.

The purpose of this WorkBook is to serve as a data entry device for Sub Campaigns. Sub Campaigns can be created via other tools and integrated into the data warehouse. This is just a simple tool that allows SeETL to also capture Sub Campaigns if desired.

The Sub Campaigns entered are processed into the BI4ALL data models and made available to all users via a suite of reports.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
Table Number	This should be set to 154 in all cases.
Source System Number	If the user feeds data into the campaign sub model from different source systems and wishes to differentiate those source systems then this number should be used to provide a unique identifier for those source systems.
Campaign Group Name Short	This is a short name given to the campaign Group. It is recommended that some form of number is used at the front of the short name to ensure easy uniqueness.
Campaign Name Short	This is a short name given to the Campaign. It is recommended that some form of number is used at the front of the short name to ensure easy uniqueness.
Short Description	This is a field for the user to document short descriptions for the Sub Campaigns.
Long Description	This is a field for the user to document long descriptions for the Sub Campaigns.
Sub Campaign Name Short	This is a short name given to the Sub Campaign. It is recommended that some form of number is used at the front of the short name to ensure easy uniqueness.
Sub Campaign Name Long	This is a long name given to the Sub Campaign.
Sub Campaign Media	This is the name of the Media that is being used for the Sub Campaign. Eg. 01-Email, 02-Google, 03-Facebook, 04-Phone etc.
Plan Total Solicits.	How many solicitations are planned to be made in this Sub Campaign.
Actual Total Solicits.	The actual number of solicitations that were believed to be made in this Sub Campaign. It may be incrementally updated over time as results come in.
Plan Total Responses	How many responses are planned to be made in this Sub Campaign.
Actual Total Responses	The actual number of responses that were believed to be made to this Sub Campaign. It may be incrementally updated over time as results come in.
Plan Start Date	The date on which the campaign is planned to be started.
Plan End Date	The date on which the campaign is planned to be ended.
Actual Start Date	The date on which the campaign was actually started.
Actual End Date	The date on which the campaign was actually ended.
Plan Total Revenue	The planned amount of revenue that should be achieved from this campaign across all sub campaigns.
Actual Total Revenue	The actual amount of revenue that is believed to have been achieved from this campaign across all sub campaigns.
Plan Total Expense	The planned amount of expenses that should be invested into this campaign across all sub campaigns.
Actual Total Expense	The actual amount of expenses that is believed to have been expended into this campaign across all sub campaigns.
Target Group	This is a field for the user to document the Target Group for the Campaign.
Plan Total Accepts	The planned number of accepts that should be achieved from this campaign across all sub campaigns.
Actual Total Accepts	The actual number of accepts that is believed to have been achieved from this campaign across all sub campaigns.
Plan Total Declines	The planned number of declines that should be achieved from this campaign across all sub campaigns.

Actual Total Declines	The actual number of declines that is believed to have been achieved from this campaign across all sub campaigns.
Target Group Name	This is an extra field for the user to document the Target Group for the Campaign.
Campaign Target Group Name	This is Target Group Name from the parent Campaign.
Campaign Solicitation Status	This is a field for the user to document the solicitation status for the Sub Campaign.
Campaign	The Campaign Long Name is redundantly stored for reports in this field.
Campaign Priority	This is a field for the user to document the priority of the Campaign. This is redundant for reporting purposes only. Eg. 01-High, 02-Medium, 03-Low.
Campaign Type	This is a field for the user to document the type of the Campaign. This is redundant for reporting purposes only. Eg. 01-Email, 02-Google Ads, 03-Facebook Ads, 04-Outbound Calls etc.
ISO Currency Code	This is a field for the user to document the currency of the Campaign. This is redundant for reporting purposes only.
Contact Method Short Description	This is a field for the user to document short descriptions for the Contact Method in case it has not been documented elsewhere properly on the record.
Contact Method Long Description	This is a field for the user to document long descriptions for the Contact Method in case it has not been documented elsewhere properly on the record.
Customer Data 01	Customer data column that is passed to the data warehouse.
Customer Data 02	Customer data column that is passed to the data warehouse.
Customer Data 03	Customer data column that is passed to the data warehouse.
Customer Data 04	Customer data column that is passed to the data warehouse.
Customer Data 05	Customer data column that is passed to the data warehouse.
Customer Data 06	Customer data column that is passed to the data warehouse.
Customer Data 07	Customer data column that is passed to the data warehouse.
Customer Data 08	Customer data column that is passed to the data warehouse.
Customer Data 09	Customer data column that is passed to the data warehouse.
Customer Data 10	Customer data column that is passed to the data warehouse.
Customer Data 11	Customer data column that is passed to the data warehouse.
Customer Data 12	Customer data column that is passed to the data warehouse.
Customer Data 13	Customer data column that is passed to the data warehouse.
Customer Data 14	Customer data column that is passed to the data warehouse.
Customer Data 15	Customer data column that is passed to the data warehouse.
Customer Data 16	Customer data column that is passed to the data warehouse.
Customer Data 17	Customer data column that is passed to the data warehouse.
Customer Data 18	Customer data column that is passed to the data warehouse.
Customer Data 19	Customer data column that is passed to the data warehouse.
Customer Data 20	Customer data column that is passed to the data warehouse.
Row Deleted from Source Indicator	Set to 0
Row Valid Indicator	Set to 1
Row Sent to EDW Indicator	Set to 0
Row Sent to ODS Indicator	Set to 0

4.1.61.BI4ALL Codes Tables

The SeETL[DT] Workbook has been upgraded to store input for the BI4ALL Codes Tables. This is to provide a simple interface with which to load and maintain the BI4ALL Codes Tables.

The BI4ALL Codes Tables entered are processed into the BI4ALL data models and made available to all users via a suite of reports.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
Table Number	This should be set to 0 in all cases.
Source System Number	If the user feeds code data into the BI4ALL model from different source systems and wishes to differentiate those source systems then this number should be used to provide a unique identifier for those source systems.
Dk_code_name	This is a code name that is defined in the BI4ALL Data Models. Refer to the BI4ALL Data Models for information for valid values.
Dk_code_value	This is the actual code that is being defined into the BI4ALL Data Models. Refer to the BI4ALL Data Models for information for example code values for example code tables.
Code Short Description	This is a field for the user to document short descriptions for the code being defined.
Code Long Description	This is a field for the user to document long descriptions for the code being defined.
Code Help Description	This is a field for the user to document help descriptions for the code being defined.
Row Deleted from Source Indicator	Set to 0
Row Valid Indicator	Set to 1
Row Sent to EDW Indicator	Set to 0
Row Sent to ODS Indicator	Set to 0

4.1.62.LinkedIn Connections

The SeETL[DT] Workbook has been upgraded to store input from LinkedIn into the vm_party table of BI4ALL. This is to provide a simple interface with which to load and maintain the LinkedIn contacts for a company and the sales reps and other appropriate staff for a company.

There are a lot of people on LinkedIn now and maintaining those links into the data warehouse is a useful thing to do.

The LinkedIn connections that are downloaded from LinkedIn are processed into the BI4ALL data models and made available to all users via a suite of reports.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
Connection Export Date	This is the date on which the LinkedIn Connection data was exported from LinkedIn.
First Name	This is the normal first name used for the party as defined in the BI4ALL data models.
Second Name	This is the normal second name used for the party as defined in the BI4ALL data models.
Surname	This is the normal surname used for the party as defined in the BI4ALL data models.
Suffix	This is the normal suffix used for the party as defined in the BI4ALL data models.
Email Address	This is the email address that the solicitation will be sent to.
Company	This is the normal company information used for the party as defined in the BI4ALL data models.
Job Title	This is the normal job title information used for the party as defined in the BI4ALL data models.
LinkedIn Web Address	This is the LinkedIn web address that is associated with the party.
Row Deleted from Source Indicator	Set to 0
Row Valid Indicator	Set to 1
Row Sent to EDW Indicator	Set to 0
Row Sent to ODS Indicator	Set to 0

4.1.63. Email Connections

The SeETL[DT] Workbook has been upgraded to store input from Email Connections into the vm_party table of BI4ALL. This is to provide a simple interface with which to load and maintain the many Email Connections that people manage to gather for a company. Currently these email connections are merely loaded into the vm_party table. There is no attempt to do any further processing at the current point in time.

There are a lot of people on email now and maintaining those links into the data warehouse is a useful thing to do.

The email connections that are downloaded from various sources such as outlook or personal email packages are processed into the BI4ALL data models and made available to all users via a suite of reports.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is a row number that is placed onto the row to make sure that the row is unique.
Connection Import Date	This is the date on which the Email Connections data in intended to be imported to the staging area.
First Name	This is the normal first name used for the party as defined in the BI4ALL data models.
Second Name	This is the normal second name used for the party as defined in the BI4ALL data models.
Surname	This is the normal surname used for the party as defined in the BI4ALL data models.
Suffix	This is the normal suffix used for the party as defined in the BI4ALL data models.
Email Address	This is the email address that the solicitation will be sent to.
Company	This is the normal company information used for the party as defined in the BI4ALL data models.
Job Title	This is the normal job title information used for the party as defined in the BI4ALL data models.
Web Address	This is the web address that is associated with the party.
Row Deleted from Source Indicator	Set to 0
Row Valid Indicator	Set to 1
Row Sent to EDW Indicator	Set to 0
Row Sent to ODS Indicator	Set to 0

4.1.64.Restart Batch

The SeETL[DT] Workbook has been upgraded to store input from the ETL Administrator as to which batch to restart. This is a simple interface so that manual restarts of batches are no longer needed. This is a rare event and the batch restart is rarely needed. Today it is the simply issuance of the appropriate SQL statement by the authorised administrator.

This function merely means that there is a log in the workbook that the restart took place. There is no log in the database that the restart statement was issued.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Batch Number	This is the number of the Batch that is to be restarted.
Batch Name	This is the name of the Batch that is to be restarted.
Started Timestamp	This is the Started Timestamp of the Batch that is to be restarted.

4.1.65. Variable Substitution

The SeETL[DT] Workbook has been upgraded to be able to perform variable substitution of any unique string in any cell in the workbook into another string. This capability is intended to allow the user to alias database names, table owner names, directory paths and similar things.

For example, in the from clause for mappings it is common to place <database>.<owner>.<table> so as to fully qualify the tables to be used. However, when moving up and down the development hierarchy these values might need to be changed. Therefore, it is now possible to change these value by merely altering the workbook in which the variable substitution is recorded.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Variable Name	This is the name of the variable that will be searched for inside each cell in order to determine if it exists. It is up to the user to ensure that he uses variable names that will be unique in the workbook. It is recommend that the variable is preceded and post ceded with 4 hashes. ####. This makes it very unlikely for the string to be found inside the workbook.
Variable Value	This is the value of the variable to be replaced in any cell in which the variable name is found.

4.1.66.Delta Definitions

The SeETL[DT] Workbook has been upgraded to be able to perform the generation of delta detection between two tables, the current table and the minus 1 version of the table. The delta processing can also optionally forward the deltas to the staging area should the designer wish. All delta detection is performed in SQL inside the staging area or via link to the current version of the table. This feature is a "must have" feature for all people who work in regression testing for systems that use relational databases.

For example, if the system you are getting data from can not deliver deltas or the volumes are small enough that delivering deltas is too much trouble for too little benefit, you can implement delta detection to determine which rows were inserted, updated, or deleted by comparing the minus one version to the current version.

This would include things like partys, addresses, products. Virtually anything can be easily delta compared as long as you have the cpu capacity to do that. With the appliance databases getting faster and faster performing delta detection inside the database is becoming more and more reasonable in cases where the operational system can not tell you what changes happened reliably.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Type	This defines the type of the row that the SeETL[DT] software will interpret the row as. There are some 'helper' values in here to make the code a little easier to write. The valid values are: • **b** – Begin. Each table delta definition should have a 'b' on the first line to warn the generator that it is the beginning of a new view. • **m** – Move. This is the 'normal' or default value of a row type. • **n** – e**N**d. This indicates the last row of the definitions of the columns for the table. The rows that follow will contain special clauses on a database specific basis. • **e** – End. Each table delta definition is required to have an 'end' line to tell the generator to generate the end portions of the create table statement. This just made life a little easier.
Source Database	This is the name of the database from which the table will be sourced. It can be a remote database linked if the data warehouse architect decides that is appropriate. This column is for documentation only as the field "Source First Qualifying Name" from the processing tab will actually be used in the generation of the SQL. However this column will be loaded into the documentation database to make analysis of the documentation database easier.
Source Owner	This is the name of the owner of the table from which the table will be sourced. It can be a remote database linked if the data warehouse architect decides that is appropriate. This column is for documentation only as the field "Source Second Qualifying Name" from the processing tab will actually be used in the generation of the SQL. However this column will be loaded into the documentation database to make analysis of the documentation database easier.
Source Table Name	This is the name of the table (or view) from which the table will be sourced. It can be a remote database linked if the data warehouse architect decides that is appropriate. It is recommended that a view is used and that it is prefixed as something like "zxt_". The reasons for this are as follows: "z" – The view will sort to the bottom of the view list. "xt" – The view is an e**XT**ract view.
Column Number	The ordinal number of the column in the table.
Column Name	The name of the column in the table.
Target Database	If the flag "Propagate Deltas to Target Table" is set to "Yes" then the deltas that are generated will be placed into this target database. This will usually be the staging database.

	This is the name of the database to which the table will be propagated. This column is for documentation only as the field “Target First Qualifying Name” from the processing tab will actually be used in the generation of the SQL. However this column will be loaded into the documentation database to make analysis of the documentation database easier.
Target Owner	If the flag “Propagate Deltas to Target Table” is set to “Yes” then the deltas that are generated will be placed into this owner/schema in the target database. This will usually be the staging database. This is the name of the owner of the table to which the table will be propagated. This column is for documentation only as the field “Target Second Qualifying Name” from the processing tab will actually be used in the generation of the SQL. However this column will be loaded into the documentation database to make analysis of the documentation database easier.
Column Name	The name of the column in the target table. It can be different to the source table.
Column Number	The ordinal number of the column in the target table. It can be different to the source table.
Key Field	This is a flag that defines the column as being one to test for being a key field. In this way rows from the different versions of the tables will be checked for equality of these key fields. Nulls are not allowed in these key fields as per most databases.
Test Field	This is a flag to tell SeETL to generate the test clauses for equality for the field. In this test NULL=NULL is established because the testing includes testing of both fields are null.
User Data 01	User data column that is passed to the data warehouse.
User Data 02	User data column that is passed to the data warehouse.
User Data 03	User data column that is passed to the data warehouse.
User Data 04	User data column that is passed to the data warehouse.
User Data 05	User data column that is passed to the data warehouse.
User Data 06	User data column that is passed to the data warehouse.
User Data 07	User data column that is passed to the data warehouse.
User Data 08	User data column that is passed to the data warehouse.
User Data 09	User data column that is passed to the data warehouse.
User Data 10	User data column that is passed to the data warehouse.
User Data 11	User data column that is passed to the data warehouse.
User Data 12	User data column that is passed to the data warehouse.
User Data 13	User data column that is passed to the data warehouse.
User Data 14	User data column that is passed to the data warehouse.
User Data 15	User data column that is passed to the data warehouse.
User Data 16	User data column that is passed to the data warehouse.
User Data 17	User data column that is passed to the data warehouse.
User Data 18	User data column that is passed to the data warehouse.
User Data 19	User data column that is passed to the data warehouse.
User Data 20	User data column that is passed to the data warehouse.
Row Deleted from Source Indicator	Set to 0
Row Valid Indicator	Set to 1
Row Sent to EDW Indicator	Set to 0
Row Sent to ODS Indicator	Set to 0

4.1.67.Product Definitions

The SeETL[DT] Workbook has been upgraded to be able to store and load product definitions for the BI4ALL Data Models. This is part of the extension of SeETL[DT] and BI4ALL Integration that was started with 3.1.00.

For example, if you are a small firm that sells a few hundred products, or even a few tens of thousands of products, you could keep your product information in this workbook as a master copy and you could load the changes that were made into the staging area to be processed through to the BI4ALL data models.

Field Name	Description
Use	This column tells SeETL[DT] whether the Architect wants this row to be used in processing or not. When it is 'N' the row is treated as a comment and not read by SeETL[DT].
Row Number	This is the number of the row in the workbook. It is used as a key on the staging table.
Product Import Date	This is the date on which the product record was imported. It is in the standard YYYY-MM-DD format.
SKU Number	This is the SKU Number for the Product.
Barcode Number	This is the Barcode Number for the Product.
Effective Date From	This is the effective date from for the Product.
Effective Date To	This is the effective date to for the Product.
Product on Sale Date	This is the Product on Sale Date.
Product Edit Date	This is the last edit date for the Product.
Product Name	This is the Product Name.
Product Sub Group	This is the Product Sub Group.
Product Group	This is the Product Group.
Product Category	This is the Product Category.
Product Class	This is the Product Class.
Product Line	This is the Product Line.
Product Size	This is the Product Size.
Product Colour	This is the Product Colour.
Product Bundle Size	This is the Product Bundle Size.
Shelf Life Days	This is the Shelf Life in Days.
Unit Allocated Cost	This is the Unit Allocated Cost.
Unit ABC Cost	This is the Unit ABC Cost.
List Price per Unit	This is the list price per unit.
Unit of Measure Code	This is the Product Unit of Measure Code.
Discount Allowed Flag	This is the Product Discount Allowed Flag.
Discount Rate 01	This is the Product Discount Rate 01.
Discount Rate 02	This is the Product Discount Rate 02.
Discount Rate 03	This is the Product Discount Rate 03.
Discount Rate 04	This is the Product Discount Rate 04.
Discount Rate 05	This is the Product Discount Rate 05.
Discount Short Description 01	This is the Product Discount Short Description 01.
Discount Short Description 02	This is the Product Discount Short Description 02.
Discount Short Description 03	This is the Product Discount Short Description 03.
Discount Short Description 04	This is the Product Discount Short Description 04.
Discount Short Description 05	This is the Product Discount Short Description 05.
Discount Long Description 01	This is the Product Discount Long Description 01.
Discount Long Description 02	This is the Product Discount Long Description 02.
Discount Long Description 03	This is the Product Discount Long Description 03.

Discount Long Description 04	This is the Product Discount Long Description 04.
Discount Long Description 05	This is the Product Discount Long Description 05.
Returnable Flag	This is the Product Returnable Flag. It indicates whether the product can be returned for a refund or not.
Return Inspection Required Long Description	This is the Product Return Inspection Required Long Description. If the product is returnable and it requires an inspection on return then this field describes the inspection that has to be performed.
Sellable Flag	This is the Product Sellable Flag. It indicates whether this product is sellable or not. Some products are not sellable such as demo products and other freebies.
Component Flag	This is the Product Component Flag. It indicates if the product is a component part for another higher order product. Some products can be sellable on their own as well as being component of higher level products.
Requires Shipping Flag	This is the Product Requires Shipping Flag. It indicates whether the physical product requires shipping or not.
Taxable Flag	This is the Product Taxable Flag. This indicates if the product is subject to some form of taxes.
Tax Type 01 Short Description	This is the Product Tax Type 01 Short Description.
Tax Type 02 Short Description	This is the Product Tax Type 02 Short Description.
User Data 01	User data column that is passed to the data warehouse.
User Data 02	User data column that is passed to the data warehouse.
User Data 03	User data column that is passed to the data warehouse.
User Data 04	User data column that is passed to the data warehouse.
User Data 05	User data column that is passed to the data warehouse.
User Data 06	User data column that is passed to the data warehouse.
User Data 07	User data column that is passed to the data warehouse.
User Data 08	User data column that is passed to the data warehouse.
User Data 09	User data column that is passed to the data warehouse.
User Data 10	User data column that is passed to the data warehouse.
User Data 11	User data column that is passed to the data warehouse.
User Data 12	User data column that is passed to the data warehouse.
User Data 13	User data column that is passed to the data warehouse.
User Data 14	User data column that is passed to the data warehouse.
User Data 15	User data column that is passed to the data warehouse.
User Data 16	User data column that is passed to the data warehouse.
User Data 17	User data column that is passed to the data warehouse.
User Data 18	User data column that is passed to the data warehouse.
User Data 19	User data column that is passed to the data warehouse.
User Data 20	User data column that is passed to the data warehouse.
Row Deleted from Source Indicator	Set to 0
Row Valid Indicator	Set to 1
Row Sent to EDW Indicator	Set to 0
Row Sent to ODS Indicator	Set to 0

4.1.68.List Definitions

The SeETL[DT] Workbook has been upgraded to be able to store List Definitions. These are the names and attributes of lists that might be kept by any company for such things as marketing campaigns. These list definitions can be loaded into the BI4ALL Data Models.

List Definitions	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Number	This is the number of the row in the workbook. It is used as a key on the staging table.
Import Date	This is the date on which the List Definition record was imported. It is in the standard YYYY-MM-DD format.
List Number	This is a reference number for the List Definition. It can be any reference number that has been allocated by any operational system or an operator. It is expected to be unique.
List Name	This is the List Name of the List.
List Short Description	This is the List Short Description of the List.
List Long Description	This is the List Long Description of the List.
List Purpose	This is the List Purpose of the List.
List Owner	This is the List Owner of the List.
List Create Date	This is the date that the List Definition was created.
List Effective Date From	This is the Effective Date From for the List.
List Effective Date To	This is the Effective Date To for the List.
List Last Update Date	This is the Last Update Date for the List.
User Data 01	User data column that is passed to the data warehouse.
User Data 02	User data column that is passed to the data warehouse.
User Data 03	User data column that is passed to the data warehouse.
User Data 04	User data column that is passed to the data warehouse.
User Data 05	User data column that is passed to the data warehouse.
User Data 06	User data column that is passed to the data warehouse.
User Data 07	User data column that is passed to the data warehouse.
User Data 08	User data column that is passed to the data warehouse.
User Data 09	User data column that is passed to the data warehouse.
User Data 10	User data column that is passed to the data warehouse.
User Data 11	User data column that is passed to the data warehouse.
User Data 12	User data column that is passed to the data warehouse.
User Data 13	User data column that is passed to the data warehouse.
User Data 14	User data column that is passed to the data warehouse.
User Data 15	User data column that is passed to the data warehouse.
User Data 16	User data column that is passed to the data warehouse.
User Data 17	User data column that is passed to the data warehouse.
User Data 18	User data column that is passed to the data warehouse.
User Data 19	User data column that is passed to the data warehouse.
User Data 20	User data column that is passed to the data warehouse.
Row Deleted from Source Indicator	Set to 0
Row Valid Indicator	Set to 1
Row Sent to EDW Indicator	Set to 0
Row Sent to ODS Indicator	Set to 0

4.1.69.List Members

The SeETL[DT] Workbook has been upgraded to be able to store List Member Definitions. These are the names and attributes of partys that lists that might be members of lists that are kept by any company for such things as marketing campaigns. These list member definitions can be loaded into the BI4ALL Data Models.

List Definitions	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Number	This is the number of the row in the workbook. It is used as a key on the staging table.
Import Date	This is the date on which the List Definition record was imported. It is in the standard YYYY-MM-DD format.
List Number	This is a reference number for the List Definition. It can be any reference number that has been allocated by any operational system or an operator. It is expected to be unique.
Pk_vm_party	If the user is using BI4ALL then this number should be the primary key for the vm_party record for the party to be solicited. It is used as documentation only. However it is placed onto the solicitation record for extra documentation.
Prefix	This is the normal prefix used for the party as defined in the BI4ALL data models.
First Name	This is the normal first name used for the party as defined in the BI4ALL data models.
Second Name	This is the normal second name used for the party as defined in the BI4ALL data models.
Third Name	This is the normal third name used for the party as defined in the BI4ALL data models.
Surname	This is the normal surname used for the party as defined in the BI4ALL data models.
Suffix	This is the normal suffix used for the party as defined in the BI4ALL data models.
Concatenated Name	This is the normal concatenated name used for the party as defined in the BI4ALL data models.
Address to Name	This is the normal concatenated name used for the party as defined in the BI4ALL data models.
Formal Salutation	This is the normal formal salutation used for the party as defined in the BI4ALL data models.
Informal Salutation	This is the normal informal salutation used for the party as defined in the BI4ALL data models.
Written Salutation	This is the normal written salutation used for the party as defined in the BI4ALL data models.
Email Address	This is the email address if the list member.
Web Address	This is the web address that is associated with the receiver. Usually it will be the receivers LinkedIn address. However it may be a personal email address or a company email address. It is whatever the party has given you to be considered their email web address.
Company	This is the normal company information used for the party as defined in the BI4ALL data models.
Department	This is the normal department information used for the party as defined in the BI4ALL data models.
Job Title	This is the normal job title information used for the party as defined in the BI4ALL data models.
User Data 01	User data column that is passed to the data warehouse.
User Data 02	User data column that is passed to the data warehouse.
User Data 03	User data column that is passed to the data warehouse.
User Data 04	User data column that is passed to the data warehouse.
User Data 05	User data column that is passed to the data warehouse.
User Data 06	User data column that is passed to the data warehouse.
User Data 07	User data column that is passed to the data warehouse.
User Data 08	User data column that is passed to the data warehouse.
User Data 09	User data column that is passed to the data warehouse.
User Data 10	User data column that is passed to the data warehouse.
User Data 11	User data column that is passed to the data warehouse.
User Data 12	User data column that is passed to the data warehouse.
User Data 13	User data column that is passed to the data warehouse.

User Data 14	User data column that is passed to the data warehouse.
User Data 15	User data column that is passed to the data warehouse.
User Data 16	User data column that is passed to the data warehouse.
User Data 17	User data column that is passed to the data warehouse.
User Data 18	User data column that is passed to the data warehouse.
User Data 19	User data column that is passed to the data warehouse.
User Data 20	User data column that is passed to the data warehouse.
Row Deleted from Source Indicator	Set to 0
Row Valid Indicator	Set to 1
Row Sent to EDW Indicator	Set to 0
Row Sent to ODS Indicator	Set to 0

4.1.70.List Member Deletes

The SeETL[DT] Workbook has been upgraded to be able to store List Member Definitions. These are the names and attributes of partys that lists that might be members of lists that are kept by any company for such things as marketing campaigns. These list member definitions can be loaded into the BI4ALL Data Models.

This tab maintains the List Member Deletes so that it is possible to track who was deleted from a list.

List Definitions	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Number	This is the number of the row in the workbook. It is used as a key on the staging table.
Import Date	This is the date on which the List Definition record was imported. It is in the standard YYYY-MM-DD format.
List Number	This is a reference number for the List Definition. It can be any reference number that has been allocated by any operational system or an operator. It is expected to be unique.
Pk_vm_party	If the user is using BI4ALL then this number should be the primary key for the vm_party record for the party to be solicited. It is used as documentation only. However it is placed onto the solicitation record for extra documentation.
Prefix	This is the normal prefix used for the party as defined in the BI4ALL data models.
First Name	This is the normal first name used for the party as defined in the BI4ALL data models.
Second Name	This is the normal second name used for the party as defined in the BI4ALL data models.
Third Name	This is the normal third name used for the party as defined in the BI4ALL data models.
Surname	This is the normal surname used for the party as defined in the BI4ALL data models.
Suffix	This is the normal suffix used for the party as defined in the BI4ALL data models.
Concatenated Name	This is the normal concatenated name used for the party as defined in the BI4ALL data models.
Address to Name	This is the normal concatenated name used for the party as defined in the BI4ALL data models.
Formal Salutation	This is the normal formal salutation used for the party as defined in the BI4ALL data models.
Informal Salutation	This is the normal informal salutation used for the party as defined in the BI4ALL data models.
Written Salutation	This is the normal written salutation used for the party as defined in the BI4ALL data models.
Email Address	This is the email address if the list member.
Web Address	This is the web address that is associated with the receiver. Usually it will be the receivers LinkedIn address. However it may be a personal email address or a company email address. It is whatever the party has given you to be considered their email web address.
Company	This is the normal company information used for the party as defined in the BI4ALL data models.
Department	This is the normal department information used for the party as defined in the BI4ALL data models.
Job Title	This is the normal job title information used for the party as defined in the BI4ALL data models.
Reason Code	This is a code that defines the reason for the List Member being deleted from the List.
Reason Short Description	This is the Short Description as to why the List Member was deleted from the List.
Reason Long Description	This is the Long Description as to why the List Member was deleted from the List.
User Data 01	User data column that is passed to the data warehouse.
User Data 02	User data column that is passed to the data warehouse.
User Data 03	User data column that is passed to the data warehouse.
User Data 04	User data column that is passed to the data warehouse.
User Data 05	User data column that is passed to the data warehouse.
User Data 06	User data column that is passed to the data warehouse.

User Data 07	User data column that is passed to the data warehouse.
User Data 08	User data column that is passed to the data warehouse.
User Data 09	User data column that is passed to the data warehouse.
User Data 10	User data column that is passed to the data warehouse.
User Data 11	User data column that is passed to the data warehouse.
User Data 12	User data column that is passed to the data warehouse.
User Data 13	User data column that is passed to the data warehouse.
User Data 14	User data column that is passed to the data warehouse.
User Data 15	User data column that is passed to the data warehouse.
User Data 16	User data column that is passed to the data warehouse.
User Data 17	User data column that is passed to the data warehouse.
User Data 18	User data column that is passed to the data warehouse.
User Data 19	User data column that is passed to the data warehouse.
User Data 20	User data column that is passed to the data warehouse.
Row Deleted from Source Indicator	Set to 0
Row Valid Indicator	Set to 1
Row Sent to EDW Indicator	Set to 0
Row Sent to ODS Indicator	Set to 0

4.1.71.Client Security Definitions

The SeETL[DT] Workbook has been upgraded to be able to store Client Security Definitions. These are the names and attributes of companies and Partys that are allowed to access portions of a data warehouse. In 3.1.02 we are adding support to the BI4ALL Data Models to optionally allow customers to co-habit data in one database so as to get lower price support for their hosted BI4ALL data warehouse.

In such a data warehouse each row will be tagged with a client number. Each user who accesses the data warehouse will be authenticated and the client number associated with the user will be retrieved. From this point the user will only be able to access rows with the correct client number on the row. This will allow the co-habitation of many clients data warehouses in one database on one machine thus reducing maintenance costs.

Client Security Definitions	
The first set of rows define the Client level Security Definitions. One Client may have many end users that are covered in the security definitions applied.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **client** – Client row for the Client Security Definitions.
Row Number	This is the number of the row in the workbook. It is used as a key on the staging table.
Import Date	This is the date on which the Client Security Definition record was imported. It is in the standard YYYY-MM-DD format.
Client Unique Security Number	This is a Client Unique Security Number that will be placed on to every record that an end user from this client is able to read. The Client Unique Security Numbers must be allocated and maintained by the data warehouse maintenance staff for the data warehouses where this number is used. Failure to maintain unique numbers may be a security concern.
Reference Number	This is a reference number for the Client Security Definition. It can be any reference number that has been allocated by any operational system or an operator. It is expected to be unique.
Client Key	This is the Client Key for the Client Security Definition. It allows the user to use any specific Client Key that may have been assigned to Clients.
Client Number	This is the Client Number for the Client Security Definition.
Client Name	This is the Client Name for the Client Security Definition.
Client Short Description	This is the Client Short Description for the Client Security Definition.
Client Long Description	This is the Client Long Description for the Client Security Definition.
Client Comments 01	This is the Client Comments 01 for the Client Security Definition.
Client Comments 02	This is the Client Comments 02 for the Client Security Definition.
Effective Date From	This is the Effective Date From for the Client Security Definition.
Effective Date To	This is the Effective Date To for the Client Security Definition.
Last Update Date	This is the Last Update Date for the Client Security Definition.
User Data 01	User data column that is passed to the data warehouse.
User Data 02	User data column that is passed to the data warehouse.
User Data 03	User data column that is passed to the data warehouse.
User Data 04	User data column that is passed to the data warehouse.
User Data 05	User data column that is passed to the data warehouse.
User Data 06	User data column that is passed to the data warehouse.
User Data 07	User data column that is passed to the data warehouse.
User Data 08	User data column that is passed to the data warehouse.
User Data 09	User data column that is passed to the data warehouse.
User Data 10	User data column that is passed to the data warehouse.
User Data 11	User data column that is passed to the data warehouse.
User Data 12	User data column that is passed to the data warehouse.
User Data 13	User data column that is passed to the data warehouse.
User Data 14	User data column that is passed to the data warehouse.
User Data 15	User data column that is passed to the data warehouse.
User Data 16	User data column that is passed to the data warehouse.
User Data 17	User data column that is passed to the data warehouse.

User Data 18	User data column that is passed to the data warehouse.
User Data 19	User data column that is passed to the data warehouse.
User Data 20	User data column that is passed to the data warehouse.
Row Deleted from Source Indicator	Set to 0
Row Valid Indicator	Set to 1
Row Sent to EDW Indicator	Set to 0
Row Sent to ODS Indicator	Set to 0

End User Security Definitions

The second set of rows define the End user level Security Definitions.
One Client may have many end users that are covered in the security definitions applied. Therefore the end users need to be linked to the client to be able to know which people from which client have access to what data.

Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **user** – End User row for the Client Security Definitions.
Row Number	This is the number of the row in the workbook. It is used as a key on the staging table.
Import Date	This is the date on which the Client Security Definition record was imported. It is in the standard YYYY-MM-DD format.
Client Unique Security Number	This is a Client Unique Security Number that will be placed on to every record that an end user from this client is able to read. The Client Unique Security Numbers must be allocated and maintained by the data warehouse maintenance staff for the data warehouses where this number is used. Failure to maintain unique numbers may be a security concern.
Reference Number	This is a reference number for the Client Security Definition. It can be any reference number that has been allocated by any operational system or an operator. It is expected to be unique.
Client Key	This is the Client Key for the Client Security Definition. It allows the user to use any specific Client Key that may have been assigned to Clients.
Client Number	This is the Client Number for the Client Security Definition.
Client Name	This is the Client Name for the Client Security Definition.
User Authentication String	This is the User Authentication String for the Client Security Definition. This is a string that must be present in the system that is making a call into the data warehouse. Usually it will be the windows authenticated login identifier or similar. It will be passed into the
User Key	This is the User Key for the End User Security Definition. This User Key field is provided to allow connection to any defined User Key that the client might have internally allocated for the End User.
User Number	This is the User Number for the End User Security Definition. This User Number field is provided to allow connection to any defined User Number that the client might have internally allocated for the End User.
Pk_vm_party	If the user is using BI4ALL then this number should be the primary key for the vm_party record for the party to be solicited. It is used as documentation only. However it is placed onto the solicitation record for extra documentation.
Prefix	This is the normal prefix used for the party as defined in the BI4ALL data models.
First Name	This is the normal first name used for the party as defined in the BI4ALL data models.
Second Name	This is the normal second name used for the party as defined in the BI4ALL data models.
Third Name	This is the normal third name used for the party as defined in the BI4ALL data models.
Surname	This is the normal surname used for the party as defined in the BI4ALL data models.
Suffix	This is the normal suffix used for the party as defined in the BI4ALL data models.
Concatenated Name	This is the normal concatenated name used for the party as defined in the BI4ALL data models.
Address to Name	This is the normal concatenated name used for the party as defined in the BI4ALL data models.
Formal Salutation	This is the normal formal salutation used for the party as defined in the BI4ALL data models.

Informal Salutation	This is the normal informal salutation used for the party as defined in the BI4ALL data models.
Written Salutation	This is the normal written salutation used for the party as defined in the BI4ALL data models.
Email Address	This is the email address if the list member.
Web Address	This is the web address that is associated with the receiver. Usually it will be the receivers LinkedIn address. However it may be a personal email address or a company email address. It is whatever the party has given you to be considered their email web address.
Company	This is the normal company information used for the party as defined in the BI4ALL data models.
Department	This is the normal department information used for the party as defined in the BI4ALL data models.
Job Title	This is the normal job title information used for the party as defined in the BI4ALL data models.
User Comments 01	This is the User Comments 01 for the End User Security Definition.
User Comments 02	This is the User Comments 02 for the End User Security Definition.
Effective Date From	This is the Effective Date From for the End User Security Definition.
Effective Date To	This is the Effective Date To for the End User Security Definition.
Last Update Date	This is the Last Update Date for the End User Security Definition.
User Data 01	User data column that is passed to the data warehouse.
User Data 02	User data column that is passed to the data warehouse.
User Data 03	User data column that is passed to the data warehouse.
User Data 04	User data column that is passed to the data warehouse.
User Data 05	User data column that is passed to the data warehouse.
User Data 06	User data column that is passed to the data warehouse.
User Data 07	User data column that is passed to the data warehouse.
User Data 08	User data column that is passed to the data warehouse.
User Data 09	User data column that is passed to the data warehouse.
User Data 10	User data column that is passed to the data warehouse.
User Data 11	User data column that is passed to the data warehouse.
User Data 12	User data column that is passed to the data warehouse.
User Data 13	User data column that is passed to the data warehouse.
User Data 14	User data column that is passed to the data warehouse.
User Data 15	User data column that is passed to the data warehouse.
User Data 16	User data column that is passed to the data warehouse.
User Data 17	User data column that is passed to the data warehouse.
User Data 18	User data column that is passed to the data warehouse.
User Data 19	User data column that is passed to the data warehouse.
User Data 20	User data column that is passed to the data warehouse.
Row Deleted from Source Indicator	Set to 0
Row Valid Indicator	Set to 1
Row Sent to EDW Indicator	Set to 0
Row Sent to ODS Indicator	Set to 0

4.1.72.Client Software Key Definitions.

SeETL 3.1.02 has been upgrade to support Software Keys. The introduction of Software Keys for consumer purchase is part of our commitment to be able to bring low cost BI development to as many companies as possible. In recent times we have limited our marketing efforts mostly to consulting firms and offering SeETL as a tool to the consulting firm to use with their clients.

We are now moving into the phase of marketing SeETL directly to end user companies because the feature set of SeETL has been dramatically expanded to be useful to companies even if they are not performing Business Intelligence style work. For example in 3.1.01+02 there are email campaigns, delta detection, data profiling, invoice management, sales, returns and voucher key generation just to name a few functions that are well outside the "Business Intelligence" set of features that have characterized SeETL up until 3.1.00.

Therefore the SeETL[DT] Workbook has been upgraded to be able to store Client Software Key Definitions for any company that uses Keys to secure their software. The definitions record the user, the key, the valid period of the key and other related data.

Client Software Key Definitions	
The first set of rows define the Client level Definitions. One Client may have many end users that are covered in the Software Key definitions supplied.	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **client** – Client row for the Client Software Key Definitions.
Row Number	This is the number of the row in the workbook. It is used as a key on the staging table.
Import Date	This is the date on which the Client Software Key Definition record was imported. It is in the standard YYYY-MM-DD format.
Reference Number	This is a reference number for the Client Software Key Definition. It can be any reference number that has been allocated by any operational system or an operator. It is expected to be unique.
Client Key	This is the Client Key for the Client Software Key Definition. It allows the user to use any specific Client Key that may have been assigned to Clients.
Client Number	This is the Client Number for the Client Software Key Definition.
Client Name	This is the Client Name for the Client Software Key Definition.
Client Short Description	This is the Client Short Description for the Client Software Key Definition.
Client Long Description	This is the Client Long Description for the Client Software Key Definition.
Client Comments 01	This is the Client Comments 01 for the Client Software Key Definition.
Client Comments 02	This is the Client Comments 02 for the Client Software Key Definition.
Effective Date From	This is the Effective Date From for the Client Software Key Definition.
Effective Date To	This is the Effective Date To for the Client Software Key Definition.
Last Update Date	This is the Last Update Date for the Client Software Key Definition.
User Data 01	User data column that is passed to the data warehouse.
User Data 02	User data column that is passed to the data warehouse.
User Data 03	User data column that is passed to the data warehouse.
User Data 04	User data column that is passed to the data warehouse.
User Data 05	User data column that is passed to the data warehouse.
User Data 06	User data column that is passed to the data warehouse.
User Data 07	User data column that is passed to the data warehouse.
User Data 08	User data column that is passed to the data warehouse.
User Data 09	User data column that is passed to the data warehouse.
User Data 10	User data column that is passed to the data warehouse.
User Data 11	User data column that is passed to the data warehouse.
User Data 12	User data column that is passed to the data warehouse.
User Data 13	User data column that is passed to the data warehouse.
User Data 14	User data column that is passed to the data warehouse.
User Data 15	User data column that is passed to the data warehouse.
User Data 16	User data column that is passed to the data warehouse.
User Data 17	User data column that is passed to the data warehouse.

User Data 18	User data column that is passed to the data warehouse.
User Data 19	User data column that is passed to the data warehouse.
User Data 20	User data column that is passed to the data warehouse.
Row Deleted from Source Indicator	Set to 0
Row Valid Indicator	Set to 1
Row Sent to EDW Indicator	Set to 0
Row Sent to ODS Indicator	Set to 0

End User Software Key Definitions

The second set of rows define the End user level Software Key Definitions.
One Client may have many end users that have bought Software Keys that need to be tracked and maintained. Therefore the end users need to be linked to the client to be able to know which people from which client have licenses for what software.

Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Type	Row type is set as follows: • **user** – End User row for the Client Security Definitions.
Row Number	This is the number of the row in the workbook. It is used as a key on the staging table.
Import Date	This is the date on which the Client Software Key Definition record was imported. It is in the standard YYYY-MM-DD format.
Reference Number	This is a reference number for the Client Software Key Definition. It can be any reference number that has been allocated by any operational system or an operator. It is expected to be unique.
Client Key	This is the Client Key for the Client Software Key Definition. It allows the user to use any specific Client Key that may have been assigned to Clients.
Client Number	This is the Client Number for the Client Software Key Definition.
Client Name	This is the Client Name for the Client Software Key Definition.
User Authentication String	This is the User Authentication String for the Client Software Key Definition. This is a string that must be present in the system that is making a call into the data warehouse. Usually it will be the windows authenticated login identifier or similar. It will be passed into the
User Key	This is the User Key for the End User Software Key Definition. This User Key field is provided to allow connection to any defined User Key that the client might have internally allocated for the End User.
User Number	This is the User Number for the End User Software Key Definition. This User Number field is provided to allow connection to any defined User Number that the client might have internally allocated for the End User.
Pk_vm_party	If the user is using BI4ALL then this number should be the primary key for the vm_party record for the party to be solicited. It is used as documentation only. However it is placed onto the solicitation record for extra documentation.
Prefix	This is the normal prefix used for the party as defined in the BI4ALL data models.
First Name	This is the normal first name used for the party as defined in the BI4ALL data models.
Second Name	This is the normal second name used for the party as defined in the BI4ALL data models.
Third Name	This is the normal third name used for the party as defined in the BI4ALL data models.
Surname	This is the normal surname used for the party as defined in the BI4ALL data models.
Suffix	This is the normal suffix used for the party as defined in the BI4ALL data models.
Concatenated Name	This is the normal concatenated name used for the party as defined in the BI4ALL data models.
Address to Name	This is the normal concatenated name used for the party as defined in the BI4ALL data models.
Formal Salutation	This is the normal formal salutation used for the party as defined in the BI4ALL data models.
Informal Salutation	This is the normal informal salutation used for the party as defined in the BI4ALL data models.
Written Salutation	This is the normal written salutation used for the party as defined in the BI4ALL data models.

Email Address	This is the email address if the list member.
Web Address	This is the web address that is associated with the receiver. Usually it will be the receivers LinkedIn address. However it may be a personal email address or a company email address. It is whatever the party has given you to be considered their email web address.
Company	This is the normal company information used for the party as defined in the BI4ALL data models.
Department	This is the normal department information used for the party as defined in the BI4ALL data models.
Job Title	This is the normal job title information used for the party as defined in the BI4ALL data models.
Product Key	This is the Product Key of the Product that is related to this Client Software Key Definition.
SKU Number	This is the SKU Number of the Product that is related to this Client Software Key Definition.
Barcode Number	This is the Barcode Number of the Product that is related to this Client Software Key Definition.
Product Name	This is the Product Name of the Product that is related to this Client Software Key Definition.
Product License Key	This is the Product License Key of the Product that is related to this Client Software Key Definition.
User Comments 01	This is the User Comments 01 for the End User Security Definition.
User Comments 02	This is the User Comments 02 for the End User Security Definition.
Effective Date From	This is the Effective Date From for the End User Security Definition.
Effective Date To	This is the Effective Date To for the End User Security Definition.
Last Update Date	This is the Last Update Date for the End User Security Definition.
User Data 01	User data column that is passed to the data warehouse.
User Data 02	User data column that is passed to the data warehouse.
User Data 03	User data column that is passed to the data warehouse.
User Data 04	User data column that is passed to the data warehouse.
User Data 05	User data column that is passed to the data warehouse.
User Data 06	User data column that is passed to the data warehouse.
User Data 07	User data column that is passed to the data warehouse.
User Data 08	User data column that is passed to the data warehouse.
User Data 09	User data column that is passed to the data warehouse.
User Data 10	User data column that is passed to the data warehouse.
User Data 11	User data column that is passed to the data warehouse.
User Data 12	User data column that is passed to the data warehouse.
User Data 13	User data column that is passed to the data warehouse.
User Data 14	User data column that is passed to the data warehouse.
User Data 15	User data column that is passed to the data warehouse.
User Data 16	User data column that is passed to the data warehouse.
User Data 17	User data column that is passed to the data warehouse.
User Data 18	User data column that is passed to the data warehouse.
User Data 19	User data column that is passed to the data warehouse.
User Data 20	User data column that is passed to the data warehouse.
Row Deleted from Source Indicator	Set to 0
Row Valid Indicator	Set to 1
Row Sent to EDW Indicator	Set to 0
Row Sent to ODS Indicator	Set to 0

4.1.73.Party Interests

The SeETL[DT] Workbook has been upgraded to be able to store the interests of partys with respect the items that they might wish to be contacted about. These Party Interests will be loaded into the BI4ALL data models and made available to analysts who wish to select Partys for campaigns of all sorts.

Party Interests Definition	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row Number	This is the number of the row in the workbook. It is used as a key on the staging table.
Import Date	This is the date on which the Party Interests Definition record was imported. It is in the standard YYYY-MM-DD format.
Pk_vm_party	If the user is using BI4ALL then this number should be the primary key for the vm_party record for the party to be solicited. It is used as documentation only. However it is placed onto the solicitation record for extra documentation.
Prefix	This is the normal prefix used for the party as defined in the BI4ALL data models.
First Name	This is the normal first name used for the party as defined in the BI4ALL data models.
Second Name	This is the normal second name used for the party as defined in the BI4ALL data models.
Third Name	This is the normal third name used for the party as defined in the BI4ALL data models.
Surname	This is the normal surname used for the party as defined in the BI4ALL data models.
Suffix	This is the normal suffix used for the party as defined in the BI4ALL data models.
Concatenated Name	This is the normal concatenated name used for the party as defined in the BI4ALL data models.
Address to Name	This is the normal concatenated name used for the party as defined in the BI4ALL data models.
Formal Salutation	This is the normal formal salutation used for the party as defined in the BI4ALL data models.
Informal Salutation	This is the normal informal salutation used for the party as defined in the BI4ALL data models.
Written Salutation	This is the normal written salutation used for the party as defined in the BI4ALL data models.
Email Address	This is the email address if the list member.
Web Address	This is the web address that is associated with the receiver. Usually it will be the receivers LinkedIn address. However it may be a personal email address or a company email address. It is whatever the party has given you to be considered their email web address.
Company	This is the normal company information used for the party as defined in the BI4ALL data models.
Department	This is the normal department information used for the party as defined in the BI4ALL data models.
Job Title	This is the normal job title information used for the party as defined in the BI4ALL data models.
Anonymous Sharing of Interests	This is a Flag to note if the Party has agreed to anonymous sharing of his/her interests so as to be able to avail of more highly targeted offers from other sellers.
Interest Code 01	This is the code that defines an interest. There are 30 codes allowed for any single Party in version 3.1.02.
Interest Code 02	This is the code that defines an interest.
Interest Code 03	This is the code that defines an interest.
Interest Code 04	This is the code that defines an interest.
Interest Code 05	This is the code that defines an interest.
Interest Code 06	This is the code that defines an interest.
Interest Code 07	This is the code that defines an interest.
Interest Code 08	This is the code that defines an interest.
Interest Code 09	This is the code that defines an interest.
Interest Code 10	This is the code that defines an interest.
Interest Code 11	This is the code that defines an interest.
Interest Code 12	This is the code that defines an interest.
Interest Code 13	This is the code that defines an interest.

Interest Code 14	This is the code that defines an interest.
Interest Code 15	This is the code that defines an interest.
Interest Code 16	This is the code that defines an interest.
Interest Code 17	This is the code that defines an interest.
Interest Code 18	This is the code that defines an interest.
Interest Code 19	This is the code that defines an interest.
Interest Code 20	This is the code that defines an interest.
Interest Code 21	This is the code that defines an interest.
Interest Code 22	This is the code that defines an interest.
Interest Code 23	This is the code that defines an interest.
Interest Code 24	This is the code that defines an interest.
Interest Code 25	This is the code that defines an interest.
Interest Code 26	This is the code that defines an interest.
Interest Code 27	This is the code that defines an interest.
Interest Code 28	This is the code that defines an interest.
Interest Code 29	This is the code that defines an interest.
Interest Code 30	This is the code that defines an interest.
User Data 01	User data column that is passed to the data warehouse.
User Data 02	User data column that is passed to the data warehouse.
User Data 03	User data column that is passed to the data warehouse.
User Data 04	User data column that is passed to the data warehouse.
User Data 05	User data column that is passed to the data warehouse.
User Data 06	User data column that is passed to the data warehouse.
User Data 07	User data column that is passed to the data warehouse.
User Data 08	User data column that is passed to the data warehouse.
User Data 09	User data column that is passed to the data warehouse.
User Data 10	User data column that is passed to the data warehouse.
User Data 11	User data column that is passed to the data warehouse.
User Data 12	User data column that is passed to the data warehouse.
User Data 13	User data column that is passed to the data warehouse.
User Data 14	User data column that is passed to the data warehouse.
User Data 15	User data column that is passed to the data warehouse.
User Data 16	User data column that is passed to the data warehouse.
User Data 17	User data column that is passed to the data warehouse.
User Data 18	User data column that is passed to the data warehouse.
User Data 19	User data column that is passed to the data warehouse.
User Data 20	User data column that is passed to the data warehouse.
Row Deleted from Source Indicator	Set to 0
Row Valid Indicator	Set to 1
Row Sent to EDW Indicator	Set to 0
Row Sent to ODS Indicator	Set to 0

4.1.74.Source Table Definitions

The SeETL[DT] Workbook has been upgraded to be able to store the definitions of source tables as part of the automatically generated delta detection processing. These source tables are used to define what tables will be placed in to the staging area via the delta detection process.

Source Table Definitions	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row #	This is the number of the row in the workbook. It is used as a key on the staging table.
Process	In this workbook the user has the option to not process the row if he wishes to. This flag is set to “Y” or “N” depending on whether the user wants the row to be processed.
Database Name	This is the name of the database for the source database. It is used primarily as documentation.
Owner	This is the owner of the source table.
Table Name	This is the name of the source table in the source database.
Description	This is a user entered descriptions of the table from the source database. It is used only as documentation for the table.
User Data 01	User data column that is passed to the staging area.
User Data 02	User data column that is passed to the staging area.
User Data 03	User data column that is passed to the staging area.
User Data 04	User data column that is passed to the staging area.
User Data 05	User data column that is passed to the staging area.
User Data 06	User data column that is passed to the staging area.
User Data 07	User data column that is passed to the staging area.
User Data 08	User data column that is passed to the staging area.
User Data 09	User data column that is passed to the staging area.
User Data 10	User data column that is passed to the staging area.
Notes1..Notes5	There are five cells for notes for each file that can be completed at the Architect’s discretion.

4.1.75.Source Column Definitions

The SeETL[DT] Workbook has been upgraded to be able to store the definitions of source columns from source systems as part of the automatically generated delta detection processing.
These source columns are used to define what columns will be placed in to the staging area via the delta detection process.

Source Column Definitions	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row #	This is the number of the row in the workbook. It is used as a key on the staging table.
Process	In this workbook the user has the option to not process the row if he wishes to. This flag is set to “Y” or “N” depending on whether the user wants the row to be processed.
Database Name	This is the name of the database for the source database. It is used primarily as documentation.
Owner	This is the owner of the source table.
Table Name	This is the name of the source table in the source database.
Column Number	This is the number of the column in the source table.
Source Column Data Type Script	This is the data type text that will be used in database scripts to create tables. It must be correct for the create table statement for the database you are using.
Source Column Data Type	This is documentation for source columns. It will not be used in create table scripts. In general it will be identical to the script column. However, there are cases where the data type has not been properly translated in the script and then these two columns are used for the DBA to set the correct script value.
Source Column Length	The length of the source column according to the database dictionary.
Source Column Precision	The precision of the source column according to the database dictionary.
Source Column Scale	The scale of the source column according to the database dictionary.
Source Column Nullable	This is a flag set to “Y” or “N” to indicate whether the source column is nullable in the source database.
Source Column Required	This is a flag that is set to “Y” or “N” to indicate whether the architect has stated whether the source column is required or not. It is used in later queries to determine if the column should be included in the Source to Staging Mappings.
Primary Key Indicator	This is a flag that is set to “Y” or “N” to indicate whether the column is a component of a primary key in the source table. Each tables should have exactly one primary key defined and the component columns of those keys should be indicated. Some ERPs do not make this data readily available. For example Oracle Applications. Therefore this column may need to be edited by the arcitect to ensure it is correct for all source tables. This column is required to be correct because it is used in the generation of the delta detection sql.
Delta Test Indicator	This is a flag set to “Y” or “N” to indicate whether this column should be used in the delta detection processing to determine if the row has changed. In some ERPs, for example Microsofts NAVision there is a timestamp column that indicates if anything on the row has changed so not all columns need to be inspected for change. In other databases this is not true and all / many columns need to be inspected for change. Please note some databases can not to equality comparisons on some data types. For example SQL Server can not compare image fields. In such cases other mechanisms must be used to determine if the data has changed.
Description	This is a user entered descriptions of the table from the source database. It is used only as documentation for the table.
User Data 01	User data column that is passed to the staging area.
User Data 02	User data column that is passed to the staging area.
User Data 03	User data column that is passed to the staging area.
User Data 04	User data column that is passed to the staging area.
User Data 05	User data column that is passed to the staging area.
User Data 06	User data column that is passed to the staging area.

User Data 07	User data column that is passed to the staging area.
User Data 08	User data column that is passed to the staging area.
User Data 09	User data column that is passed to the staging area.
User Data 10	User data column that is passed to the staging area.
Notes1..Notes5	There are five cells for notes for each file that can be completed at the Architect's discretion.

4.1.76.Source Index Definitions

The SeETL[DT] Workbook has been upgraded to be able to store the definitions of source indexes from source systems as part of the automatically generated delta detection processing.

These source indexes are used to define what columns are parts of primary keys in the source tables and the fact they are primary keys is used in the generation of the delta detection processing as well as in the generation of the landing and staging tables.

Source Index Definitions	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row #	This is the number of the row in the workbook. It is used as a key on the staging table.
Process	In this workbook the user has the option to not process the row if he wishes to. This flag is set to “Y” or “N” depending on whether the user wants the row to be processed.
Database Name	This is the name of the database for the source database. It is used primarily as documentation.
Owner	This is the owner of the source table.
Table Name	This is the name of the source table in the source database.
Index Name	This is the name of the index on the source table in the source database.
Column Number	This is the number of the column in the source table.
Primary Key Indicator	This is a flag that is set to “Y” or “N” to indicate whether the column is a component of a primary key in the source table. Each tables should have exactly one primary key defined and the component columns of those keys should be indicated. Some ERPs do not make this data readily available. For example Oracle Applications. Therefore this column may need to be edited by the arcitect to ensure it is correct for all source tables. This column is required to be correct because it is used in the generation of the delta detection sql.
Unique Index Indicator	This is a flag that is set to “Y” or “N” to indicate whether the index is a unique index. All primary indexes are presumed to be unique indexes. Please note that delta detection does not work on tables that do not have a primary index because there is no primary index to use to link current data to minus one data to check for change.
Clustered Index Indicator	This is a flag that is set to “Y” or “N” to indicate whether the index is a clustered index. It is used primarily as documentation. Clustered indexes mean different things in different databases and it is the Architect who must decide if an index or table will be clustered according to the database and the costs / benefits of clustering in that database.
Description	This is a user entered descriptions of the table from the source database. It is used only as documentation for the table.
User Data 01	User data column that is passed to the staging area.
User Data 02	User data column that is passed to the staging area.
User Data 03	User data column that is passed to the staging area.
User Data 04	User data column that is passed to the staging area.
User Data 05	User data column that is passed to the staging area.
User Data 06	User data column that is passed to the staging area.
User Data 07	User data column that is passed to the staging area.
User Data 08	User data column that is passed to the staging area.
User Data 09	User data column that is passed to the staging area.
User Data 10	User data column that is passed to the staging area.
Notes1..Notes5	There are five cells for notes for each file that can be completed at the Architect’s discretion.

4.1.77.Source To Stage Mapping Definitions

The SeETL[DT] Workbook has been upgraded to be able to store high level maps of the source to stage mappings. This workbook is used to generate a variety of objects for the source to stage mappings including the delta detection sql that performs this task.

Source to Stage Mapping Definitions	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row #	This is the number of the row in the workbook. It is used as a key on the staging table.
Process	In this workbook the user has the option to not process the row if he wishes to. This flag is set to “Y” or “N” depending on whether the user wants the row to be processed.
Source Database Name	This is the name of the database for the source database. It is used primarily as documentation.
Source Owner	This is the owner of the source table.
Source Table Name	This is the name of the source table in the source database.
Source Column Number	This is the number of the column in the source table.
Source Column Name	This is the name of the column in the source table.
Source Column Data Type Script	This is the data type text that will be used in database scripts to create tables. It must be correct for the create table statement for the database you are using.
Source Column Data Type	This is documentation for source columns. It will not be used in create table scripts. In general it will be identical to the script column. However, there are cases where the data type has not been properly translated in the script and then these two columns are used for the DBA to set the correct script value.
Source Column Length	The length of the source column according to the database dictionary.
Source Column Precision	The precision of the source column according to the database dictionary.
Source Column Scale	The scale of the source column according to the database dictionary.
Source Column Nullable	This is a flag set to “Y” or “N” to indicate whether the source column is nullable in the source database.
Source Column Required	This is a flag that is set to “Y” or “N” to indicate whether the architect has stated whether the source column is required or not. It is used in later queries to determine if the column should be included in the Source to Staging Mappings.
Source Primary Key Indicator	This is a flag that is set to “Y” or “N” to indicate whether the column is a component of a primary key in the source table. Each tables should have exactly one primary key defined and the component columns of those keys should be indicated. Some ERPs do not make this data readily available. For example Oracle Applications. Therefore this column may need to be edited by the arcitect to ensure it is correct for all source tables. This column is required to be correct because it is used in the generation of the delta detection sql.
Target Database Name	This is the name of the database for the target database. It is used in sql generation so it shold be accurate.
Target Database Owner	This is the owner of the target tables.
ZIN View Name	This is a view that is created by SeETL to be the input view to ETL processing. You can find examples of these views in the sample database provided with the free product.
ZXT View Name	This is a view that is created by SeETL that sits on top of the ZIN view to be the EXTract view that SeETL will extract data from to send in to the delta detection processing.
Target Column Name	This is the name of the column in the target table.
Target Column Data Type Script	This is the data type text that will be used in database scripts to create tables. It must be correct for the create table statement for the database you are using.
Target Column Data Type	This is documentation for target columns. It will not be used in create table scripts. In general it will be identical to the script column. However, there are cases where the data type has not been properly translated in the script and then these two columns are used for the DBA to set the correct script value.
Target Column Length	The length of the target column according to the database dictionary.
Target Column	The precision of the target column according to the database dictionary.

Precision	
Target Column Scale	The scale of the target column according to the database dictionary.
Target Column Nullable	This is a flag set to “Y” or “N” to indicate whether the target column is nullable in the target database.
Target Column Required	This is a flag that is set to “Y” or “N” to indicate whether the architect has stated whether the Target column is required or not. It is used in later queries to determine if the column should be included in the Target to Staging Mappings.
Target Primary Key Indicator	This is a flag that is set to “Y” or “N” to indicate whether the column is a component of a primary key in the target table.
Delta Test Indicator	This is a flag set to “Y” or “N” to indicate whether this column should be used in the delta detection processing to determine if the row has changed. In some ERPs, for example Microsofts NAVision there is a timestamp column that indicates if anything on the row has changed so not all columns need to be inspected for change. In other databases this is not true and all / many columns need to be inspected for change. Please note some databases can not to equality comparisons on some data types. For example SQL Server can not compare image fields. In such cases other mechanisms must be used to determine if the data has changed.
Description	This is a user entered descriptions of the column in the target database. It is used only as documentation for the table.
User Data 01	User data column that is passed to the staging area.
User Data 02	User data column that is passed to the staging area.
User Data 03	User data column that is passed to the staging area.
User Data 04	User data column that is passed to the staging area.
User Data 05	User data column that is passed to the staging area.
User Data 06	User data column that is passed to the staging area.
User Data 07	User data column that is passed to the staging area.
User Data 08	User data column that is passed to the staging area.
User Data 09	User data column that is passed to the staging area.
User Data 10	User data column that is passed to the staging area.
Notes1..Notes5	There are five cells for notes for each file that can be completed at the Architect’s discretion.

4.1.78.Pre Mapping 01 Definitions

The SeETL[DT] Workbook has been upgraded to be able to store what we have called “Pre Mappings”. These “Pre-Mappings” are high level mappings of staging tables to target data warehouse tables which are placed in to the dictionary and used to generate the mappings to place in to the mapping workbook.

The reason these are so interesting is that this gives about a 7X productivity improvement over just using the mapping spreadsheet.

Pre Mapping 01 Definitions	
Field Name	**Description**
Use	This field indicates if this row in the workbook should be used.
Row #	This is the number of the row in the workbook. It is used as a key on the staging table.
Process	In this workbook the user has the option to not process the row if he wishes to. This flag is set to “Y” or “N” depending on whether the user wants the row to be processed.
Row Type	This currently has no use. It is planned to be able to set a variable to be able to tell the mapping generated code what type of row the mapping row is. However to date this has not been seen as necessary.
SeETL View	This is the name of the view that will be placed in to the SeETL View Name in the mapping spreadsheet when this row is used to generate the rows for the mapping spreadsheet. It is what will end up being the “mapping name”. IBI has a naming standard recommendation for mapping names to sort the view names below names for data warehouse views and to increment the mapping number for those views that have more than one mapping sending data in to the view.
SeETL Table Type	SeETL has 4 table types available: • d1 – Dimension Table Type 1 • d2 – Dimension Table ype 2 • f – Fact Table • fa – Fact Association Table This column must be set to one of these values and it will effect the way the row is used in the generation of rows for the input to the mapping workbook.
Source Database Name	This is the name of the database for the source database. This is usually the staging area and not the original source database.
Source Table Owner	This is the owner of the source tables for this mapping.
Primary Source Table	This is the name of the primary table that will be used in the mapping. This is usually the table that will appear first in the from clause of the view that is generated.
Secondary Source Table	This is the name of the secondardy source tables that will be used in the mapping. There can be many secondary source tables used in one mapping. If that is so then multiple lines will appear in the workbook, one for each secondary table that is used in the mapping.
Primary Source Table Flag	This is a flag set to “Y” or “N” which indicates if this line in the workbook represents the first line of a set of lines to define the mapping. If it is the first line then the primary source table and the secondary source table will be the same value and this flag will be “Y”. If it is a subsequent line to include one or more secondary tables then the secondary table will be set to a different value to the primary table name and this flag will be set to “N”. This is used in the views to generate the code for mappings to be placed in to the spreadsheet for further manipulation. The secondary table name is needed to fetch the names of the columns for the table and place those column names in to the mapping spreadsheet.
Sort Order	This is a field which allows the architect to sort the columns coming out of the views to create the mapping spreadsheet properly. Some tables have more than 100 columns so this field makes it possible to set a secondary table input sort order to a higher

	number to sort the columns from that table further down the view than, say the table with more than 100 columns. It is a field that is available to ensure that the mapping spreadsheet rows are generated in the order the architect wants them generated.
Source Table LDesc	This is a long description field for the primary source table. It is documentation.
Table Type Description	This is a description of the table type. It is documentation.
Target Table Owner	This is the table owner name for the target table.
Target Table	This is the name of the target table for the mapping. Typically it is actually a view. The target table owner an view name will be used to generate the name of the target in the mapping spreadsheet.
Table Number	The BIDA Data Model assigns table numbers to all tables that are not simple code tables. This is to support the co-habitation of different fact tables in the one physical table in databases like Sybase IQ. If you are not using the BIDA Data Model you are welcome to allocated table numbes to your tables to be able to perform a similar function.
Mapping Key	For all mappings a string key must be created to uniquely identify the row of data that it was generated from. For example in dimension tables each row has a dim char key field and a level column which uniquely identify the row. This mapping key would feed the dim char key field in this case. For transctions we allocate a field that is a string field that is the source system unique key. These might even need to contain an element of time if the architect decides to do that. Often times the mapping key is left unset and the mapping key is put in to the mapping spreadsheet directly. It is up to the architect to decide for themselves which way they would like to do this. It is a useful option to have the ability to put the value in to this field.
From Clause	One of the issues we found with the mapping spreadsheet as we used to edit it was that we had to add the from and where clauses to each sets of mappings as we edited them etc. Sure, this is not a big deal, but as the numbers of mappings exploded it became more time consuming to put the from and where clauses in to the workbooks by hand. So this field is the from clause that willl be loaded in to the dictionary and then moved directly in to the mapping workbook in the correct place. It does not reduce the need to write the from clause, it removes the need to place it in to the correct position by hand.
Where Clause	One of the issues we found with the mapping spreadsheet as we used to edit it was that we had to add the from and where clauses to each sets of mappings as we edited them etc. Sure, this is not a big deal, but as the numbers of mappings exploded it became more time consuming to put the from and where clauses in to the workbooks by hand. So this field is the where clause that willl be loaded in to the dictionary and then moved directly in to the mapping workbook in the correct place. It does not reduce the need to write the where clause, it removes the need to place it in to the correct position by hand. Not that from and where clauses are mandatory even if it means you use where 1=1.
Order By Clause Flag	This is a flag that is set to “Y” or “N” depending on whether the user will place an order by clause on the mapping that is being generated. It is relatively rare that this is done but sometimes it has to be done.
Order By Clause	This is the text of the order by clause that will be put into the mapping spreadsheet and then generated in to the view for the mapping if included. It is relatively rare that this is done but sometimes it has to be done.
Group By Clause Flag	This is a flag that is set to “Y” or “N” depending on whether the user will place a group by clause on the mapping that is being generated.

	It is relatively rare that this is done but sometimes it has to be done.
Group By Clause	This is the text of the group by clause that will be put into the mapping spreadsheet and then generated in to the view for the mapping if included. It is relatively rare that this is done but sometimes it has to be done.
Action Required	This is a flag set to “Y” or “N” as a reminder that some form of action is required against this row. It is documentation.
Description	This is a description column for the architect to write a description against the mapping row. It is documentation.
User Data 01	User data column that is passed to the staging area.
User Data 02	User data column that is passed to the staging area.
User Data 03	User data column that is passed to the staging area.
User Data 04	User data column that is passed to the staging area.
User Data 05	User data column that is passed to the staging area.
User Data 06	User data column that is passed to the staging area.
User Data 07	User data column that is passed to the staging area.
User Data 08	User data column that is passed to the staging area.
User Data 09	User data column that is passed to the staging area.
User Data 10	User data column that is passed to the staging area.
Notes1..Notes5	There are five cells for notes for each file that can be completed at the Architect’s discretion.

5. THE SEETL MEMORY USAGE UPGRADES

In the past SeETL used to allocate all the memory that could possibly be needed for all the worksheets that could possible by loaded. It did not intelligently allocated just that memory that was needed. This was because the code that was written to allocate the arrays and structures presumed there would never be that many workbooks or that many rows.

Today there are 80+ workbooks defined to the SeETL application. Many of those workbooks can now contain more than 10,000 rows quite easily. The largest workbook we have is the BI4ALL data model that is just over 30,000 rows now. The workbook itself is 140 MB. When we were building the base code the idea that we would one day have a 140 MB workbook with 30,000 lines in it would have been quite laughable.

Now we have such and the memory allocations were slowing down processing even though our machines are much faster than 10 years ago.

So in current, and all future versions of SeETL the standard we have introduced with 3.1.02 is that only those worksheets that are actually present will have memory allocated for them. Further, we have introduced the standard that all workbooks can have up to 100,000 rows in them.

So the way that 3.1.02 now works is that it will detect if the worksheet is present, if it is present it will allocate space for 100,000 cells in the array to read the workbook into the program and another 100,000 cells in the structure to write the data out to files or databases.

Although it can be well argued that it is excessive to allocate 100,000 rows the reasoning is this. There should be no worksheet that should ever get to 100,000 rows. If it does we should split the workbook. Therefore people will never have to worry about “how many rows does this workbook allow”? They can just assume that until they get to 100,000 rows everything is ok.

We have considered migrating the vb.net code across to C++. And we may do that in the future. If we do that in the future we will take greater care of the memory allocations. But today? Because SeETL is such a massive improvement on any other way of doing this sort of work, the slight delay that it takes to allocate and deallocate memory is more than tolerable in our opinion.

Of course, all feedback from paying clients is welcome.

6. THE SEETL DATA MODEL DOCUMENTATION DATABASE

In the past IBI consultants have documented data models in tools like ErWin and Sybase PowerDesigner. These tools are great tools and we have nothing bad to say about them. However, when delivering packaged data models and customizing them these tools do take a long time and a lot of clicking in order to customize.

Further, these tools do not support multiple language implementations for data models. The language you get is the language you get. Usually English.

The GUI of the tools also 'get in the way' of making 'mass changes' or seeing the details of the model easily. They are great at 'drawing diagrams' but they are not good at doing something like changing the data type of one field in all tables it exists in.

For example. The 'row sent to EDW' flag is usually a char 1 field and set to "Y" or "N". On Netezza, because of the zone maps, it was found to produce a marked improvement if this field was set to 0 or 1 as an integer. Therefore the flags needed to change data type. Such a change is usually quite time consuming and difficult in a standard data modeling tool.

The other major change was this. In 2002 IBI consultants were working on a client where the work space was towards the middle of an open plan office. There was simply no wall space for the data model diagrams to be placed on. This was a real problem as our typical way of working was to plaster the walls with the data models so that we could stand in front of them to read them. The way we understood the data models was graphically by vision.

In the absence of wall space we were having enormous difficulty understanding and retaining the models we were designing in our heads. We were simply overwhelmed by the complexity of the model combined with the lack of ability to 'see it in a large diagram'.

To solve this problem we invented a technique that we have labeled for marketing purposes "everything is linked to everything". This is not strictly true. The truth is "Everything that should be linked to anything else is linked. And if it is not linked then it can be connected so quickly it can be considered to be connected."

This advance meant that it was no longer necessary to retain the diagram of the data model in ones head. It also meant that we could develop more complex data models safe in the knowledge that we could connect the tables if we needed to.

Therefore, when IBI developed the ***BI4ALL Data Models*** we chose to not use a data modeling tool for productivity purposes. We chose to not draw diagrams because they were not necessary. This has worked fine for IBI consultants.

However. In clients we are consistently asked to 'please draw diagrams'. We have resisted this request because of the amount of time and effort we know it takes to draw such diagrams and we know there is little value in doing so.

Finally we have agreed (succumbed to the many requests from our clients) to produce data model diagrams for our data models. The tool we are using to do this is 'Enterprise Architect' from Sparx Systems. This is a nice low cost data modeling tool that seems more than useful in doing such diagrams.

This decision then led to the next problem.

Data modeling tools read databases to draw diagrams and to maintain those diagrams. For example RI constraints can be read from the database to drawn lines between tables. The ***BI4ALL Data Models*** do not implement the finished underlying tables. The ***BI4ALL Data Models*** implements everything through views. This was be design so that the table names were meaningless and can be changed by the client.

No data modeling tools can read views and draw ER diagrams between views. These things need to be drawn by hand. Such an activity would make drawning diagrams FAR too time consuming and expensive. IBI consultants have better things to do.

The solution to these problems is to create what we have called 'The SeETL Documentation Database'. This is a database that is intended for documentation of the data model purposes only.

This documentation database can be created in two ways.

1. If the deployment and documentation databases are on the same database technology platform and both are visible to each other then the tables can be created using 'create table as' from the deployment database.

2. If the deployment and documentation databases are NOT on the same database technology (for example the documentation database might be SQL Server and the deployment database might be Netezza) then the create table statements need to be generated by the SeETL^{DT} to include all fields.

Once the Data Architect has created the tables that are the exact equivalent of the ***BI4ALL*** views that represent the data model there is more work to be done.

The tables must be altered to set Primary Keys and all Referential Integrity Constraints must be applied for Enterprise Archtect to be able to read the database.

In this release we are only supporting SQL Server as the Documentation Database with the setting of all these keys etc. Since this database contains no data, just the data model, it is reasonable for us to expect that all our clients will have one development instance of SQL Server somewhere to store this data model.

In future releases we will support other Documentation Database types.

SeETL^{DT} uses the Data Model Sheets to generate the commands for create tables as well as the command for the alter for Primary Keys and Referential Integrity Constraints.

The files that are produced go into the target directory of the Data Model Views work book.

The names of these files are as follows:

1. BI4ALL_Documentation_Tables_Create_Normal.sql
   This is the SQL that is generated to generate create table statements using normal create table statements with all portions of the create statement in the file.

2. BI4ALL _Documentation_Tables_Create_With_Select.sql
   This is the SQL that is generated to generate create table statements using a create table as syntax. Currently only SQL Server is supported.

3. BI4ALL _Documentation_Tables_Create_Constraints.sql
   This is the file that creates the constraints and the primary keys for all the tables that are created to look exactly like the views of the BI4ALL Data Models.

4. BI4ALL _Documentation_Tables_Drop.sql
   This is the SQL to drop the tables in the Documentation Database.

5. BI4ALL _Documentation_Tables_Drop_Constraints.sql
   This is the SQL Required to drop the constraints from the tables. In most databases the constraints must be dropped before the tables can be dropped.

By running these generated pieces of SQL the DBA can maintain a Documentation Database that reflects the real data model in the target Deployment Database.

7. SQL GENERATION OPTIONS

In this 3.1.01 version of the SeETL[DT] we are truly elated and excited to be bringing our clients the biggest step forward in ETL design, generation and execution since we first brought out the initial version of the SeETL[DT] and SeETL[RT] tools many years ago. The breakthrough that we have created is to be able to generate, from the ***SeETL[DT] Workbook,*** the entire ETL subsystem as a suite of sql statements.

7.1. Configurations for the SQL Generation Option

The directives for the SeETL[DT] Workbook document the configurations for the generation of the SQL for the implementation of the ETL subsystem in SQL. The basic configurations are as follows:

1. One step processing.
2. Two step processing with a copy of the data warehouse inside the staging area.
3. Two step processing without a copy of the data warehouse inside the staging area.
4. In all cases the type 2 dimensions can be configured to be with or without field level sensitivity for testing fields for change.

This section will describe the SQL Generated for each type of processing. By reading the SQL Generated the user will understand what is happening. In the sample sql generated below the source database is known as BI4ALL_01 and the target database is known as BI4ALL_02. All the sql you will see is generated and created from a sample spreadsheet for the Netezza machine and based on a few tables out of the BI4ALL Data Models.

For more information please contact Instant Business Intelligence. All IBI clients can get a copy of this workbook and all the generated code from the workbook. It is easier to read the generated code. The remainder of this code and documentation is intended for prospective clients who are considering engaging IBI on data warehousing work. This tool comprises a component of the 'toolkit' that we bring to our clients.

Prior to explaining the processing of the various configurations there are some general configuration options for the sql generation option that apply to all the specific configuration options. The reader needs to understand these configuration options prior to delving into the specific configurations.

1. Source views.
   The sql generation option is designed to be complementary to the C++ engine option. The SQL Generation portion leverages all prior work and can run along side the C++ engine option in a 'mix and match' manner. The data warehouse architect gets the option of choosing the 'mix and match' of the components he would like to implement. As such, the sql generation option reads data from the source views in exactly the same way the C++ engine option does.

2. Target Views and work files.
   The C++ engine option takes exactly the same approach as all the other major ETL tool vendors. It took the approach that most of the ETL processing must happen in flat files outside the database. Therefore the C++ option engine has views placed over the top of the target tables and passes data from the source views to the target views. It does this via files for fact table processing as fact table processing was designed to be able to split the input files and process records for the same fact table simultaneously. The standard implementation of the C++ engine enables the user to split the input files into either 10 or 100 input files and then to schedule the processing of those files in the scheduler.

   The SQL generation version provides no such features because the netezza machine does not need them. Processing 100M+ rows in a single statement on the netezza machine presents no problem at all. We see no need to provide the 'splitting' capability on netezza.

   What we do see is that it is advisable to have a 'work table' which we called a temp or _t1 table for rows that have been processed through the first stage of the ETL processing. The reason we wished to have _t1 tables was for re-runability. The first step might include massive amounts of processing, as you will soon read. We determined that this processing should complete prior to then applying the new rows to the target table so that the processing load was broken up from the creation of the 'deltas' to the application of those deltas to the target tables. These _t1 tables are named <mapping_name>_t1.

3. Sequence Numbers.
   The C++ engine option maintains and manages its own sequence numbers. These numbers are not available to the SQL Version. They may be used in a customised manner but the management of sequence numbers must be designed by the architect depending on the database being used. On Netezza there is an option for the generation of sequence numbers and we recommend this option be used when deploying on Netezza.

7.1.1. One Step Processing

The one step processing option pre-supposes that the client is happy to have single statements moving data between the source database and target database and is not concerned to minimise the interactions between the source and target databases.

This configuration uses the least amount of redundant data and the minimum number of steps. It is the simplest approach as well.

There are 5 different patterns of SQL generated for one step processing. They are:

1. Type 1 dimension table processing
2. Type 2 dimension table processing without field level sensitivity.
3. Type 2 dimension table processing with field level sensitivity.
4. Fact table processing without a key on the fact table.
5. Fact table processing with a key on the fact table.

In this section we will list all these types of processing.

7.1.1.1. Type 1 dimension table processing

The first step in one step processing is always to truncate the _t1 table and then to populate the _t1 table using the source view to perform all the 'etl processing' in selection of the view. Note the documentation presented at the start of the file.

```
-------------------------------------------------------------------------------------------------------------------------
--                                                                                                                     --
--   spreadsheet name:        bi4allmodeldemoworksheet01.xml spreadsheet version:        v3.1.00                        --
--   spreadsheet file name:   c:\seetl\designtime\3.1.00\sourcecode\seetl\bi4allmodeldemoworksheet01v3.1.00.xml         --
--   file name:               c:\temp\dmltype4\dim1p_z01_vm_address_01_dim1s_01.sql                                     --
--   windows domain:          emsbidev01                                                                               --
--   windows user:            administrator                                                                            --
--   windows workstation:     emsbidev01                                                                               --
--   seetl current directory: c:\seetl\designtime\3.1.00\sourcecode\seetl                                               --
--   windows version:         microsoft windows nt 5.2.3790 service pack 2                                             --
--   dot net version:         2.0.50727.1433                                                                           --
--   seetl workbench version: SeETL 3.1.00 - 2010-06-11                                                                --
--   timestamp:               25/07/2010 09:52:55                                                                      --
--   description:             description                                                                              --
--                                                                                                                     --
-------------------------------------------------------------------------------------------------------------------------

truncate table bi4all_01.admin.z01_vm_address_01_t1   ←Truncate the _t1 table.
;

insert into bi4all_01.admin.z01_vm_address_01_t1     ← Insert into the _t1 table.
(
 pk_vm_address                ← Note the sequence field is placed into the primary key field.
,level_col
,dim_char_ky_fld              ← Note the dim char key field must be provided as an input string and can not be constructed by directives.
,vm_address_seq               ← Note the sequence field is stored redundantly.
,address_key_str
,drop_point_id
,address_line_1
,address_line_2
,address_line_3
,address_line_4
,suburb
,city
```

```
,state
,province
,country
,postcode
,geo_code
,email
,web_address
,building_type_code
,building_type_sdesc
,building_type_ldesc
,delivery_hours_range
,delivery_days_range
,paper_mail_flag
,email_flag
,private_residence_flag
,business_residence_flag
,access_letter_delivery_flag
,access_paper_delivery_flag
,access_package_delivery_flag
,row_del_frm_src_ind
,table_number
,batch_number
,file_cycle_number
,view_number
,audit_timestamp_01
)
select
 z01_vm_address_01.z01_vm_address_01_seq          ← Note the sequence.
,'detail'
,z01_vm_address_01.z01_vm_address_01_key          ← Note the key comes out of the view as a string up to varchar 255 and must be prepared in the view
,z01_vm_address_01.z01_vm_address_01_seq          ← Note the sequence
,z01_vm_address_01.address_key_str
,z01_vm_address_01.drop_point_id
,z01_vm_address_01.address_line_1
,z01_vm_address_01.address_line_2
,z01_vm_address_01.address_line_3
,z01_vm_address_01.address_line_4
,z01_vm_address_01.suburb
,z01_vm_address_01.city
,z01_vm_address_01.state
,z01_vm_address_01.province
,z01_vm_address_01.country
```

```
,z01_vm_address_01.postcode
,z01_vm_address_01.geo_code
,z01_vm_address_01.email
,z01_vm_address_01.web_address
,z01_vm_address_01.building_type_code
,z01_vm_address_01.building_type_sdesc
,z01_vm_address_01.building_type_ldesc
,z01_vm_address_01.delivery_hours_range
,z01_vm_address_01.delivery_days_range
,z01_vm_address_01.paper_mail_flag
,z01_vm_address_01.email_flag
,z01_vm_address_01.private_residence_flag
,z01_vm_address_01.business_residence_flag
,z01_vm_address_01.access_letter_delivery_flag
,z01_vm_address_01.access_paper_delivery_flag
,z01_vm_address_01.access_package_delivery_flag
,z01_vm_address_01.row_del_frm_src_ind
,z01_vm_address_01.table_number
,z01_vm_address_01.batch_number
,z01_vm_address_01.file_cycle_number
,z01_vm_address_01.view_number
,z01_vm_address_01.audit_timestamp_01
from bi4all_01.admin.z01_vm_address_01 z01_vm_address_01     ← Note the source database and source owner.
limit 100
;
```

Once the changes have been pushed through to the _t1 table they must be applied to the target table as inserts or updates depending on whether the row already exists in the target table.

```
-----------------------------------------------------------------------------------------------------------------------------
--                                                                                                                         --
--   spreadsheet name:        bi4allmodeldemoworksheet01.xml spreadsheet version:         v3.1.00                         --
--   spreadsheet file name:   c:\seetl\designtime\3.1.00\sourcecode\seetl\bi4allmodeldemoworksheet01v3.1.00.xml          --
--   file name:               c:\temp\dmltype4\dim1p_z01_vm_address_01_dim1s_02.sql                                       --
--   windows domain:          emsbidev01                                                                                 --
--   windows user:            administrator                                                                              --
--   windows workstation:     emsbidev01                                                                                 --
--   seetl current directory: c:\seetl\designtime\3.1.00\sourcecode\seetl                                                 --
--   windows version:         microsoft windows nt 5.2.3790 service pack 2                                               --
--   dot net version:         2.0.50727.1433                                                                             --
--   seetl workbench version: SeETL 3.1.00 - 2010-06-11                                                                  --
--   timestamp:               25/07/2010 09:52:55                                                                        --
--   description:             description                                                                                --
--                                                                                                                         --
-----------------------------------------------------------------------------------------------------------------------------

update bi4all_02.admin.tm_address set
 level_col                                                    = z01_vm_address_01_t1.level_col
,dim_char_ky_fld                                              = z01_vm_address_01_t1.dim_char_ky_fld
,address_key_str                                              = z01_vm_address_01_t1.address_key_str
,drop_point_id                                                = z01_vm_address_01_t1.drop_point_id
,address_line_1                                               = z01_vm_address_01_t1.address_line_1
,address_line_2                                               = z01_vm_address_01_t1.address_line_2
,address_line_3                                               = z01_vm_address_01_t1.address_line_3
,address_line_4                                               = z01_vm_address_01_t1.address_line_4
,suburb                                                       = z01_vm_address_01_t1.suburb
,city                                                         = z01_vm_address_01_t1.city
,state                                                        = z01_vm_address_01_t1.state
,province                                                     = z01_vm_address_01_t1.province
,country                                                      = z01_vm_address_01_t1.country
,postcode                                                     = z01_vm_address_01_t1.postcode
,geo_code                                                     = z01_vm_address_01_t1.geo_code
,email                                                        = z01_vm_address_01_t1.email
,web_address                                                  = z01_vm_address_01_t1.web_address
,building_type_code                                           = z01_vm_address_01_t1.building_type_code
,building_type_sdesc                                          = z01_vm_address_01_t1.building_type_sdesc
,building_type_ldesc                                          = z01_vm_address_01_t1.building_type_ldesc
,delivery_hours_range                                         = z01_vm_address_01_t1.delivery_hours_range
,delivery_days_range                                          = z01_vm_address_01_t1.delivery_days_range
,paper_mail_flag                                              = z01_vm_address_01_t1.paper_mail_flag
,email_flag                                                   = z01_vm_address_01_t1.email_flag
,private_residence_flag                                       = z01_vm_address_01_t1.private_residence_flag
```

```
,business_residence_flag                                          = z01_vm_address_01_t1.business_residence_flag
,access_letter_delivery_flag                                      = z01_vm_address_01_t1.access_letter_delivery_flag
,access_paper_delivery_flag                                       = z01_vm_address_01_t1.access_paper_delivery_flag
,access_package_delivery_flag                                     = z01_vm_address_01_t1.access_package_delivery_flag
,row_del_frm_src_ind                                              = z01_vm_address_01_t1.row_del_frm_src_ind
,table_number                                                     = z01_vm_address_01_t1.table_number
,batch_number                                                     = z01_vm_address_01_t1.batch_number
,file_cycle_number                                                = z01_vm_address_01_t1.file_cycle_number
,view_number                                                      = z01_vm_address_01_t1.view_number
,audit_timestamp_01                                               = z01_vm_address_01_t1.audit_timestamp_01
from bi4all_01.admin.z01_vm_address_01_t1 z01_vm_address_01_t1
where 1=1
and bi4all_02.admin.tm_address.level_col                            = z01_vm_address_01_t1.level_col
and bi4all_02.admin.tm_address.dim_char_ky_fld                      = z01_vm_address_01_t1.dim_char_ky_fld
;
```

Note the test to see if the target table level col and dim char key field are equal. Notice that the target is in database BI4ALL_02.

The third step is to insert rows that do not exist in the target table.

```
-------------------------------------------------------------------------------------------------------------------------
--                                                                                                                     --
--   spreadsheet name:        bi4allmodeldemoworksheet01.xml spreadsheet version:        v3.1.00                        --
--   spreadsheet file name:   c:\seetl\designtime\3.1.00\sourcecode\seetl\bi4allmodeldemoworksheet01v3.1.00.xml         --
--   file name:               c:\temp\dmltype4\dim1p_z01_vm_address_01_dim1s_03.sql                                     --
--   windows domain:          emsbidev01                                                                               --
--   windows user:            administrator                                                                            --
--   windows workstation:     emsbidev01                                                                               --
--   seetl current directory: c:\seetl\designtime\3.1.00\sourcecode\seetl                                               --
--   windows version:         microsoft windows nt 5.2.3790 service pack 2                                             --
--   dot net version:         2.0.50727.1433                                                                           --
--   seetl workbench version: SeETL 3.1.00 - 2010-06-11                                                                --
--   timestamp:               25/07/2010 09:52:55                                                                      --
--   description:             description                                                                              --
--                                                                                                                     --
-------------------------------------------------------------------------------------------------------------------------

insert into bi4all_02.admin.tm_address
(
 pk_tm_address
,level_col
,dim_char_ky_fld
,tm_address_seq
,address_key_str
,drop_point_id
,address_line_1
,address_line_2
,address_line_3
,address_line_4
,suburb
,city
,state
,province
,country
,postcode
,geo_code
,email
,web_address
,building_type_code
```

```
,building_type_sdesc
,building_type_ldesc
,delivery_hours_range
,delivery_days_range
,paper_mail_flag
,email_flag
,private_residence_flag
,business_residence_flag
,access_letter_delivery_flag
,access_paper_delivery_flag
,access_package_delivery_flag
,row_del_frm_src_ind
,table_number
,batch_number
,file_cycle_number
,view_number
,audit_timestamp_01
)
select
 z01_vm_address_01_t1.pk_vm_address
,z01_vm_address_01_t1.level_col
,z01_vm_address_01_t1.dim_char_ky_fld
,z01_vm_address_01_t1.vm_address_seq
,z01_vm_address_01_t1.address_key_str
,z01_vm_address_01_t1.drop_point_id
,z01_vm_address_01_t1.address_line_1
,z01_vm_address_01_t1.address_line_2
,z01_vm_address_01_t1.address_line_3
,z01_vm_address_01_t1.address_line_4
,z01_vm_address_01_t1.suburb
,z01_vm_address_01_t1.city
,z01_vm_address_01_t1.state
,z01_vm_address_01_t1.province
,z01_vm_address_01_t1.country
,z01_vm_address_01_t1.postcode
,z01_vm_address_01_t1.geo_code
,z01_vm_address_01_t1.email
,z01_vm_address_01_t1.web_address
,z01_vm_address_01_t1.building_type_code
,z01_vm_address_01_t1.building_type_sdesc
,z01_vm_address_01_t1.building_type_ldesc
,z01_vm_address_01_t1.delivery_hours_range
```

```
,z01_vm_address_01_t1.delivery_days_range
,z01_vm_address_01_t1.paper_mail_flag
,z01_vm_address_01_t1.email_flag
,z01_vm_address_01_t1.private_residence_flag
,z01_vm_address_01_t1.business_residence_flag
,z01_vm_address_01_t1.access_letter_delivery_flag
,z01_vm_address_01_t1.access_paper_delivery_flag
,z01_vm_address_01_t1.access_package_delivery_flag
,z01_vm_address_01_t1.row_del_frm_src_ind
,z01_vm_address_01_t1.table_number
,z01_vm_address_01_t1.batch_number
,z01_vm_address_01_t1.file_cycle_number
,z01_vm_address_01_t1.view_number
,z01_vm_address_01_t1.audit_timestamp_01
from bi4all_01.admin.z01_vm_address_01_t1 z01_vm_address_01_t1
left join bi4all_02.admin.tm_address tm_address
on
(
tm_address.level_col = z01_vm_address_01_t1.level_col
and
tm_address.dim_char_ky_fld = z01_vm_address_01_t1.dim_char_ky_fld
)
where tm_address.dim_char_ky_fld is null
;
```

Note that the left outer join was decided upon to give the netezza optimiser a better change to be more efficient than a subselect. It was recommended to us that the sub select not be used in a statement like this on the netezza machine. It may well be that on other databases that a subselect might be more efficient. We will be seeking feedback from clients as to which they find more appropriate for their database.

7.1.1.2. Type 2 dimension table processing without field level sensitivity.

```
-------------------------------------------------------------------------------------------------------------------------
--                                                                                                                     --
--   spreadsheet name:        bi4allmodeldemoworksheet01.xml spreadsheet version:        v3.1.00                        --
--   spreadsheet file name:   c:\seetl\designtime\3.1.00\sourcecode\seetl\bi4allmodeldemoworksheet01v3.1.00.xml         --
--   file name:               c:\temp\dmltype4\dim2p_z01_vm_product_01_dim2s_01.sql                                      --
--   windows domain:          emsbidev01                                                                                --
--   windows user:            administrator                                                                             --
--   windows workstation:     emsbidev01                                                                                --
--   seetl current directory: c:\seetl\designtime\3.1.00\sourcecode\seetl                                               --
--   windows version:         microsoft windows nt 5.2.3790 service pack 2                                              --
--   dot net version:         2.0.50727.1433                                                                            --
--   seetl workbench version: SeETL 3.1.00 - 2010-06-11                                                                 --
--   timestamp:               25/07/2010 09:52:55                                                                       --
--   description:             description                                                                               --
--                                                                                                                     --
-------------------------------------------------------------------------------------------------------------------------

truncate table bi4all_01.admin.z01_vm_product_01_t1
;

insert into bi4all_01.admin.z01_vm_product_01_t1
(
 pk_vm_product
,level_col
,dim_char_ky_fld
,vm_product_seq
,current_flag
,date_from
,date_to
,product_number
,sku_number
,barcode_num
,prod_name
,prod_sub_group
,prod_group
,prod_category
```

```
,prod_class
,prod_line
,prod_size
,prod_colour
,prod_bundle_size
,shelf_life_days
,unit_alloc_cost
,unit_abc_cost
,list_price_per_unit
,unit_of_measure_code
,unit_of_measure_sdesc
,discount_allowed
,discount_rate_1
,discount_rate_2
,discount_rate_3
,discount_rate_4
,discount_rate_5
,discount_sdesc_1
,discount_ldesc_1
,discount_sdesc_2
,discount_ldesc_2
,discount_sdesc_3
,discount_ldesc_3
,discount_sdesc_4
,discount_ldesc_4
,discount_sdesc_5
,discount_ldesc_5
,returnable_flag
,return_inspection_req_ldesc
,sellable_flag
,component_flag
,requires_shipping_flag
,taxable_flag
,tax_type_1_sdesc
,tax_type_2_sdesc
,row_del_frm_src_ind
,table_number
,batch_number
,file_cycle_number
,view_number
,audit_timestamp_01
)
```

```
select
 z01_vm_product_01.z01_vm_product_01_seq
,'detail'
,z01_vm_product_01.z01_vm_product_01_key
,z01_vm_product_01.z01_vm_product_01_seq
,1
,ctl_batch_control.batch_date
,ctl_month_control.high_value_date
,z01_vm_product_01.product_number
,z01_vm_product_01.sku_number
,z01_vm_product_01.barcode_num
,z01_vm_product_01.prod_name
,z01_vm_product_01.prod_sub_group
,z01_vm_product_01.prod_group
,z01_vm_product_01.prod_category
,z01_vm_product_01.prod_class
,z01_vm_product_01.prod_line
,z01_vm_product_01.prod_size
,z01_vm_product_01.prod_colour
,z01_vm_product_01.prod_bundle_size
,z01_vm_product_01.shelf_life_days
,z01_vm_product_01.unit_alloc_cost
,z01_vm_product_01.unit_abc_cost
,z01_vm_product_01.list_price_per_unit
,z01_vm_product_01.unit_of_measure_code
,z01_vm_product_01.unit_of_measure_sdesc
,z01_vm_product_01.discount_allowed
,z01_vm_product_01.discount_rate_1
,z01_vm_product_01.discount_rate_2
,z01_vm_product_01.discount_rate_3
,z01_vm_product_01.discount_rate_4
,z01_vm_product_01.discount_rate_5
,z01_vm_product_01.discount_sdesc_1
,z01_vm_product_01.discount_ldesc_1
,z01_vm_product_01.discount_sdesc_2
,z01_vm_product_01.discount_ldesc_2
,z01_vm_product_01.discount_sdesc_3
,z01_vm_product_01.discount_ldesc_3
,z01_vm_product_01.discount_sdesc_4
,z01_vm_product_01.discount_ldesc_4
,z01_vm_product_01.discount_sdesc_5
,z01_vm_product_01.discount_ldesc_5
```

```
,z01_vm_product_01.returnable_flag
,z01_vm_product_01.return_inspection_req_ldesc
,z01_vm_product_01.sellable_flag
,z01_vm_product_01.component_flag
,z01_vm_product_01.requires_shipping_flag
,z01_vm_product_01.taxable_flag
,z01_vm_product_01.tax_type_1_sdesc
,z01_vm_product_01.tax_type_2_sdesc
,z01_vm_product_01.row_del_frm_src_ind
,z01_vm_product_01.table_number
,z01_vm_product_01.batch_number
,z01_vm_product_01.file_cycle_number
,z01_vm_product_01.view_number
,z01_vm_product_01.audit_timestamp_01

from bi4all_01.admin.z01_vm_product_01 z01_vm_product_01
left outer join bi4all_02.admin.ctl_batch_control ctl_batch_control on 1=1
and ctl_batch_control.batch_complete_flag = 0

left outer join bi4all_02.admin.ctl_month_control ctl_month_control on 1=1
limit 100
;
```

The next step is to close out the rows that are about to be inserted as being 'new versions' of existing rows. Not that with no field level sensitivity ALL rows that have the same dim char key field will be replaced.

```
-------------------------------------------------------------------------------------------------------------------------
--                                                                                                                     --
--   spreadsheet name:        bi4allmodeldemoworksheet01.xml spreadsheet version:         v3.1.00                      --
--   spreadsheet file name:   c:\seetl\designtime\3.1.00\sourcecode\seetl\bi4allmodeldemoworksheet01v3.1.00.xml        --
--   file name:               c:\temp\dmltype4\dim2p_z01_vm_product_01_dim2s_02.sql                                      --
--   windows domain:          emsbidev01                                                                                --
--   windows user:            administrator                                                                             --
--   windows workstation:     emsbidev01                                                                                --
--   seetl current directory: c:\seetl\designtime\3.1.00\sourcecode\seetl                                               --
--   windows version:         microsoft windows nt 5.2.3790 service pack 2                                              --
--   dot net version:         2.0.50727.1433                                                                            --
--   seetl workbench version: SeETL 3.1.00 - 2010-06-11                                                                 --
--   timestamp:               25/07/2010 10:20:33                                                                       --
--   description:             description                                                                               --
--                                                                                                                     --
-------------------------------------------------------------------------------------------------------------------------

update bi4all_02.admin.tm_product set
 current_flag                                                                    = 0
,date_to                                                                         = ctl_batch_control.batch_date - 1
from bi4all_01.admin.z01_vm_product_01_t1 z01_vm_product_01_t1
left outer join bi4all_02.admin.ctl_batch_control ctl_batch_control on 1=1
and ctl_batch_control.batch_complete_flag = 0
where 1=1
and tm_product.level_col                                                         = z01_vm_product_01_t1.level_col
and tm_product.dim_char_ky_fld                                                   = z01_vm_product_01_t1.dim_char_ky_fld
and tm_product.current_flag                                                      = 1
;
```

The next step is to insert the rows that are considered new given that they exist in the _t1 table.

```
-------------------------------------------------------------------------------------------------------------------------
--                                                                                                                     --
--   spreadsheet name:        bi4allmodeldemoworksheet01.xml spreadsheet version:        v3.1.00                        --
--   spreadsheet file name:   c:\seetl\designtime\3.1.00\sourcecode\seetl\bi4allmodeldemoworksheet01v3.1.00.xml         --
--   file name:               c:\temp\dmltype4\dim2p_z01_vm_product_01_dim2s_03.sql                                     --
--   windows domain:          emsbidev01                                                                                --
--   windows user:            administrator                                                                             --
--   windows workstation:     emsbidev01                                                                                --
--   seetl current directory: c:\seetl\designtime\3.1.00\sourcecode\seetl                                               --
--   windows version:         microsoft windows nt 5.2.3790 service pack 2                                              --
--   dot net version:         2.0.50727.1433                                                                            --
--   seetl workbench version: SeETL 3.1.00 - 2010-06-11                                                                 --
--   timestamp:               25/07/2010 10:20:33                                                                       --
--   description:             description                                                                               --
--                                                                                                                     --
-------------------------------------------------------------------------------------------------------------------------

insert into bi4all_02.admin.tm_product
(
 pk_tm_product
,level_col
,dim_char_ky_fld
,current_flag
,date_from
,date_to
,tm_product_seq
,product_number
,sku_number
,barcode_num
,prod_name
,prod_sub_group
,prod_group
,prod_category
,prod_class
,prod_line
,prod_size
,prod_colour
,prod_bundle_size
,shelf_life_days
```

```
,unit_alloc_cost
,unit_abc_cost
,list_price_per_unit
,unit_of_measure_code
,unit_of_measure_sdesc
,discount_allowed
,discount_rate_1
,discount_rate_2
,discount_rate_3
,discount_rate_4
,discount_rate_5
,discount_sdesc_1
,discount_ldesc_1
,discount_sdesc_2
,discount_ldesc_2
,discount_sdesc_3
,discount_ldesc_3
,discount_sdesc_4
,discount_ldesc_4
,discount_sdesc_5
,discount_ldesc_5
,returnable_flag
,return_inspection_req_ldesc
,sellable_flag
,component_flag
,requires_shipping_flag
,taxable_flag
,tax_type_1_sdesc
,tax_type_2_sdesc
,row_del_frm_src_ind
,table_number
,batch_number
,file_cycle_number
,view_number
,audit_timestamp_01
)
select
 z01_vm_product_01_t1.pk_vm_product
,z01_vm_product_01_t1.level_col
,z01_vm_product_01_t1.dim_char_ky_fld
,1
,z01_vm_product_01_t1.date_from
```

```
,z01_vm_product_01_t1.date_to
,z01_vm_product_01_t1.vm_product_seq
,z01_vm_product_01_t1.product_number
,z01_vm_product_01_t1.sku_number
,z01_vm_product_01_t1.barcode_num
,z01_vm_product_01_t1.prod_name
,z01_vm_product_01_t1.prod_sub_group
,z01_vm_product_01_t1.prod_group
,z01_vm_product_01_t1.prod_category
,z01_vm_product_01_t1.prod_class
,z01_vm_product_01_t1.prod_line
,z01_vm_product_01_t1.prod_size
,z01_vm_product_01_t1.prod_colour
,z01_vm_product_01_t1.prod_bundle_size
,z01_vm_product_01_t1.shelf_life_days
,z01_vm_product_01_t1.unit_alloc_cost
,z01_vm_product_01_t1.unit_abc_cost
,z01_vm_product_01_t1.list_price_per_unit
,z01_vm_product_01_t1.unit_of_measure_code
,z01_vm_product_01_t1.unit_of_measure_sdesc
,z01_vm_product_01_t1.discount_allowed
,z01_vm_product_01_t1.discount_rate_1
,z01_vm_product_01_t1.discount_rate_2
,z01_vm_product_01_t1.discount_rate_3
,z01_vm_product_01_t1.discount_rate_4
,z01_vm_product_01_t1.discount_rate_5
,z01_vm_product_01_t1.discount_sdesc_1
,z01_vm_product_01_t1.discount_ldesc_1
,z01_vm_product_01_t1.discount_sdesc_2
,z01_vm_product_01_t1.discount_ldesc_2
,z01_vm_product_01_t1.discount_sdesc_3
,z01_vm_product_01_t1.discount_ldesc_3
,z01_vm_product_01_t1.discount_sdesc_4
,z01_vm_product_01_t1.discount_ldesc_4
,z01_vm_product_01_t1.discount_sdesc_5
,z01_vm_product_01_t1.discount_ldesc_5
,z01_vm_product_01_t1.returnable_flag
,z01_vm_product_01_t1.return_inspection_req_ldesc
,z01_vm_product_01_t1.sellable_flag
,z01_vm_product_01_t1.component_flag
,z01_vm_product_01_t1.requires_shipping_flag
,z01_vm_product_01_t1.taxable_flag
```

```
,z01_vm_product_01_t1.tax_type_1_sdesc
,z01_vm_product_01_t1.tax_type_2_sdesc
,z01_vm_product_01_t1.row_del_frm_src_ind
,z01_vm_product_01_t1.table_number
,z01_vm_product_01_t1.batch_number
,z01_vm_product_01_t1.file_cycle_number
,z01_vm_product_01_t1.view_number
,z01_vm_product_01_t1.audit_timestamp_01
from bi4all_01.admin.z01_vm_product_01_t1 z01_vm_product_01_t1
;
```

Note that ALL rows in the _t1 table are inserted as they are all considered new.

7.1.1.3. Type 2 dimension table processing with field level sensitivity.

```
-------------------------------------------------------------------------------------------------------------------------
--                                                                                                                     --
--   spreadsheet name:        bi4allmodeldemoworksheet01.xml spreadsheet version:        v3.1.00                        --
--   spreadsheet file name:   c:\seetl\designtime\3.1.00\sourcecode\seetl\bi4allmodeldemoworksheet01v3.1.00.xml         --
--   file name:               c:\temp\dmltype4\dim2p_z01_vm_product_01_dim2s_01.sql                                      --
--   windows domain:          emsbidev01                                                                                --
--   windows user:            administrator                                                                             --
--   windows workstation:     emsbidev01                                                                                --
--   seetl current directory: c:\seetl\designtime\3.1.00\sourcecode\seetl                                               --
--   windows version:         microsoft windows nt 5.2.3790 service pack 2                                              --
--   dot net version:         2.0.50727.1433                                                                            --
--   seetl workbench version: SeETL 3.1.00 - 2010-06-11                                                                 --
--   timestamp:               25/07/2010 09:52:55                                                                       --
--   description:             description                                                                               --
--                                                                                                                     --
-------------------------------------------------------------------------------------------------------------------------

truncate table bi4all_01.admin.z01_vm_product_01_t1
;

insert into bi4all_01.admin.z01_vm_product_01_t1
(
 pk_vm_product
,level_col
,dim_char_ky_fld
,vm_product_seq
,current_flag
,date_from
,date_to
,product_number
,sku_number
,barcode_num
,prod_name
,prod_sub_group
,prod_group
,prod_category
,prod_class
,prod_line
```

```
,prod_size
,prod_colour
,prod_bundle_size
,shelf_life_days
,unit_alloc_cost
,unit_abc_cost
,list_price_per_unit
,unit_of_measure_code
,unit_of_measure_sdesc
,discount_allowed
,discount_rate_1
,discount_rate_2
,discount_rate_3
,discount_rate_4
,discount_rate_5
,discount_sdesc_1
,discount_ldesc_1
,discount_sdesc_2
,discount_ldesc_2
,discount_sdesc_3
,discount_ldesc_3
,discount_sdesc_4
,discount_ldesc_4
,discount_sdesc_5
,discount_ldesc_5
,returnable_flag
,return_inspection_req_ldesc
,sellable_flag
,component_flag
,requires_shipping_flag
,taxable_flag
,tax_type_1_sdesc
,tax_type_2_sdesc
,row_del_frm_src_ind
,table_number
,batch_number
,file_cycle_number
,view_number
,audit_timestamp_01
)
select
 z01_vm_product_01.z01_vm_product_01_seq
```

```
,'detail'
,z01_vm_product_01.z01_vm_product_01_key
,z01_vm_product_01.z01_vm_product_01_seq
,1
,ctl_batch_control.batch_date
,ctl_month_control.high_value_date
,z01_vm_product_01.product_number
,z01_vm_product_01.sku_number
,z01_vm_product_01.barcode_num
,z01_vm_product_01.prod_name
,z01_vm_product_01.prod_sub_group
,z01_vm_product_01.prod_group
,z01_vm_product_01.prod_category
,z01_vm_product_01.prod_class
,z01_vm_product_01.prod_line
,z01_vm_product_01.prod_size
,z01_vm_product_01.prod_colour
,z01_vm_product_01.prod_bundle_size
,z01_vm_product_01.shelf_life_days
,z01_vm_product_01.unit_alloc_cost
,z01_vm_product_01.unit_abc_cost
,z01_vm_product_01.list_price_per_unit
,z01_vm_product_01.unit_of_measure_code
,z01_vm_product_01.unit_of_measure_sdesc
,z01_vm_product_01.discount_allowed
,z01_vm_product_01.discount_rate_1
,z01_vm_product_01.discount_rate_2
,z01_vm_product_01.discount_rate_3
,z01_vm_product_01.discount_rate_4
,z01_vm_product_01.discount_rate_5
,z01_vm_product_01.discount_sdesc_1
,z01_vm_product_01.discount_ldesc_1
,z01_vm_product_01.discount_sdesc_2
,z01_vm_product_01.discount_ldesc_2
,z01_vm_product_01.discount_sdesc_3
,z01_vm_product_01.discount_ldesc_3
,z01_vm_product_01.discount_sdesc_4
,z01_vm_product_01.discount_ldesc_4
,z01_vm_product_01.discount_sdesc_5
,z01_vm_product_01.discount_ldesc_5
,z01_vm_product_01.returnable_flag
,z01_vm_product_01.return_inspection_req_ldesc
```

```
,z01_vm_product_01.sellable_flag
,z01_vm_product_01.component_flag
,z01_vm_product_01.requires_shipping_flag
,z01_vm_product_01.taxable_flag
,z01_vm_product_01.tax_type_1_sdesc
,z01_vm_product_01.tax_type_2_sdesc
,z01_vm_product_01.row_del_frm_src_ind
,z01_vm_product_01.table_number
,z01_vm_product_01.batch_number
,z01_vm_product_01.file_cycle_number
,z01_vm_product_01.view_number
,z01_vm_product_01.audit_timestamp_01

from bi4all_01.admin.z01_vm_product_01 z01_vm_product_01
left outer join bi4all_02.admin.ctl_batch_control ctl_batch_control on 1=1
and ctl_batch_control.batch_complete_flag = 0

left outer join bi4all_02.admin.ctl_month_control ctl_month_control on 1=1
limit 100
;
```

Note the joins to the batch control table and the month control table to get the batch date and the high date for the date_from and date_to fields on the type 2 dimension.

The next step is to close out the rows in the target where rows in the _t1 table indicate that the row has been replaced according to the rules for fields to test for change.

```
-------------------------------------------------------------------------------------------------------------------------------
--                                                                                                                           --
--   spreadsheet name:        bi4allmodeldemoworksheet01.xml spreadsheet version:         v3.1.00                            --
--   spreadsheet file name:   c:\seetl\designtime\3.1.00\sourcecode\seetl\bi4allmodeldemoworksheet01v3.1.00.xml              --
--   file name:               c:\temp\dmltype4\dim2p_z01_vm_product_01_dim2s_02.sql                                           --
--   windows domain:          emsbidev01                                                                                      --
--   windows user:            administrator                                                                                   --
--   windows workstation:     emsbidev01                                                                                      --
--   seetl current directory: c:\seetl\designtime\3.1.00\sourcecode\seetl                                                     --
--   windows version:         microsoft windows nt 5.2.3790 service pack 2                                                    --
--   dot net version:         2.0.50727.1433                                                                                  --
--   seetl workbench version: SeETL 3.1.00 - 2010-06-11                                                                       --
--   timestamp:               25/07/2010 09:52:55                                                                             --
--   description:             description                                                                                     --
--                                                                                                                           --
-------------------------------------------------------------------------------------------------------------------------------

update bi4all_02.admin.tm_product set
 current_flag                                                                    = 0
,date_to                                                                         = ctl_batch_control.batch_date - 1

from bi4all_01.admin.z01_vm_product_01_t1 z01_vm_product_01_t1
left outer join bi4all_02.admin.ctl_batch_control ctl_batch_control on 1=1
and ctl_batch_control.batch_complete_flag = 0

where 1=1
and tm_product.level_col                                                         = z01_vm_product_01_t1.level_col
and tm_product.dim_char_ky_fld                                                   = z01_vm_product_01_t1.dim_char_ky_fld
and tm_product.current_flag                                                      = 1

and
(
   not ( z01_vm_product_01_t1.sku_number = tm_product.sku_number or ( z01_vm_product_01_t1.sku_number is null and tm_product.sku_number
is null ))
or not ( z01_vm_product_01_t1.barcode_num = tm_product.barcode_num or ( z01_vm_product_01_t1.barcode_num is null and
tm_product.barcode_num is null ))
or not ( z01_vm_product_01_t1.prod_name = tm_product.prod_name or ( z01_vm_product_01_t1.prod_name is null and tm_product.prod_name is
null ))
or not ( z01_vm_product_01_t1.prod_sub_group = tm_product.prod_sub_group or ( z01_vm_product_01_t1.prod_sub_group is null and
tm_product.prod_sub_group is null ))
)
;
```

Not the test of the fields for change. In sql the test must be for the fields being equal or both being null to get the null equals null condition handled properly

The next step is to update the target table where the field to be tested for change show that there is no change.

```
-------------------------------------------------------------------------------------------------------------------------------
--                                                                                                                           --
--   spreadsheet name:        bi4allmodeldemoworksheet01.xml spreadsheet version:         v3.1.00                          --
--   spreadsheet file name:   c:\seetl\designtime\3.1.00\sourcecode\seetl\bi4allmodeldemoworksheet01v3.1.00.xml            --
--   file name:               c:\temp\dmltype4\dim2p_z01_vm_product_01_dim2s_03.sql                                        --
--   windows domain:          emsbidev01                                                                                  --
--   windows user:            administrator                                                                               --
--   windows workstation:     emsbidev01                                                                                  --
--   seetl current directory: c:\seetl\designtime\3.1.00\sourcecode\seetl                                                  --
--   windows version:         microsoft windows nt 5.2.3790 service pack 2                                                --
--   dot net version:         2.0.50727.1433                                                                              --
--   seetl workbench version: SeETL 3.1.00 - 2010-06-11                                                                   --
--   timestamp:               25/07/2010 09:52:55                                                                         --
--   description:             description                                                                                 --
--                                                                                                                           --
-------------------------------------------------------------------------------------------------------------------------------

update bi4all_02.admin.tm_product set
 product_number                                                  = z01_vm_product_01_t1.product_number
,sku_number                                                      = z01_vm_product_01_t1.sku_number
,barcode_num                                                     = z01_vm_product_01_t1.barcode_num
,prod_name                                                       = z01_vm_product_01_t1.prod_name
,prod_sub_group                                                  = z01_vm_product_01_t1.prod_sub_group
,prod_group                                                      = z01_vm_product_01_t1.prod_group
,prod_category                                                   = z01_vm_product_01_t1.prod_category
,prod_class                                                      = z01_vm_product_01_t1.prod_class
,prod_line                                                       = z01_vm_product_01_t1.prod_line
,prod_size                                                       = z01_vm_product_01_t1.prod_size
,prod_colour                                                     = z01_vm_product_01_t1.prod_colour
,prod_bundle_size                                                = z01_vm_product_01_t1.prod_bundle_size
,shelf_life_days                                                 = z01_vm_product_01_t1.shelf_life_days
,unit_alloc_cost                                                 = z01_vm_product_01_t1.unit_alloc_cost
,unit_abc_cost                                                   = z01_vm_product_01_t1.unit_abc_cost
,list_price_per_unit                                             = z01_vm_product_01_t1.list_price_per_unit
,unit_of_measure_code                                            = z01_vm_product_01_t1.unit_of_measure_code
,unit_of_measure_sdesc                                           = z01_vm_product_01_t1.unit_of_measure_sdesc
,discount_allowed                                                = z01_vm_product_01_t1.discount_allowed
,discount_rate_1                                                 = z01_vm_product_01_t1.discount_rate_1
,discount_rate_2                                                 = z01_vm_product_01_t1.discount_rate_2
,discount_rate_3                                                 = z01_vm_product_01_t1.discount_rate_3
,discount_rate_4                                                 = z01_vm_product_01_t1.discount_rate_4
,discount_rate_5                                                 = z01_vm_product_01_t1.discount_rate_5
,discount_sdesc_1                                                = z01_vm_product_01_t1.discount_sdesc_1
,discount_ldesc_1                                                = z01_vm_product_01_t1.discount_ldesc_1
```

```
,discount_sdesc_2                                                    = z01_vm_product_01_t1.discount_sdesc_2
,discount_ldesc_2                                                    = z01_vm_product_01_t1.discount_ldesc_2
,discount_sdesc_3                                                    = z01_vm_product_01_t1.discount_sdesc_3
,discount_ldesc_3                                                    = z01_vm_product_01_t1.discount_ldesc_3
,discount_sdesc_4                                                    = z01_vm_product_01_t1.discount_sdesc_4
,discount_ldesc_4                                                    = z01_vm_product_01_t1.discount_ldesc_4
,discount_sdesc_5                                                    = z01_vm_product_01_t1.discount_sdesc_5
,discount_ldesc_5                                                    = z01_vm_product_01_t1.discount_ldesc_5
,returnable_flag                                                     = z01_vm_product_01_t1.returnable_flag
,return_inspection_req_ldesc                                         = z01_vm_product_01_t1.return_inspection_req_ldesc
,sellable_flag                                                       = z01_vm_product_01_t1.sellable_flag
,component_flag                                                      = z01_vm_product_01_t1.component_flag
,requires_shipping_flag                                              = z01_vm_product_01_t1.requires_shipping_flag
,taxable_flag                                                        = z01_vm_product_01_t1.taxable_flag
,tax_type_1_sdesc                                                    = z01_vm_product_01_t1.tax_type_1_sdesc
,tax_type_2_sdesc                                                    = z01_vm_product_01_t1.tax_type_2_sdesc
,row_del_frm_src_ind                                                 = z01_vm_product_01_t1.row_del_frm_src_ind
,table_number                                                        = z01_vm_product_01_t1.table_number
,batch_number                                                        = z01_vm_product_01_t1.batch_number
,file_cycle_number                                                   = z01_vm_product_01_t1.file_cycle_number
,view_number                                                         = z01_vm_product_01_t1.view_number
,audit_timestamp_01                                                  = z01_vm_product_01_t1.audit_timestamp_01

from bi4all_01.admin.z01_vm_product_01_t1 z01_vm_product_01_t1
left outer join bi4all_02.admin.ctl_batch_control ctl_batch_control on 1=1
and ctl_batch_control.batch_complete_flag = 0

where 1=1
and tm_product.level_col                                             = z01_vm_product_01_t1.level_col
and tm_product.dim_char_ky_fld                                       = z01_vm_product_01_t1.dim_char_ky_fld
and tm_product.current_flag                                          = 1

and ( z01_vm_product_01_t1.sku_number = tm_product.sku_number or ( z01_vm_product_01_t1.sku_number is null and tm_product.sku_number is
null ))
and ( z01_vm_product_01_t1.barcode_num = tm_product.barcode_num or ( z01_vm_product_01_t1.barcode_num is null and tm_product.barcode_num
is null ))
and ( z01_vm_product_01_t1.prod_name = tm_product.prod_name or ( z01_vm_product_01_t1.prod_name is null and tm_product.prod_name is null
))
and ( z01_vm_product_01_t1.prod_sub_group = tm_product.prod_sub_group or ( z01_vm_product_01_t1.prod_sub_group is null and
tm_product.prod_sub_group is null ))
;
```

The next step is to insert the new rows into the target table where it has been found that the rows need to be inserted.

```
-------------------------------------------------------------------------------------------------------------------------
--                                                                                                                     --
--   spreadsheet name:        bi4allmodeldemoworksheet01.xml spreadsheet version:        v3.1.00                        --
--   spreadsheet file name:   c:\seetl\designtime\3.1.00\sourcecode\seetl\bi4allmodeldemoworksheet01v3.1.00.xml         --
--   file name:               c:\temp\dmltype4\dim2p_z01_vm_product_01_dim2s_04.sql                                     --
--   windows domain:          emsbidev01                                                                                --
--   windows user:            administrator                                                                             --
--   windows workstation:     emsbidev01                                                                                --
--   seetl current directory: c:\seetl\designtime\3.1.00\sourcecode\seetl                                               --
--   windows version:         microsoft windows nt 5.2.3790 service pack 2                                              --
--   dot net version:         2.0.50727.1433                                                                            --
--   seetl workbench version: SeETL 3.1.00 - 2010-06-11                                                                 --
--   timestamp:               25/07/2010 09:52:55                                                                       --
--   description:             description                                                                               --
--                                                                                                                     --
-------------------------------------------------------------------------------------------------------------------------

insert into bi4all_02.admin.tm_product
(
 pk_tm_product
,level_col
,dim_char_ky_fld
,current_flag
,date_from
,date_to
,tm_product_seq
,product_number
,sku_number
,barcode_num
,prod_name
,prod_sub_group
,prod_group
,prod_category
,prod_class
,prod_line
,prod_size
,prod_colour
,prod_bundle_size
,shelf_life_days
```

```
,unit_alloc_cost
,unit_abc_cost
,list_price_per_unit
,unit_of_measure_code
,unit_of_measure_sdesc
,discount_allowed
,discount_rate_1
,discount_rate_2
,discount_rate_3
,discount_rate_4
,discount_rate_5
,discount_sdesc_1
,discount_ldesc_1
,discount_sdesc_2
,discount_ldesc_2
,discount_sdesc_3
,discount_ldesc_3
,discount_sdesc_4
,discount_ldesc_4
,discount_sdesc_5
,discount_ldesc_5
,returnable_flag
,return_inspection_req_ldesc
,sellable_flag
,component_flag
,requires_shipping_flag
,taxable_flag
,tax_type_1_sdesc
,tax_type_2_sdesc
,row_del_frm_src_ind
,table_number
,batch_number
,file_cycle_number
,view_number
,audit_timestamp_01
)
select
 z01_vm_product_01_t1.pk_vm_product
,z01_vm_product_01_t1.level_col
,z01_vm_product_01_t1.dim_char_ky_fld
,1
,z01_vm_product_01_t1.date_from
```

```
,z01_vm_product_01_t1.date_to
,z01_vm_product_01_t1.vm_product_seq
,z01_vm_product_01_t1.product_number
,z01_vm_product_01_t1.sku_number
,z01_vm_product_01_t1.barcode_num
,z01_vm_product_01_t1.prod_name
,z01_vm_product_01_t1.prod_sub_group
,z01_vm_product_01_t1.prod_group
,z01_vm_product_01_t1.prod_category
,z01_vm_product_01_t1.prod_class
,z01_vm_product_01_t1.prod_line
,z01_vm_product_01_t1.prod_size
,z01_vm_product_01_t1.prod_colour
,z01_vm_product_01_t1.prod_bundle_size
,z01_vm_product_01_t1.shelf_life_days
,z01_vm_product_01_t1.unit_alloc_cost
,z01_vm_product_01_t1.unit_abc_cost
,z01_vm_product_01_t1.list_price_per_unit
,z01_vm_product_01_t1.unit_of_measure_code
,z01_vm_product_01_t1.unit_of_measure_sdesc
,z01_vm_product_01_t1.discount_allowed
,z01_vm_product_01_t1.discount_rate_1
,z01_vm_product_01_t1.discount_rate_2
,z01_vm_product_01_t1.discount_rate_3
,z01_vm_product_01_t1.discount_rate_4
,z01_vm_product_01_t1.discount_rate_5
,z01_vm_product_01_t1.discount_sdesc_1
,z01_vm_product_01_t1.discount_ldesc_1
,z01_vm_product_01_t1.discount_sdesc_2
,z01_vm_product_01_t1.discount_ldesc_2
,z01_vm_product_01_t1.discount_sdesc_3
,z01_vm_product_01_t1.discount_ldesc_3
,z01_vm_product_01_t1.discount_sdesc_4
,z01_vm_product_01_t1.discount_ldesc_4
,z01_vm_product_01_t1.discount_sdesc_5
,z01_vm_product_01_t1.discount_ldesc_5
,z01_vm_product_01_t1.returnable_flag
,z01_vm_product_01_t1.return_inspection_req_ldesc
,z01_vm_product_01_t1.sellable_flag
,z01_vm_product_01_t1.component_flag
,z01_vm_product_01_t1.requires_shipping_flag
,z01_vm_product_01_t1.taxable_flag
```

```
,z01_vm_product_01_t1.tax_type_1_sdesc
,z01_vm_product_01_t1.tax_type_2_sdesc
,z01_vm_product_01_t1.row_del_frm_src_ind
,z01_vm_product_01_t1.table_number
,z01_vm_product_01_t1.batch_number
,z01_vm_product_01_t1.file_cycle_number
,z01_vm_product_01_t1.view_number
,z01_vm_product_01_t1.audit_timestamp_01

from bi4all_01.admin.z01_vm_product_01_t1 z01_vm_product_01_t1

left join bi4all_02.admin.tm_product tm_product on 1=1
and tm_product.level_col                                    = z01_vm_product_01_t1.level_col
and tm_product.dim_char_ky_fld                              = z01_vm_product_01_t1.dim_char_ky_fld
and tm_product.current_flag                                 = 1

where 1=1
and tm_product.level_col                                    is null
and tm_product.dim_char_ky_fld                              is null
and tm_product.current_flag                                 is null
;
```

Note there is no need to test for fields that are sensitive to change. If there is no ‘current row’ as represented by the current_flag being set to 1 then the row will need to be inserted.

7.1.1.4. Fact table processing without a key on the fact table.

This is the first example of the processing required for fact table processing. It is best at this time to point out some features of this sql that is generated.

```
-------------------------------------------------------------------------------------------------------------------------
--                                                                                                                     --
--   spreadsheet name:        bi4allmodeldemoworksheet01.xml spreadsheet version:        v3.1.00                       --
--   spreadsheet file name:   c:\seetl\designtime\3.1.00\sourcecode\seetl\bi4allmodeldemoworksheet01v3.1.00.xml         --
--   file name:               c:\temp\dmltype4\factp_z01_vf_sale_txn_01_facts_01.sql                                    --
--   windows domain:          emsbidev01                                                                                --
--   windows user:            administrator                                                                             --
--   windows workstation:     emsbidev01                                                                                --
--   seetl current directory: c:\seetl\designtime\3.1.00\sourcecode\seetl                                               --
--   windows version:         microsoft windows nt 5.2.3790 service pack 2                                              --
--   dot net version:         2.0.50727.1433                                                                            --
--   seetl workbench version: SeETL 3.1.00 - 2010-06-11                                                                 --
--   timestamp:               25/07/2010 10:32:49                                                                       --
--   description:             description                                                                               --
--                                                                                                                     --
-------------------------------------------------------------------------------------------------------------------------

truncate table bi4all_02.admin.z01_vf_sale_txn_01_t1
;

insert into bi4all_02.admin.z01_vf_sale_txn_01_t1
(
 dk_vm_sale_date
,dk_vm_sale_minute
,dk_vm_sub_campaign
,dk_vm_product
,dk_vm_customer
,dk_vf_customer_dims_asoc
,dk_vm_account
,dk_vf_account_dims_asoc
,dk_vm_customer_demographic
,dk_vm_customer_geography
,dk_vm_geo_code
,dk_vm_map_reference
,dk_vm_response_type
,dk_vm_currency
```

```
,dk_vm_sales_rep
,dk_vm_unit_of_measure
,dk_vm_txn_type
,dk_vm_card
,dk_vm_card_type
,dk_vm_comp_reporting_struct
,dk_vm_account_status
,dk_vm_account_type
,dk_vf_campaign_dims_asoc
,dk_vf_solicitation_txn
,pk_vf_sale_txn                    ← Note the creation of the primary key.
,product_number
,customer_number
,card_type
,card_number
,account_number
,account_type
,account_status
,currency_str
,postcode_str
,address_key_str
,transaction_type
,sales_rep_number
,sale_unit_amount
,sale_extended_amount
,cost_unit_amount
,cost_extended_amount
,tax1_unit_amount
,tax1_extended_amount
,tax2_unit_amount
,tax2_extended_amount
,sale_units
,appsys_reference_number_str
,sale_sdesc
,sale_ldesc
,sale_tstamp
,number_sales
,sale_txn_returned_flag
,return_period_expired_flag
,return_probability_pct
,partitioning_column
,partitioning_column_part_01
```

```
,partitioning_column_part_02
,row_del_frm_src_ind
,table_number                          ← Note the table number included.
,batch_number                          ← Note the batch number included.
,file_cycle_number
,view_number
,audit_timestamp_01
)
select
 nvl(dk_zvm_sale_date, 0 )             ← Note the test for a null value that places the value 0 in the output if the lookup failed. This is done for all dims.
,nvl(dk_zvm_sale_minute, 0 )
,nvl(dk_zvm_sub_campaign, 0 )
,nvl(dk_zvm_product, 0 )
,nvl(dk_zvm_customer, 0 )
,nvl(dk_zvf_customer_dims_asoc, 0 )
,nvl(dk_zvm_account, 0 )
,nvl(dk_zvf_account_dims_asoc, 0 )
,nvl(dk_zvm_customer_demographic, 0 )
,nvl(dk_zvm_customer_geography, 0 )
,nvl(dk_zvm_geo_code, 0 )
,nvl(dk_zvm_map_reference, 0 )
,nvl(dk_zvm_response_type, 0 )
,nvl(dk_zvm_currency, 0 )
,nvl(dk_zvm_sales_rep, 0 )
,nvl(dk_zvm_unit_of_measure, 0 )
,nvl(dk_zvm_txn_type, 0 )
,nvl(dk_zvm_card, 0 )
,nvl(dk_zvm_card_type, 0 )
,nvl(dk_zvm_comp_reporting_struct, 0 )
,nvl(dk_zvm_account_status, 0 )
,nvl(dk_zvm_account_type, 0 )
,nvl(dk_zvf_campaign_dims_asoc, 0 )
,nvl(dk_zvf_solicitation_txn, 0 )
,ctl_sequence_number                   ← Note the ctl sequence number which is detected and identified.
,product_number
,customer_number
,card_type
,card_number
,account_number
,account_type
,account_status
,currency_str
```

```
,postcode_str
,address_key_str
,transaction_type
,sales_rep_number
,sale_unit_amount
,sale_extended_amount
,cost_unit_amount
,cost_extended_amount
,tax1_unit_amount
,tax1_extended_amount
,tax2_unit_amount
,tax2_extended_amount
,sale_units
,appsys_reference_number_str
,sale_sdesc
,sale_ldesc
,sale_tstamp
,number_sales
,sale_txn_returned_flag
,return_period_expired_flag
,return_probability_pct
,partitioning_column
,partitioning_column_part_01
,partitioning_column_part_02
,row_del_frm_src_ind
,table_number
,batch_number
,file_cycle_number
,view_number
,audit_timestamp_01
from bi4all_01.admin.z01_vf_sale_txn_01 z01_vf_sale_txn_01
```

This is the important piece. Note that the left joins join the string passed in from the input view to the _t3 tables which are the dim char key fields and integer keys from the dimension tables loaded into the _t3 tables specifically for the join. These joins are all generated out of the spreadsheet based on the spreadsheets knowledge of the joins to be coded. These joins replace 'lookups' in tools like informatica and datastage. It is obviously an order of magnitude, if not more, easier to create these joins like this than to code them through a GUI as provided by informatica or datastage.

```
left join bi4all_02.admin.zvm_sale_date_at3   zvm_sale_date_at3
on z01_vf_sale_txn_01.char_key_zvm_sale_date = zvm_sale_date_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_sale_minute_at3   zvm_sale_minute_at3
```

```
on z01_vf_sale_txn_01.char_key_zvm_sale_minute = zvm_sale_minute_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_sub_campaign_at3   zvm_sub_campaign_at3
on z01_vf_sale_txn_01.char_key_zvm_sub_campaign = zvm_sub_campaign_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_product_at3   zvm_product_at3
on z01_vf_sale_txn_01.char_key_zvm_product = zvm_product_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_customer_at3   zvm_customer_at3
on z01_vf_sale_txn_01.char_key_zvm_customer = zvm_customer_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvf_customer_dims_asoc_at3   zvf_customer_dims_asoc_at3
on z01_vf_sale_txn_01.char_key_zvf_customer_dims_asoc = zvf_customer_dims_asoc_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_account_at3   zvm_account_at3
on z01_vf_sale_txn_01.char_key_zvm_account = zvm_account_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvf_account_dims_asoc_at3   zvf_account_dims_asoc_at3
on z01_vf_sale_txn_01.char_key_zvf_account_dims_asoc = zvf_account_dims_asoc_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_customer_demographic_at3   zvm_customer_demographic_at3
on z01_vf_sale_txn_01.char_key_zvm_customer_demographic = zvm_customer_demographic_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_customer_geography_at3   zvm_customer_geography_at3
on z01_vf_sale_txn_01.char_key_zvm_customer_geography = zvm_customer_geography_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_geo_code_at3   zvm_geo_code_at3
on z01_vf_sale_txn_01.char_key_zvm_geo_code = zvm_geo_code_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_map_reference_at3   zvm_map_reference_at3
on z01_vf_sale_txn_01.char_key_zvm_map_reference = zvm_map_reference_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_response_type_at3   zvm_response_type_at3
on z01_vf_sale_txn_01.char_key_zvm_response_type = zvm_response_type_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_currency_at3   zvm_currency_at3
on z01_vf_sale_txn_01.char_key_zvm_currency = zvm_currency_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_sales_rep_at3   zvm_sales_rep_at3
on z01_vf_sale_txn_01.char_key_zvm_sales_rep = zvm_sales_rep_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_unit_of_measure_at3   zvm_unit_of_measure_at3
```

```
on z01_vf_sale_txn_01.char_key_zvm_unit_of_measure = zvm_unit_of_measure_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_txn_type_at3   zvm_txn_type_at3
on z01_vf_sale_txn_01.char_key_zvm_txn_type = zvm_txn_type_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_card_at3   zvm_card_at3
on z01_vf_sale_txn_01.char_key_zvm_card = zvm_card_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_card_type_at3   zvm_card_type_at3
on z01_vf_sale_txn_01.char_key_zvm_card_type = zvm_card_type_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_comp_reporting_struct_at3   zvm_comp_reporting_struct_at3
on z01_vf_sale_txn_01.char_key_zvm_comp_reporting_struct = zvm_comp_reporting_struct_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_account_status_at3   zvm_account_status_at3
on z01_vf_sale_txn_01.char_key_zvm_account_status = zvm_account_status_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_account_type_at3   zvm_account_type_at3
on z01_vf_sale_txn_01.char_key_zvm_account_type = zvm_account_type_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvf_campaign_dims_asoc_at3   zvf_campaign_dims_asoc_at3
on z01_vf_sale_txn_01.char_key_zvf_campaign_dims_asoc = zvf_campaign_dims_asoc_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvf_solicitation_txn_at3   zvf_solicitation_txn_at3
on z01_vf_sale_txn_01.char_key_zvf_solicitation_txn = zvf_solicitation_txn_at3.pk_dim_char_ky_fld
limit 200
;
```

In this example there are no keys defined on the fact table and it is assumed that all rows being presented are new. If a fact table is to have rows updated then some fields on the row are required to identify the unique key of the row.

```
-------------------------------------------------------------------------------------------------------------------------
--                                                                                                                     --
--   spreadsheet name:        bi4allmodeldemoworksheet01.xml spreadsheet version:         v3.1.00                      --
--   spreadsheet file name:   c:\seetl\designtime\3.1.00\sourcecode\seetl\bi4allmodeldemoworksheet01v3.1.00.xml        --
--   file name:               c:\temp\dmltype4\factp_z01_vf_sale_txn_01_facts_02.sql                                    --
--   windows domain:          emsbidev01                                                                                --
--   windows user:            administrator                                                                             --
--   windows workstation:     emsbidev01                                                                                --
--   seetl current directory: c:\seetl\designtime\3.1.00\sourcecode\seetl                                               --
--   windows version:         microsoft windows nt 5.2.3790 service pack 2                                              --
--   dot net version:         2.0.50727.1433                                                                            --
--   seetl workbench version: SeETL 3.1.00 - 2010-06-11                                                                 --
--   timestamp:               25/07/2010 10:32:49                                                                       --
--   description:             description                                                                               --
--                                                                                                                     --
-------------------------------------------------------------------------------------------------------------------------

insert into bi4all_02.admin.tf_sale_txn
(
 dk_vm_sale_date
,dk_vm_sale_minute
,dk_vm_sub_campaign
,dk_vm_product
,dk_vm_customer
,dk_vf_customer_dims_asoc
,dk_vm_account
,dk_vf_account_dims_asoc
,dk_vm_customer_demographic
,dk_vm_customer_geography
,dk_vm_geo_code
,dk_vm_map_reference
,dk_vm_response_type
,dk_vm_currency
,dk_vm_sales_rep
,dk_vm_unit_of_measure
,dk_vm_txn_type
,dk_vm_card
,dk_vm_card_type
```

```
,dk_vm_comp_reporting_struct
,dk_vm_account_status
,dk_vm_account_type
,dk_vf_campaign_dims_asoc
,dk_vf_solicitation_txn
,pk_tf_sale_txn
,product_number
,customer_number
,card_type
,card_number
,account_number
,account_type
,account_status
,currency_str
,postcode_str
,address_key_str
,transaction_type
,sales_rep_number
,sale_unit_amount
,sale_extended_amount
,cost_unit_amount
,cost_extended_amount
,tax1_unit_amount
,tax1_extended_amount
,tax2_unit_amount
,tax2_extended_amount
,sale_units
,appsys_reference_number_str
,sale_sdesc
,sale_ldesc
,sale_tstamp
,number_sales
,sale_txn_returned_flag
,return_period_expired_flag
,return_probability_pct
,partitioning_column
,partitioning_column_part_01
,partitioning_column_part_02
,row_del_frm_src_ind
,table_number
,batch_number
,file_cycle_number
```

```
,view_number
,audit_timestamp_01
)
select
 z01_vf_sale_txn_01_t1.dk_vm_sale_date
,z01_vf_sale_txn_01_t1.dk_vm_sale_minute
,z01_vf_sale_txn_01_t1.dk_vm_sub_campaign
,z01_vf_sale_txn_01_t1.dk_vm_product
,z01_vf_sale_txn_01_t1.dk_vm_customer
,z01_vf_sale_txn_01_t1.dk_vf_customer_dims_asoc
,z01_vf_sale_txn_01_t1.dk_vm_account
,z01_vf_sale_txn_01_t1.dk_vf_account_dims_asoc
,z01_vf_sale_txn_01_t1.dk_vm_customer_demographic
,z01_vf_sale_txn_01_t1.dk_vm_customer_geography
,z01_vf_sale_txn_01_t1.dk_vm_geo_code
,z01_vf_sale_txn_01_t1.dk_vm_map_reference
,z01_vf_sale_txn_01_t1.dk_vm_response_type
,z01_vf_sale_txn_01_t1.dk_vm_currency
,z01_vf_sale_txn_01_t1.dk_vm_sales_rep
,z01_vf_sale_txn_01_t1.dk_vm_unit_of_measure
,z01_vf_sale_txn_01_t1.dk_vm_txn_type
,z01_vf_sale_txn_01_t1.dk_vm_card
,z01_vf_sale_txn_01_t1.dk_vm_card_type
,z01_vf_sale_txn_01_t1.dk_vm_comp_reporting_struct
,z01_vf_sale_txn_01_t1.dk_vm_account_status
,z01_vf_sale_txn_01_t1.dk_vm_account_type
,z01_vf_sale_txn_01_t1.dk_vf_campaign_dims_asoc
,z01_vf_sale_txn_01_t1.dk_vf_solicitation_txn
,z01_vf_sale_txn_01_t1.pk_vf_sale_txn
,z01_vf_sale_txn_01_t1.product_number
,z01_vf_sale_txn_01_t1.customer_number
,z01_vf_sale_txn_01_t1.card_type
,z01_vf_sale_txn_01_t1.card_number
,z01_vf_sale_txn_01_t1.account_number
,z01_vf_sale_txn_01_t1.account_type
,z01_vf_sale_txn_01_t1.account_status
,z01_vf_sale_txn_01_t1.currency_str
,z01_vf_sale_txn_01_t1.postcode_str
,z01_vf_sale_txn_01_t1.address_key_str
,z01_vf_sale_txn_01_t1.transaction_type
,z01_vf_sale_txn_01_t1.sales_rep_number
,z01_vf_sale_txn_01_t1.sale_unit_amount
```

```
,z01_vf_sale_txn_01_t1.sale_extended_amount
,z01_vf_sale_txn_01_t1.cost_unit_amount
,z01_vf_sale_txn_01_t1.cost_extended_amount
,z01_vf_sale_txn_01_t1.tax1_unit_amount
,z01_vf_sale_txn_01_t1.tax1_extended_amount
,z01_vf_sale_txn_01_t1.tax2_unit_amount
,z01_vf_sale_txn_01_t1.tax2_extended_amount
,z01_vf_sale_txn_01_t1.sale_units
,z01_vf_sale_txn_01_t1.appsys_reference_number_str
,z01_vf_sale_txn_01_t1.sale_sdesc
,z01_vf_sale_txn_01_t1.sale_ldesc
,z01_vf_sale_txn_01_t1.sale_tstamp
,z01_vf_sale_txn_01_t1.number_sales
,z01_vf_sale_txn_01_t1.sale_txn_returned_flag
,z01_vf_sale_txn_01_t1.return_period_expired_flag
,z01_vf_sale_txn_01_t1.return_probability_pct
,z01_vf_sale_txn_01_t1.partitioning_column
,z01_vf_sale_txn_01_t1.partitioning_column_part_01
,z01_vf_sale_txn_01_t1.partitioning_column_part_02
,z01_vf_sale_txn_01_t1.row_del_frm_src_ind
,z01_vf_sale_txn_01_t1.table_number
,z01_vf_sale_txn_01_t1.batch_number
,z01_vf_sale_txn_01_t1.file_cycle_number
,z01_vf_sale_txn_01_t1.view_number
,z01_vf_sale_txn_01_t1.audit_timestamp_01
from bi4all_02.admin.z01_vf_sale_txn_01_t1 z01_vf_sale_txn_01_t1
;
```

7.1.1.5. Fact table processing with a key on the fact table.

This is again the first step and it is identical to when the fact table does not have a key defined as above. It is included here for completeness.

```
-------------------------------------------------------------------------------------------------------------------------
--                                                                                                                     --
--   spreadsheet name:        bi4allmodeldemoworksheet01.xml spreadsheet version:        v3.1.00                       --
--   spreadsheet file name:   c:\seetl\designtime\3.1.00\sourcecode\seetl\bi4allmodeldemoworksheet01v3.1.00.xml         --
--   file name:               c:\temp\dmltype4\factp_z01_vf_sale_txn_01_facts_01.sql                                    --
--   windows domain:          emsbidev01                                                                                --
--   windows user:            administrator                                                                             --
--   windows workstation:     emsbidev01                                                                                --
--   seetl current directory: c:\seetl\designtime\3.1.00\sourcecode\seetl                                               --
--   windows version:         microsoft windows nt 5.2.3790 service pack 2                                              --
--   dot net version:         2.0.50727.1433                                                                            --
--   seetl workbench version: SeETL 3.1.00 - 2010-06-11                                                                 --
--   timestamp:               25/07/2010 10:25:24                                                                       --
--   description:             description                                                                               --
--                                                                                                                     --
-------------------------------------------------------------------------------------------------------------------------

truncate table bi4all_02.admin.z01_vf_sale_txn_01_t1
;

insert into bi4all_02.admin.z01_vf_sale_txn_01_t1
(
 dk_vm_sale_date
,dk_vm_sale_minute
,dk_vm_sub_campaign
,dk_vm_product
,dk_vm_customer
,dk_vf_customer_dims_asoc
,dk_vm_account
,dk_vf_account_dims_asoc
,dk_vm_customer_demographic
,dk_vm_customer_geography
,dk_vm_geo_code
,dk_vm_map_reference
,dk_vm_response_type
,dk_vm_currency
```

```
,dk_vm_sales_rep
,dk_vm_unit_of_measure
,dk_vm_txn_type
,dk_vm_card
,dk_vm_card_type
,dk_vm_comp_reporting_struct
,dk_vm_account_status
,dk_vm_account_type
,dk_vf_campaign_dims_asoc
,dk_vf_solicitation_txn
,pk_vf_sale_txn
,product_number
,customer_number
,card_type
,card_number
,account_number
,account_type
,account_status
,currency_str
,postcode_str
,address_key_str
,transaction_type
,sales_rep_number
,sale_unit_amount
,sale_extended_amount
,cost_unit_amount
,cost_extended_amount
,tax1_unit_amount
,tax1_extended_amount
,tax2_unit_amount
,tax2_extended_amount
,sale_units
,appsys_reference_number_str
,sale_sdesc
,sale_ldesc
,sale_tstamp
,number_sales
,sale_txn_returned_flag
,return_period_expired_flag
,return_probability_pct
,partitioning_column
,partitioning_column_part_01
```

```
,partitioning_column_part_02
,row_del_frm_src_ind
,table_number
,batch_number
,file_cycle_number
,view_number
,audit_timestamp_01
)
select
 nvl(dk_zvm_sale_date, 0 )
,nvl(dk_zvm_sale_minute, 0 )
,nvl(dk_zvm_sub_campaign, 0 )
,nvl(dk_zvm_product, 0 )
,nvl(dk_zvm_customer, 0 )
,nvl(dk_zvf_customer_dims_asoc, 0 )
,nvl(dk_zvm_account, 0 )
,nvl(dk_zvf_account_dims_asoc, 0 )
,nvl(dk_zvm_customer_demographic, 0 )
,nvl(dk_zvm_customer_geography, 0 )
,nvl(dk_zvm_geo_code, 0 )
,nvl(dk_zvm_map_reference, 0 )
,nvl(dk_zvm_response_type, 0 )
,nvl(dk_zvm_currency, 0 )
,nvl(dk_zvm_sales_rep, 0 )
,nvl(dk_zvm_unit_of_measure, 0 )
,nvl(dk_zvm_txn_type, 0 )
,nvl(dk_zvm_card, 0 )
,nvl(dk_zvm_card_type, 0 )
,nvl(dk_zvm_comp_reporting_struct, 0 )
,nvl(dk_zvm_account_status, 0 )
,nvl(dk_zvm_account_type, 0 )
,nvl(dk_zvf_campaign_dims_asoc, 0 )
,nvl(dk_zvf_solicitation_txn, 0 )
,ctl_sequence_number
,product_number
,customer_number
,card_type
,card_number
,account_number
,account_type
,account_status
,currency_str
```

```
,postcode_str
,address_key_str
,transaction_type
,sales_rep_number
,sale_unit_amount
,sale_extended_amount
,cost_unit_amount
,cost_extended_amount
,tax1_unit_amount
,tax1_extended_amount
,tax2_unit_amount
,tax2_extended_amount
,sale_units
,appsys_reference_number_str
,sale_sdesc
,sale_ldesc
,sale_tstamp
,number_sales
,sale_txn_returned_flag
,return_period_expired_flag
,return_probability_pct
,partitioning_column
,partitioning_column_part_01
,partitioning_column_part_02
,row_del_frm_src_ind
,table_number
,batch_number
,file_cycle_number
,view_number
,audit_timestamp_01
from bi4all_01.admin.z01_vf_sale_txn_01 z01_vf_sale_txn_01

left join bi4all_02.admin.zvm_sale_date_at3   zvm_sale_date_at3
on z01_vf_sale_txn_01.char_key_zvm_sale_date = zvm_sale_date_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_sale_minute_at3   zvm_sale_minute_at3
on z01_vf_sale_txn_01.char_key_zvm_sale_minute = zvm_sale_minute_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_sub_campaign_at3   zvm_sub_campaign_at3
on z01_vf_sale_txn_01.char_key_zvm_sub_campaign = zvm_sub_campaign_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_product_at3   zvm_product_at3
```

```
on z01_vf_sale_txn_01.char_key_zvm_product = zvm_product_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_customer_at3   zvm_customer_at3
on z01_vf_sale_txn_01.char_key_zvm_customer = zvm_customer_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvf_customer_dims_asoc_at3   zvf_customer_dims_asoc_at3
on z01_vf_sale_txn_01.char_key_zvf_customer_dims_asoc = zvf_customer_dims_asoc_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_account_at3   zvm_account_at3
on z01_vf_sale_txn_01.char_key_zvm_account = zvm_account_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvf_account_dims_asoc_at3   zvf_account_dims_asoc_at3
on z01_vf_sale_txn_01.char_key_zvf_account_dims_asoc = zvf_account_dims_asoc_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_customer_demographic_at3   zvm_customer_demographic_at3
on z01_vf_sale_txn_01.char_key_zvm_customer_demographic = zvm_customer_demographic_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_customer_geography_at3   zvm_customer_geography_at3
on z01_vf_sale_txn_01.char_key_zvm_customer_geography = zvm_customer_geography_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_geo_code_at3   zvm_geo_code_at3
on z01_vf_sale_txn_01.char_key_zvm_geo_code = zvm_geo_code_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_map_reference_at3   zvm_map_reference_at3
on z01_vf_sale_txn_01.char_key_zvm_map_reference = zvm_map_reference_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_response_type_at3   zvm_response_type_at3
on z01_vf_sale_txn_01.char_key_zvm_response_type = zvm_response_type_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_currency_at3   zvm_currency_at3
on z01_vf_sale_txn_01.char_key_zvm_currency = zvm_currency_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_sales_rep_at3   zvm_sales_rep_at3
on z01_vf_sale_txn_01.char_key_zvm_sales_rep = zvm_sales_rep_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_unit_of_measure_at3   zvm_unit_of_measure_at3
on z01_vf_sale_txn_01.char_key_zvm_unit_of_measure = zvm_unit_of_measure_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_txn_type_at3   zvm_txn_type_at3
on z01_vf_sale_txn_01.char_key_zvm_txn_type = zvm_txn_type_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_card_at3   zvm_card_at3
```

```
on z01_vf_sale_txn_01.char_key_zvm_card = zvm_card_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_card_type_at3   zvm_card_type_at3
on z01_vf_sale_txn_01.char_key_zvm_card_type = zvm_card_type_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_comp_reporting_struct_at3   zvm_comp_reporting_struct_at3
on z01_vf_sale_txn_01.char_key_zvm_comp_reporting_struct = zvm_comp_reporting_struct_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_account_status_at3   zvm_account_status_at3
on z01_vf_sale_txn_01.char_key_zvm_account_status = zvm_account_status_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvm_account_type_at3   zvm_account_type_at3
on z01_vf_sale_txn_01.char_key_zvm_account_type = zvm_account_type_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvf_campaign_dims_asoc_at3   zvf_campaign_dims_asoc_at3
on z01_vf_sale_txn_01.char_key_zvf_campaign_dims_asoc = zvf_campaign_dims_asoc_at3.pk_dim_char_ky_fld

left join bi4all_02.admin.zvf_solicitation_txn_at3   zvf_solicitation_txn_at3
on z01_vf_sale_txn_01.char_key_zvf_solicitation_txn = zvf_solicitation_txn_at3.pk_dim_char_ky_fld
limit 200
;
```

The next step is to apply updates to the target table based on the rows in the _t1 table where the natural keys in the _t1 table are found in the target fact table. Note that the join across to check to see if the row exists in the target table may be a very large join and may require large amounts of processing power.

```
-------------------------------------------------------------------------------------------------------------------------------
--                                                                                                                           --
--   spreadsheet name:        bi4allmodeldemoworksheet01.xml spreadsheet version:         v3.1.00                           --
--   spreadsheet file name:   c:\seetl\designtime\3.1.00\sourcecode\seetl\bi4allmodeldemoworksheet01v3.1.00.xml             --
--   file name:               c:\temp\dmltype4\factp_z01_vf_sale_txn_01_facts_03.sql                                       --
--   windows domain:          emsbidev01                                                                                   --
--   windows user:            administrator                                                                                --
--   windows workstation:     emsbidev01                                                                                   --
--   seetl current directory: c:\seetl\designtime\3.1.00\sourcecode\seetl                                                   --
--   windows version:         microsoft windows nt 5.2.3790 service pack 2                                                 --
--   dot net version:         2.0.50727.1433                                                                               --
--   seetl workbench version: SeETL 3.1.00 - 2010-06-11                                                                    --
--   timestamp:               25/07/2010 10:25:24                                                                          --
--   description:             description                                                                                  --
--                                                                                                                           --
-------------------------------------------------------------------------------------------------------------------------------

update bi4all_02.admin.tf_sale_txn set
 dk_vm_sale_date                                                = z01_vf_sale_txn_01_t1.dk_vm_sale_date
,dk_vm_sale_minute                                              = z01_vf_sale_txn_01_t1.dk_vm_sale_minute
,dk_vm_sub_campaign                                             = z01_vf_sale_txn_01_t1.dk_vm_sub_campaign
,dk_vm_product                                                  = z01_vf_sale_txn_01_t1.dk_vm_product
,dk_vm_customer                                                 = z01_vf_sale_txn_01_t1.dk_vm_customer
,dk_vf_customer_dims_asoc                                       = z01_vf_sale_txn_01_t1.dk_vf_customer_dims_asoc
,dk_vm_account                                                  = z01_vf_sale_txn_01_t1.dk_vm_account
,dk_vf_account_dims_asoc                                        = z01_vf_sale_txn_01_t1.dk_vf_account_dims_asoc
,dk_vm_customer_demographic                                     = z01_vf_sale_txn_01_t1.dk_vm_customer_demographic
,dk_vm_customer_geography                                       = z01_vf_sale_txn_01_t1.dk_vm_customer_geography
,dk_vm_geo_code                                                 = z01_vf_sale_txn_01_t1.dk_vm_geo_code
,dk_vm_map_reference                                            = z01_vf_sale_txn_01_t1.dk_vm_map_reference
,dk_vm_response_type                                            = z01_vf_sale_txn_01_t1.dk_vm_response_type
,dk_vm_currency                                                 = z01_vf_sale_txn_01_t1.dk_vm_currency
,dk_vm_sales_rep                                                = z01_vf_sale_txn_01_t1.dk_vm_sales_rep
,dk_vm_unit_of_measure                                          = z01_vf_sale_txn_01_t1.dk_vm_unit_of_measure
,dk_vm_txn_type                                                 = z01_vf_sale_txn_01_t1.dk_vm_txn_type
,dk_vm_card                                                     = z01_vf_sale_txn_01_t1.dk_vm_card
,dk_vm_card_type                                                = z01_vf_sale_txn_01_t1.dk_vm_card_type
,dk_vm_comp_reporting_struct                                    = z01_vf_sale_txn_01_t1.dk_vm_comp_reporting_struct
,dk_vm_account_status                                           = z01_vf_sale_txn_01_t1.dk_vm_account_status
,dk_vm_account_type                                             = z01_vf_sale_txn_01_t1.dk_vm_account_type
,dk_vf_campaign_dims_asoc                                       = z01_vf_sale_txn_01_t1.dk_vf_campaign_dims_asoc
,dk_vf_solicitation_txn                                         = z01_vf_sale_txn_01_t1.dk_vf_solicitation_txn
,product_number                                                 = z01_vf_sale_txn_01_t1.product_number
```

```
,customer_number                                                = z01_vf_sale_txn_01_t1.customer_number
,card_type                                                      = z01_vf_sale_txn_01_t1.card_type
,card_number                                                    = z01_vf_sale_txn_01_t1.card_number
,account_number                                                 = z01_vf_sale_txn_01_t1.account_number
,account_type                                                   = z01_vf_sale_txn_01_t1.account_type
,account_status                                                 = z01_vf_sale_txn_01_t1.account_status
,currency_str                                                   = z01_vf_sale_txn_01_t1.currency_str
,postcode_str                                                   = z01_vf_sale_txn_01_t1.postcode_str
,address_key_str                                                = z01_vf_sale_txn_01_t1.address_key_str
,transaction_type                                               = z01_vf_sale_txn_01_t1.transaction_type
,sales_rep_number                                               = z01_vf_sale_txn_01_t1.sales_rep_number
,sale_unit_amount                                               = z01_vf_sale_txn_01_t1.sale_unit_amount
,sale_extended_amount                                           = z01_vf_sale_txn_01_t1.sale_extended_amount
,cost_unit_amount                                               = z01_vf_sale_txn_01_t1.cost_unit_amount
,cost_extended_amount                                           = z01_vf_sale_txn_01_t1.cost_extended_amount
,tax1_unit_amount                                               = z01_vf_sale_txn_01_t1.tax1_unit_amount
,tax1_extended_amount                                           = z01_vf_sale_txn_01_t1.tax1_extended_amount
,tax2_unit_amount                                               = z01_vf_sale_txn_01_t1.tax2_unit_amount
,tax2_extended_amount                                           = z01_vf_sale_txn_01_t1.tax2_extended_amount
,sale_units                                                     = z01_vf_sale_txn_01_t1.sale_units
,appsys_reference_number_str                                    = z01_vf_sale_txn_01_t1.appsys_reference_number_str
,sale_sdesc                                                     = z01_vf_sale_txn_01_t1.sale_sdesc
,sale_ldesc                                                     = z01_vf_sale_txn_01_t1.sale_ldesc
,sale_tstamp                                                    = z01_vf_sale_txn_01_t1.sale_tstamp
,number_sales                                                   = z01_vf_sale_txn_01_t1.number_sales
,sale_txn_returned_flag                                         = z01_vf_sale_txn_01_t1.sale_txn_returned_flag
,return_period_expired_flag                                     = z01_vf_sale_txn_01_t1.return_period_expired_flag
,return_probability_pct                                         = z01_vf_sale_txn_01_t1.return_probability_pct
,partitioning_column                                            = z01_vf_sale_txn_01_t1.partitioning_column
,partitioning_column_part_01                                    = z01_vf_sale_txn_01_t1.partitioning_column_part_01
,partitioning_column_part_02                                    = z01_vf_sale_txn_01_t1.partitioning_column_part_02
,row_del_frm_src_ind                                            = z01_vf_sale_txn_01_t1.row_del_frm_src_ind
,table_number                                                   = z01_vf_sale_txn_01_t1.table_number
,batch_number                                                   = z01_vf_sale_txn_01_t1.batch_number
,file_cycle_number                                              = z01_vf_sale_txn_01_t1.file_cycle_number
,view_number                                                    = z01_vf_sale_txn_01_t1.view_number
,audit_timestamp_01                                             = z01_vf_sale_txn_01_t1.audit_timestamp_01
from bi4all_02.admin.z01_vf_sale_txn_01_t1 z01_vf_sale_txn_01_t1
where 1=1
and tf_sale_txn.product_number                                  = z01_vf_sale_txn_01_t1.product_number
and tf_sale_txn.customer_number                                 = z01_vf_sale_txn_01_t1.customer_number
and tf_sale_txn.account_number                                  = z01_vf_sale_txn_01_t1.account_number
and tf_sale_txn.transaction_type                                = z01_vf_sale_txn_01_t1.transaction_type
and tf_sale_txn.sale_tstamp                                     = z01_vf_sale_txn_01_t1.sale_tstamp
;
```

The final step is the insert of the rows from the _t1 table that were not found in the target fact table based on the natural keys. Naturally this join is a very large join and may require large amounts of processing to complete.

```
-------------------------------------------------------------------------------------------------------------------------
--                                                                                                                     --
--   spreadsheet name:        bi4allmodeldemoworksheet01.xml spreadsheet version:        v3.1.00                      --
--   spreadsheet file name:   c:\seetl\designtime\3.1.00\sourcecode\seetl\bi4allmodeldemoworksheet01v3.1.00.xml        --
--   file name:               c:\temp\dmltype4\factp_z01_vf_sale_txn_01_facts_04.sql                                    --
--   windows domain:          emsbidev01                                                                                --
--   windows user:            administrator                                                                             --
--   windows workstation:     emsbidev01                                                                                --
--   seetl current directory: c:\seetl\designtime\3.1.00\sourcecode\seetl                                               --
--   windows version:         microsoft windows nt 5.2.3790 service pack 2                                              --
--   dot net version:         2.0.50727.1433                                                                            --
--   seetl workbench version: SeETL 3.1.00 - 2010-06-11                                                                 --
--   timestamp:               25/07/2010 10:25:24                                                                       --
--   description:             description                                                                               --
--                                                                                                                     --
-------------------------------------------------------------------------------------------------------------------------
insert into bi4all_02.admin.tf_sale_txn
(
 dk_vm_sale_date
,dk_vm_sale_minute
,dk_vm_sub_campaign
,dk_vm_product
,dk_vm_customer
,dk_vf_customer_dims_asoc
,dk_vm_account
,dk_vf_account_dims_asoc
,dk_vm_customer_demographic
,dk_vm_customer_geography
,dk_vm_geo_code
,dk_vm_map_reference
,dk_vm_response_type
,dk_vm_currency
,dk_vm_sales_rep
,dk_vm_unit_of_measure
,dk_vm_txn_type
,dk_vm_card
,dk_vm_card_type
,dk_vm_comp_reporting_struct
```

```
,dk_vm_account_status
,dk_vm_account_type
,dk_vf_campaign_dims_asoc
,dk_vf_solicitation_txn
,pk_tf_sale_txn
,product_number
,customer_number
,card_type
,card_number
,account_number
,account_type
,account_status
,currency_str
,postcode_str
,address_key_str
,transaction_type
,sales_rep_number
,sale_unit_amount
,sale_extended_amount
,cost_unit_amount
,cost_extended_amount
,tax1_unit_amount
,tax1_extended_amount
,tax2_unit_amount
,tax2_extended_amount
,sale_units
,appsys_reference_number_str
,sale_sdesc
,sale_ldesc
,sale_tstamp
,number_sales
,sale_txn_returned_flag
,return_period_expired_flag
,return_probability_pct
,partitioning_column
,partitioning_column_part_01
,partitioning_column_part_02
,row_del_frm_src_ind
,table_number
,batch_number
,file_cycle_number
,view_number
```

```
,audit_timestamp_01
)
select
 z01_vf_sale_txn_01_t1.dk_vm_sale_date
,z01_vf_sale_txn_01_t1.dk_vm_sale_minute
,z01_vf_sale_txn_01_t1.dk_vm_sub_campaign
,z01_vf_sale_txn_01_t1.dk_vm_product
,z01_vf_sale_txn_01_t1.dk_vm_customer
,z01_vf_sale_txn_01_t1.dk_vf_customer_dims_asoc
,z01_vf_sale_txn_01_t1.dk_vm_account
,z01_vf_sale_txn_01_t1.dk_vf_account_dims_asoc
,z01_vf_sale_txn_01_t1.dk_vm_customer_demographic
,z01_vf_sale_txn_01_t1.dk_vm_customer_geography
,z01_vf_sale_txn_01_t1.dk_vm_geo_code
,z01_vf_sale_txn_01_t1.dk_vm_map_reference
,z01_vf_sale_txn_01_t1.dk_vm_response_type
,z01_vf_sale_txn_01_t1.dk_vm_currency
,z01_vf_sale_txn_01_t1.dk_vm_sales_rep
,z01_vf_sale_txn_01_t1.dk_vm_unit_of_measure
,z01_vf_sale_txn_01_t1.dk_vm_txn_type
,z01_vf_sale_txn_01_t1.dk_vm_card
,z01_vf_sale_txn_01_t1.dk_vm_card_type
,z01_vf_sale_txn_01_t1.dk_vm_comp_reporting_struct
,z01_vf_sale_txn_01_t1.dk_vm_account_status
,z01_vf_sale_txn_01_t1.dk_vm_account_type
,z01_vf_sale_txn_01_t1.dk_vf_campaign_dims_asoc
,z01_vf_sale_txn_01_t1.dk_vf_solicitation_txn
,z01_vf_sale_txn_01_t1.pk_vf_sale_txn
,z01_vf_sale_txn_01_t1.product_number
,z01_vf_sale_txn_01_t1.customer_number
,z01_vf_sale_txn_01_t1.card_type
,z01_vf_sale_txn_01_t1.card_number
,z01_vf_sale_txn_01_t1.account_number
,z01_vf_sale_txn_01_t1.account_type
,z01_vf_sale_txn_01_t1.account_status
,z01_vf_sale_txn_01_t1.currency_str
,z01_vf_sale_txn_01_t1.postcode_str
,z01_vf_sale_txn_01_t1.address_key_str
,z01_vf_sale_txn_01_t1.transaction_type
,z01_vf_sale_txn_01_t1.sales_rep_number
,z01_vf_sale_txn_01_t1.sale_unit_amount
,z01_vf_sale_txn_01_t1.sale_extended_amount
```

```
,z01_vf_sale_txn_01_t1.cost_unit_amount
,z01_vf_sale_txn_01_t1.cost_extended_amount
,z01_vf_sale_txn_01_t1.tax1_unit_amount
,z01_vf_sale_txn_01_t1.tax1_extended_amount
,z01_vf_sale_txn_01_t1.tax2_unit_amount
,z01_vf_sale_txn_01_t1.tax2_extended_amount
,z01_vf_sale_txn_01_t1.sale_units
,z01_vf_sale_txn_01_t1.appsys_reference_number_str
,z01_vf_sale_txn_01_t1.sale_sdesc
,z01_vf_sale_txn_01_t1.sale_ldesc
,z01_vf_sale_txn_01_t1.sale_tstamp
,z01_vf_sale_txn_01_t1.number_sales
,z01_vf_sale_txn_01_t1.sale_txn_returned_flag
,z01_vf_sale_txn_01_t1.return_period_expired_flag
,z01_vf_sale_txn_01_t1.return_probability_pct
,z01_vf_sale_txn_01_t1.partitioning_column
,z01_vf_sale_txn_01_t1.partitioning_column_part_01
,z01_vf_sale_txn_01_t1.partitioning_column_part_02
,z01_vf_sale_txn_01_t1.row_del_frm_src_ind
,z01_vf_sale_txn_01_t1.table_number
,z01_vf_sale_txn_01_t1.batch_number
,z01_vf_sale_txn_01_t1.file_cycle_number
,z01_vf_sale_txn_01_t1.view_number
,z01_vf_sale_txn_01_t1.audit_timestamp_01
from bi4all_02.admin.z01_vf_sale_txn_01_t1 z01_vf_sale_txn_01_t1
left join bi4all_02.admin.tf_sale_txn tf_sale_txn
on
(
    tf_sale_txn.product_number = z01_vf_sale_txn_01_t1.product_number
and tf_sale_txn.customer_number = z01_vf_sale_txn_01_t1.customer_number
and tf_sale_txn.account_number = z01_vf_sale_txn_01_t1.account_number
and tf_sale_txn.transaction_type = z01_vf_sale_txn_01_t1.transaction_type
and tf_sale_txn.sale_tstamp = z01_vf_sale_txn_01_t1.sale_tstamp
)
where 1=1
and tf_sale_txn.product_number is null
and tf_sale_txn.customer_number is null
and tf_sale_txn.account_number is null
and tf_sale_txn.transaction_type is null
and tf_sale_txn.sale_tstamp is null
;
```

7.1.2. Two Step Processing without Help Key Tables

The two step processing without help keys configuration presumes that there is an identical structure of the data warehouse in the staging area that contains all data that might be updated. There are a number of advantages of doing this.

1. The staging area provides a readily available ‘hot backup’ for production.
2. It is possible to distribute copies of the data warehouse downstream to numerous exactly copies of the data warehouse.

Again, there are the same 5 sets of processing that is performed. In these samples we will cut out some of the redundant list of fields. We have included all fields in the samples above. The reader is asked to take it on faith that the sql is generated for each field as it has been above. By cutting out much of the list of fields we will produce a little less volume in the documentation.

1. Type 1 dimension table processing
2. Type 2 dimension table processing without field level sensitivity.
3. Type 2 dimension table processing with field level sensitivity.
4. Fact table processing without a key on the fact table.
5. Fact table processing with a key on the fact table.

7.1.2.1. Type 1 dimension table processing

The first step is to create the data in the _t1 temp table based on the input view as per normal.

```
truncate table bi4all_01.admin.z01_vm_address_01_t1
;

insert into bi4all_01.admin.z01_vm_address_01_t1
(
 pk_vm_address
,level_col
,dim_char_ky_fld
,vm_address_seq
,address_key_str
,drop_point_id
,address_line_1
,address_line_2
,address_line_3
,address_line_4
)
select
 z01_vm_address_01.z01_vm_address_01_seq
,'detail'
,z01_vm_address_01.z01_vm_address_01_key
,z01_vm_address_01.z01_vm_address_01_seq
,z01_vm_address_01.address_key_str
,z01_vm_address_01.drop_point_id
,z01_vm_address_01.address_line_1
,z01_vm_address_01.address_line_2
,z01_vm_address_01.address_line_3
,z01_vm_address_01.address_line_4
from bi4all_01.admin.z01_vm_address_01 z01_vm_address_01
limit 100
;
```

The next step is to update the target table inside the staging area.

```
update bi4all_01.admin.tm_address set
 level_col                                                    = z01_vm_address_01_t1.level_col
,dim_char_ky_fld                                              = z01_vm_address_01_t1.dim_char_ky_fld
,address_key_str                                              = z01_vm_address_01_t1.address_key_str
,drop_point_id                                                = z01_vm_address_01_t1.drop_point_id
,address_line_1                                               = z01_vm_address_01_t1.address_line_1
,address_line_2                                               = z01_vm_address_01_t1.address_line_2
,address_line_3                                               = z01_vm_address_01_t1.address_line_3
,address_line_4                                               = z01_vm_address_01_t1.address_line_4

from bi4all_01.admin.z01_vm_address_01_t1 z01_vm_address_01_t1

where 1=1
and bi4all_01.admin.tm_address.level_col                        = z01_vm_address_01_t1.level_col
and bi4all_01.admin.tm_address.dim_char_ky_fld                  = z01_vm_address_01_t1.dim_char_ky_fld
;
```

The next step is to insert the rows into the target table in the staging area for those rows that are new.

```
insert into bi4all_01.admin.tm_address
(
 pk_tm_address
,level_col
,dim_char_ky_fld
,tm_address_seq
,address_key_str
,drop_point_id
,address_line_1
,address_line_2
,address_line_3
,address_line_4
)
select
 z01_vm_address_01_t1.pk_vm_address
,z01_vm_address_01_t1.level_col
,z01_vm_address_01_t1.dim_char_ky_fld
,z01_vm_address_01_t1.vm_address_seq
,z01_vm_address_01_t1.address_key_str
,z01_vm_address_01_t1.drop_point_id
,z01_vm_address_01_t1.address_line_1
,z01_vm_address_01_t1.address_line_2
,z01_vm_address_01_t1.address_line_3
,z01_vm_address_01_t1.address_line_4

from bi4all_01.admin.z01_vm_address_01_t1 z01_vm_address_01_t1

left join bi4all_01.admin.tm_address tm_address on 1=1
and
(
    tm_address.level_col       = z01_vm_address_01_t1.level_col
and tm_address.dim_char_ky_fld = z01_vm_address_01_t1.dim_char_ky_fld
)
where tm_address.dim_char_ky_fld is null
;
```

The next step is to update the target table in the target database where the row exists in the target table. Note that the table number and batch number is also used as well as the primary key because it is possible for primary keys of tables to be reused if data is co-habited in a single table. Notice that the batch number is also used as a constraint to make sure that only rows affected in the current batch are moved. The target table in the staging area has all rows for all dimensions in it and care must be taken not to move these all forward.

```
update bi4all_02.admin.tm_address set
 level_col                                                                  = tm_address_s.level_col
,dim_char_ky_fld                                                            = tm_address_s.dim_char_ky_fld
,address_key_str                                                            = tm_address_s.address_key_str
,drop_point_id                                                              = tm_address_s.drop_point_id
,address_line_1                                                             = tm_address_s.address_line_1
,address_line_2                                                             = tm_address_s.address_line_2
,address_line_3                                                             = tm_address_s.address_line_3
,address_line_4                                                             = tm_address_s.address_line_4

from bi4all_01.admin.tm_address tm_address_s

where 1=1
and bi4all_02.admin.tm_address.pk_tm_address                                  = tm_address_s.pk_tm_address
and bi4all_02.admin.tm_address.table_number                                   = tm_address_s.table_number
and tm_address_s.batch_number                                               = (select pk_batch_number from
bi4all_01.admin.ctl_batch_control where batch_complete_flag = 0 )
;
```

The next step is to move the rows forward that do not exist in the target table in the target database. Again notice the use of table number and the ‘batch number to minimise the number of keys to be sent from the staging database to the target database to be tested.

```
insert into bi4all_02.admin.tm_address
(
 pk_tm_address
,level_col
,dim_char_ky_fld
,tm_address_seq
,address_key_str
,drop_point_id
,address_line_1
,address_line_2
,address_line_3
,address_line_4
)
select
 tm_address_s.pk_tm_address
,tm_address_s.level_col
,tm_address_s.dim_char_ky_fld
,tm_address_s.tm_address_seq
,tm_address_s.address_key_str
,tm_address_s.drop_point_id
,tm_address_s.address_line_1
,tm_address_s.address_line_2
,tm_address_s.address_line_3
,tm_address_s.address_line_4

from bi4all_01.admin.tm_address tm_address_s

left join bi4all_02.admin.tm_address tm_address_t
on 1=1
and tm_address_s.pk_tm_address                                          = tm_address_t.pk_tm_address
and tm_address_s.table_number                                           = tm_address_t.table_number

where 1=1
and tm_address_t.pk_tm_address is null
and tm_address_s.batch_number                                           = (select pk_batch_number from
bi4all_01.admin.ctl_batch_control where batch_complete_flag = 0 )
;
```

7.1.2.2. Type 2 dimension table processing without field level sensitivity.

The first step as usual is to populate the _t1 temp table.

```
truncate table bi4all_01.admin.z01_vm_product_01_t1
;

insert into bi4all_01.admin.z01_vm_product_01_t1
(
 pk_vm_product
,level_col
,dim_char_ky_fld
,vm_product_seq
,current_flag
,date_from
,date_to
,product_number
,sku_number
,barcode_num
,prod_name
)
select
 z01_vm_product_01.z01_vm_product_01_seq
,'detail'
,z01_vm_product_01.z01_vm_product_01_key
,z01_vm_product_01.z01_vm_product_01_seq
,1
,ctl_batch_control.batch_date
,ctl_month_control.high_value_date
,z01_vm_product_01.product_number
,z01_vm_product_01.sku_number
,z01_vm_product_01.barcode_num
,z01_vm_product_01.prod_name
from bi4all_01.admin.z01_vm_product_01 z01_vm_product_01
left outer join bi4all_01.admin.ctl_batch_control ctl_batch_control on 1=1
and ctl_batch_control.batch_complete_flag = 0
left outer join bi4all_01.admin.ctl_month_control ctl_month_control on 1=1
limit 100;
```

The next step is to update the current flag and close the record in the target table inside the staging area.

```
update bi4all_01.admin.tm_product set
 current_flag                                                                = 0
,date_to                                                                     = ctl_batch_control.batch_date - 1
,batch_number                                                                = z01_vm_product_01_t1.batch_number

from bi4all_01.admin.z01_vm_product_01_t1 z01_vm_product_01_t1

left outer join bi4all_01.admin.ctl_batch_control ctl_batch_control on 1=1
and ctl_batch_control.batch_complete_flag = 0

where 1=1
and tm_product.level_col                                                     = z01_vm_product_01_t1.level_col
and tm_product.dim_char_ky_fld                                               = z01_vm_product_01_t1.dim_char_ky_fld
and tm_product.current_flag                                                  = 1
;
```

The next step is to insert the records that are new into the target table in the staging area.

```
insert into bi4all_01.admin.tm_product
(
 pk_tm_product
,level_col
,dim_char_ky_fld
,current_flag
,date_from
,date_to
,tm_product_seq
,product_number
,sku_number
,barcode_num
,prod_name
)
select
 z01_vm_product_01_t1.pk_vm_product
,z01_vm_product_01_t1.level_col
,z01_vm_product_01_t1.dim_char_ky_fld
,1
,z01_vm_product_01_t1.date_from
,z01_vm_product_01_t1.date_to
,z01_vm_product_01_t1.vm_product_seq
,z01_vm_product_01_t1.product_number
,z01_vm_product_01_t1.sku_number
,z01_vm_product_01_t1.barcode_num
,z01_vm_product_01_t1.prod_name

from bi4all_01.admin.z01_vm_product_01_t1 z01_vm_product_01_t1

;
```

The next step is to update the records that are changed in the target table in the target database.

```
update bi4all_02.admin.tm_product set
 level_col                                                    = tm_product_s.level_col
,dim_char_ky_fld                                              = tm_product_s.dim_char_ky_fld
,current_flag                                                 = tm_product_s.current_flag
,date_from                                                    = tm_product_s.date_from
,date_to                                                      = tm_product_s.date_to
,product_number                                               = tm_product_s.product_number
,sku_number                                                   = tm_product_s.sku_number
,barcode_num                                                  = tm_product_s.barcode_num
,prod_name                                                    = tm_product_s.prod_name

from bi4all_01.admin.tm_product tm_product_s

where 1=1
and tm_product_s.pk_tm_product                                = tm_product.pk_tm_product
and tm_product_s.table_number                                 = tm_product.table_number
and tm_product_s.batch_number                                 = (select pk_batch_number from
bi4all_01.admin.ctl_batch_control where batch_complete_flag = 0 )
;
```

The next step is to insert the records that are new into the target table in the staging area.

```
insert into bi4all_02.admin.tm_product
(
 pk_tm_product
,level_col
,dim_char_ky_fld
,current_flag
,date_from
,date_to
,tm_product_seq
,product_number
,sku_number
,barcode_num
,prod_name
)
select
 tm_product_s.pk_tm_product
,tm_product_s.level_col
,tm_product_s.dim_char_ky_fld
,tm_product_s.current_flag
,tm_product_s.date_from
,tm_product_s.date_to
,tm_product_s.tm_product_seq
,tm_product_s.product_number
,tm_product_s.sku_number
,tm_product_s.barcode_num
,tm_product_s.prod_name

from bi4all_01.admin.tm_product tm_product_s
left join bi4all_02.admin.tm_product tm_product on 1=1
and tm_product_s.pk_tm_product                                          = tm_product.pk_tm_product
and tm_product_s.table_number                                           = tm_product.table_number
where 1=1
and tm_product.pk_tm_product is null
and tm_product_s.batch_number                                           = (select pk_batch_number from
bi4all_01.admin.ctl_batch_control where batch_complete_flag = 0 )
```

7.1.2.3. Type 2 dimesnion table processing with field level sensitivity.

The first step as normal is to select the source data and place it into the _t1 temporary table.

```
insert into bi4all_01.admin.z01_vm_product_01_t1
(
 pk_vm_product
,level_col
,dim_char_ky_fld
,vm_product_seq
,current_flag
,date_from
,date_to
,product_number
,sku_number
,barcode_num
,prod_name
)
select
 z01_vm_product_01.z01_vm_product_01_seq
,'detail'
,z01_vm_product_01.z01_vm_product_01_key
,z01_vm_product_01.z01_vm_product_01_seq
,1
,ctl_batch_control.batch_date
,ctl_month_control.high_value_date
,z01_vm_product_01.product_number
,z01_vm_product_01.sku_number
,z01_vm_product_01.barcode_num
,z01_vm_product_01.prod_name
from bi4all_01.admin.z01_vm_product_01 z01_vm_product_01

left outer join bi4all_01.admin.ctl_batch_control ctl_batch_control on 1=1
and ctl_batch_control.batch_complete_flag = 0

left outer join bi4all_01.admin.ctl_month_control ctl_month_control on 1=1
```

The second part is to update the current_flag and date to fields for the rows that will be closed.

```
update bi4all_01.admin.tm_product set
 current_flag                                                                    = 0
,date_to                                                                         = ctl_batch_control.batch_date - 1
,batch_number                                                                    = z01_vm_product_01_t1.batch_number
from bi4all_01.admin.z01_vm_product_01_t1 z01_vm_product_01_t1
left outer join bi4all_01.admin.ctl_batch_control ctl_batch_control on 1=1
and ctl_batch_control.batch_complete_flag = 0
where 1=1
and tm_product.level_col                                                         = z01_vm_product_01_t1.level_col
and tm_product.dim_char_ky_fld                                                   = z01_vm_product_01_t1.dim_char_ky_fld
and tm_product.current_flag                                                      = 1
and
(
   not ( z01_vm_product_01_t1.sku_number = tm_product.sku_number or ( z01_vm_product_01_t1.sku_number is null and tm_product.sku_number
is null ))
or not ( z01_vm_product_01_t1.barcode_num = tm_product.barcode_num or ( z01_vm_product_01_t1.barcode_num is null and
tm_product.barcode_num is null ))
or not ( z01_vm_product_01_t1.prod_name = tm_product.prod_name or ( z01_vm_product_01_t1.prod_name is null and tm_product.prod_name is
null ))
or not ( z01_vm_product_01_t1.prod_sub_group = tm_product.prod_sub_group or ( z01_vm_product_01_t1.prod_sub_group is null and
tm_product.prod_sub_group is null ))
)
```

The next step is to update the rows that the field level sensitivity indicates should have type 1 changes applied to them.

```
update bi4all_01.admin.tm_product set
 product_number                                                    = z01_vm_product_01_t1.product_number
,sku_number                                                        = z01_vm_product_01_t1.sku_number
,barcode_num                                                       = z01_vm_product_01_t1.barcode_num
,prod_name                                                         = z01_vm_product_01_t1.prod_name
from bi4all_01.admin.z01_vm_product_01_t1 z01_vm_product_01_t1

where 1=1
and tm_product.level_col                                           = z01_vm_product_01_t1.level_col
and tm_product.dim_char_ky_fld                                     = z01_vm_product_01_t1.dim_char_ky_fld
and tm_product.current_flag                                        = 1
and ( z01_vm_product_01_t1.sku_number = tm_product.sku_number or ( z01_vm_product_01_t1.sku_number is null and
tm_product.sku_number is null ))
and ( z01_vm_product_01_t1.barcode_num = tm_product.barcode_num or ( z01_vm_product_01_t1.barcode_num is null and
tm_product.barcode_num is null ))
and ( z01_vm_product_01_t1.prod_name = tm_product.prod_name or ( z01_vm_product_01_t1.prod_name is null and
tm_product.prod_name is null ))
and ( z01_vm_product_01_t1.prod_sub_group = tm_product.prod_sub_group or ( z01_vm_product_01_t1.prod_sub_group is null and
tm_product.prod_sub_group is null ))
;
```

The next step is to insert the rows that are new into the target table in the staging area.

```
insert into bi4all_01.admin.tm_product
(
 pk_tm_product
,level_col
,dim_char_ky_fld
,current_flag
,date_from
,date_to
,tm_product_seq
,product_number
,sku_number
,barcode_num
,prod_name
)
select
 z01_vm_product_01_t1.pk_vm_product
,z01_vm_product_01_t1.level_col
,z01_vm_product_01_t1.dim_char_ky_fld
,1
,z01_vm_product_01_t1.date_from
,z01_vm_product_01_t1.date_to
,z01_vm_product_01_t1.vm_product_seq
,z01_vm_product_01_t1.product_number
,z01_vm_product_01_t1.sku_number
,z01_vm_product_01_t1.barcode_num
,z01_vm_product_01_t1.prod_name
from bi4all_01.admin.z01_vm_product_01_t1 z01_vm_product_01_t1

left outer join bi4all_01.admin.ctl_batch_control ctl_batch_control on 1=1
and ctl_batch_control.batch_complete_flag = 0

left outer join bi4all_01.admin.ctl_month_control ctl_month_control on 1=1

left join bi4all_01.admin.tm_product tm_product on 1=1
and tm_product.level_col                                    = z01_vm_product_01_t1.level_col
and tm_product.dim_char_ky_fld                              = z01_vm_product_01_t1.dim_char_ky_fld
and tm_product.current_flag                                 = 1

where 1=1
```

```
and tm_product.level_col                                    is null
and tm_product.dim_char_ky_fld                              is null
and tm_product.current_flag                                 is null
;
```

The next step is to perform the update of the target rows in the target database for those rows that are detected as having changed in the current batch.

```
update bi4all_02.admin.tm_product set
 level_col                                                                  = tm_product_s.level_col
,dim_char_ky_fld                                                            = tm_product_s.dim_char_ky_fld
,current_flag                                                               = tm_product_s.current_flag
,date_from                                                                  = tm_product_s.date_from
,date_to                                                                    = tm_product_s.date_to
,product_number                                                             = tm_product_s.product_number
,sku_number                                                                 = tm_product_s.sku_number
,barcode_num                                                                = tm_product_s.barcode_num
,prod_name                                                                  = tm_product_s.prod_name
from bi4all_01.admin.tm_product tm_product_s

where 1=1
and tm_product_s.pk_tm_product                                              = tm_product.pk_tm_product
and tm_product_s.table_number                                               = tm_product.table_number
and tm_product_s.batch_number                                               = (select pk_batch_number from
bi4all_01.admin.ctl_batch_control where batch_complete_flag = 0 )
;
```

The next step is to move the rows that are new from the target table in the staging area to the target table in the target database.

```
insert into bi4all_02.admin.tm_product
(
 pk_tm_product
,level_col
,dim_char_ky_fld
,current_flag
,date_from
,date_to
,tm_product_seq
,product_number
,sku_number
,barcode_num
,prod_name

)
select
 tm_product_s.pk_tm_product
,tm_product_s.level_col
,tm_product_s.dim_char_ky_fld
,tm_product_s.current_flag
,tm_product_s.date_from
,tm_product_s.date_to
,tm_product_s.tm_product_seq
,tm_product_s.product_number
,tm_product_s.sku_number
,tm_product_s.barcode_num
,tm_product_s.prod_name
from bi4all_01.admin.tm_product tm_product_s
left join bi4all_02.admin.tm_product tm_product
on 1=1
and tm_product_s.pk_tm_product                                          = tm_product.pk_tm_product
and tm_product_s.table_number                                           = tm_product.table_number

where 1=1
and tm_product.pk_tm_product is null
and tm_product_s.batch_number                                           = (select pk_batch_number from
bi4all_01.admin.ctl_batch_control where batch_complete_flag = 0 )
;
```

7.1.2.4. Fact table processing without a key on the fact table.

The first step as per usual is to place the new records into the _t1 temp table, Please note that we have cut many of the columns out for brevity.

```
insert into bi4all_01.admin.z01_vf_sale_txn_01_t1
(
 dk_vm_sale_date
,dk_vm_sale_minute
,dk_vm_sub_campaign
,dk_vm_product
,dk_vm_customer
,pk_vf_sale_txn
,product_number
,customer_number
)
select
 nvl(dk_zvm_sale_date, 0 )
,nvl(dk_zvm_sale_minute, 0 )
,nvl(dk_zvm_sub_campaign, 0 )
,nvl(dk_zvm_product, 0 )
,nvl(dk_zvm_customer, 0 )
,ctl_sequence_number
,product_number
,customer_number
from bi4all_01.admin.z01_vf_sale_txn_01 z01_vf_sale_txn_01
left join bi4all_01.admin.zvm_sale_date_at3   zvm_sale_date_at3
on z01_vf_sale_txn_01.char_key_zvm_sale_date = zvm_sale_date_at3.pk_dim_char_ky_fld
left join bi4all_01.admin.zvm_sale_minute_at3   zvm_sale_minute_at3
on z01_vf_sale_txn_01.char_key_zvm_sale_minute = zvm_sale_minute_at3.pk_dim_char_ky_fld
left join bi4all_01.admin.zvm_sub_campaign_at3   zvm_sub_campaign_at3
on z01_vf_sale_txn_01.char_key_zvm_sub_campaign = zvm_sub_campaign_at3.pk_dim_char_ky_fld
left join bi4all_01.admin.zvm_product_at3   zvm_product_at3
on z01_vf_sale_txn_01.char_key_zvm_product = zvm_product_at3.pk_dim_char_ky_fld
left join bi4all_01.admin.zvm_customer_at3   zvm_customer_at3
on z01_vf_sale_txn_01.char_key_zvm_customer = zvm_customer_at3.pk_dim_char_ky_fld
```

The new fact rows are then placed into the target table in the staging area.

```
insert into bi4all_01.admin.tf_sale_txn
(
 dk_vm_sale_date
,dk_vm_sale_minute
,dk_vm_sub_campaign
,dk_vm_product
,dk_vm_customer
,pk_tf_sale_txn
,product_number
,customer_number
)
select
 z01_vf_sale_txn_01_t1.dk_vm_sale_date
,z01_vf_sale_txn_01_t1.dk_vm_sale_minute
,z01_vf_sale_txn_01_t1.dk_vm_sub_campaign
,z01_vf_sale_txn_01_t1.dk_vm_product
,z01_vf_sale_txn_01_t1.dk_vm_customer
,z01_vf_sale_txn_01_t1.pk_vf_sale_txn
,z01_vf_sale_txn_01_t1.product_number
,z01_vf_sale_txn_01_t1.customer_number

from bi4all_01.admin.z01_vf_sale_txn_01_t1 z01_vf_sale_txn_01_t1
;
```

The next step is to update the rows of the target table in the target database based on those that have changed in the target table in the staging area.

```
update bi4all_02.admin.tf_sale_txn set
 dk_vm_sale_date                                                    = tf_sale_txn_s.dk_vm_sale_date
,dk_vm_sale_minute                                                  = tf_sale_txn_s.dk_vm_sale_minute
,dk_vm_sub_campaign                                                 = tf_sale_txn_s.dk_vm_sub_campaign
,dk_vm_product                                                      = tf_sale_txn_s.dk_vm_product
,dk_vm_customer                                                     = tf_sale_txn_s.dk_vm_customer
,dk_vf_solicitation_txn                                             = tf_sale_txn_s.dk_vf_solicitation_txn
,product_number                                                     = tf_sale_txn_s.product_number
,customer_number                                                    = tf_sale_txn_s.customer_number
from bi4all_01.admin.tf_sale_txn tf_sale_txn_s
where 1=1
and tf_sale_txn.pk_tf_sale_txn                                      = tf_sale_txn_s.pk_tf_sale_txn
and tf_sale_txn.table_number                                        = tf_sale_txn_s.table_number
and tf_sale_txn_s.batch_number                                      = (select pk_batch_number from
bi4all_01.admin.ctl_batch_control where ctl_batch_control.batch_complete_flag = 0 )
;
```

The last part is to insert those rows into the target fact table in the target database that are new in the target table in the source database.

```
insert into bi4all_02.admin.tf_sale_txn
(
 dk_vm_sale_date
,dk_vm_sale_minute
,dk_vm_sub_campaign
,dk_vm_product
,dk_vm_customer
,pk_tf_sale_txn
,product_number
,customer_number
)
select
 tf_sale_txn_s.dk_vm_sale_date
,tf_sale_txn_s.dk_vm_sale_minute
,tf_sale_txn_s.dk_vm_sub_campaign
,tf_sale_txn_s.dk_vm_product
,tf_sale_txn_s.dk_vm_customer
,tf_sale_txn_s.pk_tf_sale_txn
,tf_sale_txn_s.product_number
,tf_sale_txn_s.customer_number
from bi4all_01.admin.tf_sale_txn tf_sale_txn_s
left join bi4all_02.admin.tf_sale_txn tf_sale_txn on 1=1
and tf_sale_txn.pk_tf_sale_txn                                          = tf_sale_txn_s.pk_tf_sale_txn
and tf_sale_txn.table_number                                            = tf_sale_txn_s.table_number
where 1=1
and tf_sale_txn.pk_tf_sale_txn                                          is null
and tf_sale_txn_s.batch_number                                          = (select pk_batch_number from
bi4all_01.admin.ctl_batch_control where ctl_batch_control.batch_complete_flag = 0 )
;
```

7.1.2.5. Fact table processing with a key on the fact table.

The first step as per usual is to place the new records into the _t1 temp table, Please note that we have cut many of the columns out for brevity.

```
insert into bi4all_01.admin.z01_vf_sale_txn_01_t1
(
 dk_vm_sale_date
,dk_vm_sale_minute
,dk_vm_sub_campaign
,dk_vm_product
,dk_vm_customer
,pk_vf_sale_txn
,product_number
,customer_number
)
select
 nvl(dk_zvm_sale_date, 0 )
,nvl(dk_zvm_sale_minute, 0 )
,nvl(dk_zvm_sub_campaign, 0 )
,nvl(dk_zvm_product, 0 )
,nvl(dk_zvm_customer, 0 )
,ctl_sequence_number
,product_number
,customer_number
from bi4all_01.admin.z01_vf_sale_txn_01 z01_vf_sale_txn_01
left join bi4all_01.admin.zvm_sale_date_at3   zvm_sale_date_at3
on z01_vf_sale_txn_01.char_key_zvm_sale_date = zvm_sale_date_at3.pk_dim_char_ky_fld
left join bi4all_01.admin.zvm_sale_minute_at3   zvm_sale_minute_at3
on z01_vf_sale_txn_01.char_key_zvm_sale_minute = zvm_sale_minute_at3.pk_dim_char_ky_fld
left join bi4all_01.admin.zvm_sub_campaign_at3   zvm_sub_campaign_at3
on z01_vf_sale_txn_01.char_key_zvm_sub_campaign = zvm_sub_campaign_at3.pk_dim_char_ky_fld
left join bi4all_01.admin.zvm_product_at3   zvm_product_at3
on z01_vf_sale_txn_01.char_key_zvm_product = zvm_product_at3.pk_dim_char_ky_fld
left join bi4all_01.admin.zvm_customer_at3   zvm_customer_at3
on z01_vf_sale_txn_01.char_key_zvm_customer = zvm_customer_at3.pk_dim_char_ky_fld
```

The next step is to update the rows that have been updated in the target table in the staging area.

```
update bi4all_01.admin.tf_sale_txn set
 dk_vm_sale_date                                                    = z01_vf_sale_txn_01_t1.dk_vm_sale_date
,dk_vm_sale_minute                                                  = z01_vf_sale_txn_01_t1.dk_vm_sale_minute
,dk_vm_sub_campaign                                                 = z01_vf_sale_txn_01_t1.dk_vm_sub_campaign
,dk_vm_product                                                      = z01_vf_sale_txn_01_t1.dk_vm_product
,dk_vm_customer                                                     = z01_vf_sale_txn_01_t1.dk_vm_customer
,product_number                                                     = z01_vf_sale_txn_01_t1.product_number
,customer_number                                                    = z01_vf_sale_txn_01_t1.customer_number
from bi4all_01.admin.z01_vf_sale_txn_01_t1 z01_vf_sale_txn_01_t1
where 1=1
and tf_sale_txn.product_number                                      = z01_vf_sale_txn_01_t1.product_number
and tf_sale_txn.customer_number                                     = z01_vf_sale_txn_01_t1.customer_number
and tf_sale_txn.account_number                                      = z01_vf_sale_txn_01_t1.account_number
and tf_sale_txn.transaction_type                                    = z01_vf_sale_txn_01_t1.transaction_type
and tf_sale_txn.sale_tstamp                                         = z01_vf_sale_txn_01_t1.sale_tstamp
;
```

The next step is to move the rows that are considered new into the target table in the staging area.

```
insert into bi4all_01.admin.tf_sale_txn
(
 dk_vm_sale_date
,dk_vm_sale_minute
,dk_vm_sub_campaign
,dk_vm_product
,dk_vm_customer
,pk_tf_sale_txn
,product_number
,customer_number
)
select
 z01_vf_sale_txn_01_t1.dk_vm_sale_date
,z01_vf_sale_txn_01_t1.dk_vm_sale_minute
,z01_vf_sale_txn_01_t1.dk_vm_sub_campaign
,z01_vf_sale_txn_01_t1.dk_vm_product
,z01_vf_sale_txn_01_t1.dk_vm_customer
,z01_vf_sale_txn_01_t1.pk_vf_sale_txn
,z01_vf_sale_txn_01_t1.product_number
,z01_vf_sale_txn_01_t1.customer_number

from bi4all_01.admin.z01_vf_sale_txn_01_t1 z01_vf_sale_txn_01_t1

left join bi4all_01.admin.tf_sale_txn tf_sale_txn
on 1=1
and tf_sale_txn.product_number                                  = z01_vf_sale_txn_01_t1.product_number
and tf_sale_txn.customer_number                                 = z01_vf_sale_txn_01_t1.customer_number
and tf_sale_txn.account_number                                  = z01_vf_sale_txn_01_t1.account_number
and tf_sale_txn.transaction_type                                = z01_vf_sale_txn_01_t1.transaction_type
and tf_sale_txn.sale_tstamp                                     = z01_vf_sale_txn_01_t1.sale_tstamp
where 1=1
and tf_sale_txn.product_number is null
and tf_sale_txn.customer_number is null
and tf_sale_txn.account_number is null
and tf_sale_txn.transaction_type is null
and tf_sale_txn.sale_tstamp is null
;
```

The next step is to move the rows that have been updated from the target table in the staging area to the target table in the target database.

```
update bi4all_02.admin.tf_sale_txn set
 dk_vm_sale_date                                                    = tf_sale_txn_s.dk_vm_sale_date
,dk_vm_sale_minute                                                  = tf_sale_txn_s.dk_vm_sale_minute
,dk_vm_sub_campaign                                                 = tf_sale_txn_s.dk_vm_sub_campaign
,dk_vm_product                                                      = tf_sale_txn_s.dk_vm_product
,dk_vm_customer                                                     = tf_sale_txn_s.dk_vm_customer
,product_number                                                     = tf_sale_txn_s.product_number
,customer_number                                                    = tf_sale_txn_s.customer_number

from bi4all_01.admin.tf_sale_txn tf_sale_txn_s

where 1=1
and tf_sale_txn.pk_tf_sale_txn                                      = tf_sale_txn_s.pk_tf_sale_txn
and tf_sale_txn.table_number                                        = tf_sale_txn_s.table_number
and tf_sale_txn_s.batch_number                                      = (select pk_batch_number from
bi4all_01.admin.ctl_batch_control where ctl_batch_control.batch_complete_flag = 0 )
```

The next step is to insert the rows that are new into the target table in the target database.

```
insert into bi4all_02.admin.tf_sale_txn
(
 dk_vm_sale_date
,dk_vm_sale_minute
,dk_vm_sub_campaign
,dk_vm_product
,dk_vm_customer
,pk_tf_sale_txn
,product_number
,customer_number
)
select
 tf_sale_txn_s.dk_vm_sale_date
,tf_sale_txn_s.dk_vm_sale_minute
,tf_sale_txn_s.dk_vm_sub_campaign
,tf_sale_txn_s.dk_vm_product
,tf_sale_txn_s.dk_vm_customer
,tf_sale_txn_s.pk_tf_sale_txn
,tf_sale_txn_s.product_number
,tf_sale_txn_s.customer_number

from bi4all_01.admin.tf_sale_txn tf_sale_txn_s

left join bi4all_02.admin.tf_sale_txn tf_sale_txn on 1=1
and tf_sale_txn.pk_tf_sale_txn                                          = tf_sale_txn_s.pk_tf_sale_txn
and tf_sale_txn.table_number                                            = tf_sale_txn_s.table_number

where 1=1
and tf_sale_txn.pk_tf_sale_txn                                          is null
and tf_sale_txn_s.batch_number                                          = (select pk_batch_number from
bi4all_01.admin.ctl_batch_control where ctl_batch_control.batch_complete_flag = 0 )
;
```

7.1.3. Two Step Processing with Help Key Tables

The two step processing with help keys configuration presumes that there is NO identical structure of the data warehouse in the staging area that contains all data that might be updated. The relationships between source system keys and generated integer keys are maintained in the help key tables redundantly so that they do not need to be dragged back from the data warehouse for testing.

Again, there are the same 5 sets of processing that is performed. In these samples we will cut out some of the redundant list of fields. We have included all fields in the samples above. The reader is asked to take it on faith that the sql is generated for each field as it has been above. By cutting out much of the list of fields we will produce a little less volume in the documentation.

1. Type 1 dimension table processing
2. Type 2 dimension table processing without field level sensitivity.
3. Type 2 dimension table processing with field level sensitivity.
4. Fact table processing without a key on the fact table.
5. Fact table processing with a key on the fact table.

7.1.3.1. Type 1 dimension table processing

The first step is to select data out of the source view and place it into the target _t1 temp table.

```
insert into bi4all_01.admin.z01_vm_address_01_t1
(
 pk_vm_address
,level_col
,dim_char_ky_fld
,vm_address_seq
,address_key_str
,drop_point_id
,address_line_1
,address_line_2
,address_line_3
,address_line_4
)
select
 z01_vm_address_01.z01_vm_address_01_seq
,'detail'
,z01_vm_address_01.z01_vm_address_01_key
,z01_vm_address_01.z01_vm_address_01_seq
,z01_vm_address_01.address_key_str
,z01_vm_address_01.drop_point_id
,z01_vm_address_01.address_line_1
,z01_vm_address_01.address_line_2
,z01_vm_address_01.address_line_3
,z01_vm_address_01.address_line_4
from bi4all_01.admin.z01_vm_address_01 z01_vm_address_01
```

The reader is encouraged to notice that the _k1 key help table table is used to recognise which rows are inserts and updates in this configuration. Therefore the update forward to the target table in the target database is performed as the next step to take the new data for the existing rows across to the target table in the target data warehouse.

```
update bi4all_02.admin.tm_address set
 level_col                                                    = z01_vm_address_01_k1_s.level_col
,dim_char_ky_fld                                              = z01_vm_address_01_k1_s.dim_char_ky_fld
,address_key_str                                              = z01_vm_address_01_t1_s.address_key_str
,drop_point_id                                                = z01_vm_address_01_t1_s.drop_point_id
,address_line_1                                               = z01_vm_address_01_t1_s.address_line_1
,address_line_2                                               = z01_vm_address_01_t1_s.address_line_2
,address_line_3                                               = z01_vm_address_01_t1_s.address_line_3
,address_line_4                                               = z01_vm_address_01_t1_s.address_line_4

from bi4all_01.admin.z01_vm_address_01_t1 z01_vm_address_01_t1_s

inner join bi4all_01.admin.z01_vm_address_01_k1 z01_vm_address_01_k1_s on 1=1
and z01_vm_address_01_t1_s.level_col                          = z01_vm_address_01_k1_s.level_col
and z01_vm_address_01_t1_s.dim_char_ky_fld                    = z01_vm_address_01_k1_s.dim_char_ky_fld
where 1=1
and tm_address.level_col                                      = z01_vm_address_01_k1_s.level_col
and tm_address.dim_char_ky_fld                                = z01_vm_address_01_k1_s.dim_char_ky_fld
;
```

The next step is to move the data forward to the target table that is found to be new due to the keys not being currently found in the key help table.

```
insert into bi4all_02.admin.tm_address
(
 pk_tm_address
,level_col
,dim_char_ky_fld
,tm_address_seq
,address_key_str
,drop_point_id
,address_line_1
,address_line_2
,address_line_3
,address_line_4
)
select
 z01_vm_address_01_t1.pk_vm_address
,z01_vm_address_01_t1.level_col
,z01_vm_address_01_t1.dim_char_ky_fld
,z01_vm_address_01_t1.vm_address_seq
,z01_vm_address_01_t1.address_key_str
,z01_vm_address_01_t1.drop_point_id
,z01_vm_address_01_t1.address_line_1
,z01_vm_address_01_t1.address_line_2
,z01_vm_address_01_t1.address_line_3
,z01_vm_address_01_t1.address_line_4

from bi4all_01.admin.z01_vm_address_01_t1 z01_vm_address_01_t1
left join bi4all_01.admin.z01_vm_address_01_k1 z01_vm_address_01_k1 on 1=1
and z01_vm_address_01_t1.level_col                                  = z01_vm_address_01_k1.level_col
and z01_vm_address_01_t1.dim_char_ky_fld                            = z01_vm_address_01_k1.dim_char_ky_fld

where 1=1
and z01_vm_address_01_k1.dim_char_ky_fld                            is null
;
```

Lastly, the keys from the _t1 table are propagated forward to the hlp keys table so that it now records all the keys that have been used.

```
insert into bi4all_01.admin.z01_vm_address_01_k1
(
 pk_vm_address
,level_col
,dim_char_ky_fld
,vm_address_seq
)
select
 z01_vm_address_01_t1.pk_vm_address
,z01_vm_address_01_t1.level_col
,z01_vm_address_01_t1.dim_char_ky_fld
,z01_vm_address_01_t1.vm_address_seq

from bi4all_01.admin.z01_vm_address_01_t1 z01_vm_address_01_t1

left join bi4all_01.admin.z01_vm_address_01_k1 z01_vm_address_01_k1 on 1=1
and z01_vm_address_01_t1.level_col                                  = z01_vm_address_01_k1.level_col
and z01_vm_address_01_t1.dim_char_ky_fld                            = z01_vm_address_01_k1.dim_char_ky_fld

where 1=1
and z01_vm_address_01_k1.dim_char_ky_fld                            is null
;
```

7.1.3.2. Type 2 dimension table processing without field level sensitivity.

The first step as usual is to move the data from the source view to the _t1 temporary table.

```
insert into bi4all_01.admin.z01_vm_product_01_t1
(
 pk_vm_product
,level_col
,dim_char_ky_fld
,vm_product_seq
,current_flag
,date_from
,date_to
,product_number
,sku_number
,barcode_num
,prod_name
)
select
 z01_vm_product_01.z01_vm_product_01_seq
,'detail'
,z01_vm_product_01.z01_vm_product_01_key
,z01_vm_product_01.z01_vm_product_01_seq
,1
,ctl_batch_control.batch_date
,ctl_month_control.high_value_date
,z01_vm_product_01.product_number
,z01_vm_product_01.sku_number
,z01_vm_product_01.barcode_num
,z01_vm_product_01.prod_name

from bi4all_01.admin.z01_vm_product_01 z01_vm_product_01
left outer join bi4all_01.admin.ctl_batch_control ctl_batch_control on 1=1
and ctl_batch_control.batch_complete_flag = 0
left outer join bi4all_01.admin.ctl_month_control ctl_month_control on 1=1
```

The next step is to update the target table to set the current flag and date to zero and the current batch date minus one day for the rows that have been found to be being updated and will therefore be closed due to this processing.

```
update bi4all_02.admin.tm_product set
 current_flag                                                    = 0
,date_to                                                         = ctl_batch_control.batch_date - 1
from bi4all_01.admin.z01_vm_product_01_t1 z01_vm_product_01_t1_s

left outer join bi4all_01.admin.ctl_batch_control ctl_batch_control on 1=1
and ctl_batch_control.batch_complete_flag = 0

inner join bi4all_01.admin.z01_vm_product_01_k1 z01_vm_product_01_k1_s on 1=1
and z01_vm_product_01_t1_s.level_col                             = z01_vm_product_01_k1_s.level_col
and z01_vm_product_01_t1_s.dim_char_ky_fld                       = z01_vm_product_01_k1_s.dim_char_ky_fld
and z01_vm_product_01_t1_s.current_flag                          = z01_vm_product_01_k1_s.current_flag

where 1=1
and tm_product.level_col                                         = z01_vm_product_01_t1_s.level_col
and tm_product.dim_char_ky_fld                                   = z01_vm_product_01_t1_s.dim_char_ky_fld
and tm_product.current_flag                                      = 1

;
```

The next step is to move all the rows from the _t1 table directly forward to the target table in the target database because all rows are assumed to be new rows because there is no field sensitivity in place.

```
insert into bi4all_02.admin.tm_product
(
 pk_tm_product
,level_col
,dim_char_ky_fld
,current_flag
,date_from
,date_to
,tm_product_seq
,product_number
,sku_number
,barcode_num
,prod_name
)
select
 z01_vm_product_01_t1_s.pk_vm_product
,z01_vm_product_01_t1_s.level_col
,z01_vm_product_01_t1_s.dim_char_ky_fld
,z01_vm_product_01_t1_s.current_flag
,z01_vm_product_01_t1_s.date_from
,z01_vm_product_01_t1_s.date_to
,z01_vm_product_01_t1_s.vm_product_seq
,z01_vm_product_01_t1_s.product_number
,z01_vm_product_01_t1_s.sku_number
,z01_vm_product_01_t1_s.barcode_num
,z01_vm_product_01_t1_s.prod_name
from bi4all_01.admin.z01_vm_product_01_t1 z01_vm_product_01_t1_s
;
```

The keys help table is then updated to turn off the rows that have been updated. It was decided that these keys, though now redundant, would not be deleted from the keys help table. It was decided that they would be redundantly stored in the keys help table permanently.

```
update bi4all_01.admin.z01_vm_product_01_k1 set
 current_flag                                                                    = 0
,date_to                                                                         = ctl_batch_control.batch_date - 1
from bi4all_01.admin.z01_vm_product_01_t1 z01_vm_product_01_t1
left outer join bi4all_01.admin.ctl_batch_control ctl_batch_control on 1=1
and ctl_batch_control.batch_complete_flag = 0
where 1=1
and z01_vm_product_01_k1.level_col                                               = z01_vm_product_01_t1.level_col
and z01_vm_product_01_k1.dim_char_ky_fld                                         = z01_vm_product_01_t1.dim_char_ky_fld
and z01_vm_product_01_k1.current_flag                                            = 1
;
```

The new keys in the key help table are then moved forward into the key help table from the _t1 temporary table.

```
insert into bi4all_01.admin.z01_vm_product_01_k1
(
 pk_vm_product
,level_col
,dim_char_ky_fld
,current_flag
,date_from
,date_to
,vm_product_seq
)
select
 z01_vm_product_01_t1.pk_vm_product
,z01_vm_product_01_t1.level_col
,z01_vm_product_01_t1.dim_char_ky_fld
,1
,ctl_batch_control.batch_date
,ctl_month_control.high_value_date
,z01_vm_product_01_t1.vm_product_seq

from bi4all_01.admin.z01_vm_product_01_t1 z01_vm_product_01_t1
left outer join bi4all_01.admin.ctl_batch_control ctl_batch_control on 1=1
and ctl_batch_control.batch_complete_flag = 0
left outer join bi4all_01.admin.ctl_month_control ctl_month_control on 1=1
;
```

7.1.3.3. Type 2 dimension table processing with field level sensitivity.

The first step as usual is to move the data from the souce view to the _t1 temporary table.

```
insert into bi4all_01.admin.z01_vm_product_01_t1
(
 pk_vm_product
,level_col
,dim_char_ky_fld
,vm_product_seq
,current_flag
,date_from
,date_to
,product_number
,sku_number
,barcode_num
,prod_name
)
select
 z01_vm_product_01.z01_vm_product_01_seq
,'detail'
,z01_vm_product_01.z01_vm_product_01_key
,z01_vm_product_01.z01_vm_product_01_seq
,1
,ctl_batch_control.batch_date
,ctl_month_control.high_value_date
,z01_vm_product_01.product_number
,z01_vm_product_01.sku_number
,z01_vm_product_01.barcode_num
,z01_vm_product_01.prod_name

from bi4all_01.admin.z01_vm_product_01 z01_vm_product_01
left outer join bi4all_01.admin.ctl_batch_control ctl_batch_control on 1=1
and ctl_batch_control.batch_complete_flag = 0
left outer join bi4all_01.admin.ctl_month_control ctl_month_control on 1=1
```

The next step is to update the target table to set the current flag and date to fields to zero and the batch date minus one day so that the records in the target table that will be inserted are closed. Notice the code for field level sensitivity is added to the end of the statements here.

```
update bi4all_02.admin.tm_product set
 current_flag                                                                    = 0
,date_to                                                                         = ctl_batch_control.batch_date - 1

from bi4all_01.admin.z01_vm_product_01_t1 z01_vm_product_01_t1

left outer join bi4all_01.admin.ctl_batch_control ctl_batch_control on 1=1
and ctl_batch_control.batch_complete_flag = 0

where 1=1
and tm_product.level_col                                                         = z01_vm_product_01_t1.level_col
and tm_product.dim_char_ky_fld                                                   = z01_vm_product_01_t1.dim_char_ky_fld
and tm_product.current_flag                                                      = 1
and
(
   not ( z01_vm_product_01_t1.sku_number = tm_product.sku_number or ( z01_vm_product_01_t1.sku_number is null and
tm_product.sku_number is null ))
or not ( z01_vm_product_01_t1.barcode_num = tm_product.barcode_num or ( z01_vm_product_01_t1.barcode_num is null and
tm_product.barcode_num is null ))
or not ( z01_vm_product_01_t1.prod_name = tm_product.prod_name or ( z01_vm_product_01_t1.prod_name is null and
tm_product.prod_name is null ))
or not ( z01_vm_product_01_t1.prod_sub_group = tm_product.prod_sub_group or ( z01_vm_product_01_t1.prod_sub_group is null
and tm_product.prod_sub_group is null ))
)
;
```

The next step is to update the rows in the target table where the _t1 temporary table and the rules for field level sensitivity tell you that the rows need to be updated with the data in the _t1 temporary working table.

```
update bi4all_02.admin.tm_product set
 level_col                                                              = z01_vm_product_01_t1_s.level_col
,dim_char_ky_fld                                                        = z01_vm_product_01_t1_s.dim_char_ky_fld
,current_flag                                                           = z01_vm_product_01_t1_s.current_flag
,date_from                                                              = z01_vm_product_01_t1_s.date_from
,date_to                                                                = z01_vm_product_01_t1_s.date_to
,product_number                                                         = z01_vm_product_01_t1_s.product_number
,sku_number                                                             = z01_vm_product_01_t1_s.sku_number
,barcode_num                                                            = z01_vm_product_01_t1_s.barcode_num
,prod_name                                                              = z01_vm_product_01_t1_s.prod_name

from bi4all_01.admin.z01_vm_product_01_t1 z01_vm_product_01_t1_s

where 1=1
and tm_product.level_col                                                = z01_vm_product_01_t1_s.level_col
and tm_product.dim_char_ky_fld                                          = z01_vm_product_01_t1_s.dim_char_ky_fld
and tm_product.current_flag                                             = 1
and ( z01_vm_product_01_t1_s.sku_number = tm_product.sku_number or ( z01_vm_product_01_t1_s.sku_number is null and
tm_product.sku_number is null ))
and ( z01_vm_product_01_t1_s.barcode_num = tm_product.barcode_num or ( z01_vm_product_01_t1_s.barcode_num is null and
tm_product.barcode_num is null ))
and ( z01_vm_product_01_t1_s.prod_name = tm_product.prod_name or ( z01_vm_product_01_t1_s.prod_name is null and
tm_product.prod_name is null ))
and ( z01_vm_product_01_t1_s.prod_sub_group = tm_product.prod_sub_group or ( z01_vm_product_01_t1_s.prod_sub_group is null
and tm_product.prod_sub_group is null ))
;
```

The next step is to move the rows that are indicated as being inserts into the target table in the target database.

```
insert into bi4all_02.admin.tm_product
(
 pk_tm_product
,level_col
,dim_char_ky_fld
,current_flag
,date_from
,date_to
,tm_product_seq
,product_number
,sku_number
,barcode_num
,prod_name
)
select
 z01_vm_product_01_t1_s.pk_vm_product
,z01_vm_product_01_t1_s.level_col
,z01_vm_product_01_t1_s.dim_char_ky_fld
,z01_vm_product_01_t1_s.current_flag
,z01_vm_product_01_t1_s.date_from
,z01_vm_product_01_t1_s.date_to
,z01_vm_product_01_t1_s.vm_product_seq
,z01_vm_product_01_t1_s.product_number
,z01_vm_product_01_t1_s.sku_number
,z01_vm_product_01_t1_s.barcode_num
,z01_vm_product_01_t1_s.prod_name

from bi4all_01.admin.z01_vm_product_01_t1 z01_vm_product_01_t1_s
left join bi4all_02.admin.tm_product tm_product on 1=1
and tm_product.level_col                                    = z01_vm_product_01_t1_s.level_col
and tm_product.dim_char_ky_fld                              = z01_vm_product_01_t1_s.dim_char_ky_fld
and tm_product.current_flag                                 = 1

where 1=1
and tm_product.level_col                                    is null
and tm_product.dim_char_ky_fld                              is null
and tm_product.current_flag                                 is null
;
```

The next step is to perform the update against the keys help table to close out the records where the row has been closed in the target table. It was decided that these rows would not be deleted and that they would be redundantly stored in the help keys table.

```
update bi4all_01.admin.z01_vm_product_01_k1 set
 current_flag                                                                    = 0
,date_to                                                                         = ctl_batch_control.batch_date - 1

from bi4all_01.admin.z01_vm_product_01_t1 z01_vm_product_01_t1
left outer join bi4all_01.admin.ctl_batch_control ctl_batch_control on 1=1
and ctl_batch_control.batch_complete_flag = 0

where 1=1
and z01_vm_product_01_k1.level_col                                               = z01_vm_product_01_t1.level_col
and z01_vm_product_01_k1.dim_char_ky_fld                                         = z01_vm_product_01_t1.dim_char_ky_fld
and z01_vm_product_01_k1.current_flag                                            = 1
and
(
   not ( z01_vm_product_01_t1.sku_number = z01_vm_product_01_k1.sku_number or ( z01_vm_product_01_t1.sku_number is null
and z01_vm_product_01_k1.sku_number is null ))
or not ( z01_vm_product_01_t1.barcode_num = z01_vm_product_01_k1.barcode_num or ( z01_vm_product_01_t1.barcode_num is null
and z01_vm_product_01_k1.barcode_num is null ))
or not ( z01_vm_product_01_t1.prod_name = z01_vm_product_01_k1.prod_name or ( z01_vm_product_01_t1.prod_name is null and
z01_vm_product_01_k1.prod_name is null ))
or not ( z01_vm_product_01_t1.prod_sub_group = z01_vm_product_01_k1.prod_sub_group or (
z01_vm_product_01_t1.prod_sub_group is null and z01_vm_product_01_k1.prod_sub_group is null ))
)
;
```

The next step is to insert the new rows for key relationships into the help keys table.

```
insert into bi4all_01.admin.z01_vm_product_01_k1
(
 pk_vm_product
,level_col
,dim_char_ky_fld
,current_flag
,date_from
,date_to
,vm_product_seq
,sku_number
,barcode_num
,prod_name
,prod_sub_group
)
select
 z01_vm_product_01_t1.pk_vm_product
,z01_vm_product_01_t1.level_col
,z01_vm_product_01_t1.dim_char_ky_fld
,1
,z01_vm_product_01_t1.date_from
,z01_vm_product_01_t1.date_to
,z01_vm_product_01_t1.vm_product_seq
,z01_vm_product_01_t1.sku_number
,z01_vm_product_01_t1.barcode_num
,z01_vm_product_01_t1.prod_name
,z01_vm_product_01_t1.prod_sub_group

from bi4all_01.admin.z01_vm_product_01_t1 z01_vm_product_01_t1

left join bi4all_01.admin.z01_vm_product_01_k1 z01_vm_product_01_k1 on 1=1
and z01_vm_product_01_k1.level_col                                    = z01_vm_product_01_t1.level_col
and z01_vm_product_01_k1.dim_char_ky_fld                              = z01_vm_product_01_t1.dim_char_ky_fld
and z01_vm_product_01_k1.current_flag                                 = 1
where 1=1
and z01_vm_product_01_k1.dim_char_ky_fld                              is null
;
```

7.1.3.4. Fact table processing without a key on the fact table.

When there are no keys on the fact table there is no difference in the sql processing used for fact tables as from the one step processing.

7.1.3.5. Fact table processing with a key on the fact table.

The first step asa per usual is to place the new records into the _t1 temp table, Please note that we have cut many of the columns out for brevity.

```
insert into bi4all_01.admin.z01_vf_sale_txn_01_t1
(
 dk_vm_sale_date
,dk_vm_sale_minute
,dk_vm_sub_campaign
,dk_vm_product
,dk_vm_customer
,pk_vf_sale_txn
,product_number
,customer_number
)
select
 nvl(dk_zvm_sale_date, 0 )
,nvl(dk_zvm_sale_minute, 0 )
,nvl(dk_zvm_sub_campaign, 0 )
,nvl(dk_zvm_product, 0 )
,nvl(dk_zvm_customer, 0 )
,ctl_sequence_number
,product_number
,customer_number
from bi4all_01.admin.z01_vf_sale_txn_01 z01_vf_sale_txn_01
left join bi4all_01.admin.zvm_sale_date_at3   zvm_sale_date_at3
on z01_vf_sale_txn_01.char_key_zvm_sale_date = zvm_sale_date_at3.pk_dim_char_ky_fld
left join bi4all_01.admin.zvm_sale_minute_at3   zvm_sale_minute_at3
on z01_vf_sale_txn_01.char_key_zvm_sale_minute = zvm_sale_minute_at3.pk_dim_char_ky_fld
left join bi4all_01.admin.zvm_sub_campaign_at3   zvm_sub_campaign_at3
on z01_vf_sale_txn_01.char_key_zvm_sub_campaign = zvm_sub_campaign_at3.pk_dim_char_ky_fld
left join bi4all_01.admin.zvm_product_at3   zvm_product_at3
on z01_vf_sale_txn_01.char_key_zvm_product = zvm_product_at3.pk_dim_char_ky_fld
left join bi4all_01.admin.zvm_customer_at3   zvm_customer_at3
on z01_vf_sale_txn_01.char_key_zvm_customer = zvm_customer_at3.pk_dim_char_ky_fld
```

The next step is to perform the update of the target transaction table for those rows where the key help table indicates that the row will be found in the target table.

```
update bi4all_02.admin.tf_sale_txn set
 dk_vm_sale_date                                                    = z01_vf_sale_txn_01_t1.dk_vm_sale_date
,dk_vm_sale_minute                                                  = z01_vf_sale_txn_01_t1.dk_vm_sale_minute
,dk_vm_sub_campaign                                                 = z01_vf_sale_txn_01_t1.dk_vm_sub_campaign
,dk_vm_product                                                      = z01_vf_sale_txn_01_t1.dk_vm_product
,dk_vm_customer                                                     = z01_vf_sale_txn_01_t1.dk_vm_customer
,product_number                                                     = z01_vf_sale_txn_01_t1.product_number
,customer_number                                                    = z01_vf_sale_txn_01_t1.customer_number
,card_type                                                          = z01_vf_sale_txn_01_t1.card_type
,card_number                                                        = z01_vf_sale_txn_01_t1.card_number

from bi4all_01.admin.z01_vf_sale_txn_01_t1 z01_vf_sale_txn_01_t1

inner join bi4all_01.admin.z01_vf_sale_txn_01_k1 z01_vf_sale_txn_01_k1 on 1=1
and z01_vf_sale_txn_01_t1.product_number                            = z01_vf_sale_txn_01_k1.product_number
and z01_vf_sale_txn_01_t1.customer_number                           = z01_vf_sale_txn_01_k1.customer_number
and z01_vf_sale_txn_01_t1.account_number                            = z01_vf_sale_txn_01_k1.account_number
and z01_vf_sale_txn_01_t1.transaction_type                          = z01_vf_sale_txn_01_k1.transaction_type
and z01_vf_sale_txn_01_t1.sale_tstamp                               = z01_vf_sale_txn_01_k1.sale_tstamp

where 1=1
and tf_sale_txn.product_number                                      = z01_vf_sale_txn_01_t1.product_number
and tf_sale_txn.customer_number                                     = z01_vf_sale_txn_01_t1.customer_number
and tf_sale_txn.account_number                                      = z01_vf_sale_txn_01_t1.account_number
and tf_sale_txn.transaction_type                                    = z01_vf_sale_txn_01_t1.transaction_type
and tf_sale_txn.sale_tstamp                                         = z01_vf_sale_txn_01_t1.sale_tstamp
;
```

The next step is to insert the rows into the target fact table in the target database where the keys help table indicates that the row will not be found in the target table.

```
insert into bi4all_02.admin.tf_sale_txn
(
 dk_vm_sale_date
,dk_vm_sale_minute
,dk_vm_sub_campaign
,dk_vm_product
,dk_vm_customer
,pk_tf_sale_txn
,product_number
,customer_number
)
select
 z01_vf_sale_txn_01_t1.dk_vm_sale_date
,z01_vf_sale_txn_01_t1.dk_vm_sale_minute
,z01_vf_sale_txn_01_t1.dk_vm_sub_campaign
,z01_vf_sale_txn_01_t1.dk_vm_product
,z01_vf_sale_txn_01_t1.dk_vm_customer
,z01_vf_sale_txn_01_t1.pk_vf_sale_txn
,z01_vf_sale_txn_01_t1.product_number
,z01_vf_sale_txn_01_t1.customer_number

from bi4all_01.admin.z01_vf_sale_txn_01_t1 z01_vf_sale_txn_01_t1

left join bi4all_01.admin.z01_vf_sale_txn_01_k1 z01_vf_sale_txn_01_k1 on 1=1
and z01_vf_sale_txn_01_t1.product_number                = z01_vf_sale_txn_01_k1.product_number
and z01_vf_sale_txn_01_t1.customer_number               = z01_vf_sale_txn_01_k1.customer_number
and z01_vf_sale_txn_01_t1.account_number                = z01_vf_sale_txn_01_k1.account_number
and z01_vf_sale_txn_01_t1.transaction_type              = z01_vf_sale_txn_01_k1.transaction_type
and z01_vf_sale_txn_01_t1.sale_tstamp                   = z01_vf_sale_txn_01_k1.sale_tstamp

where 1=1
and z01_vf_sale_txn_01_k1.product_number                is null
and z01_vf_sale_txn_01_k1.customer_number               is null
and z01_vf_sale_txn_01_k1.account_number                is null
and z01_vf_sale_txn_01_k1.transaction_type              is null
and z01_vf_sale_txn_01_k1.sale_tstamp                   is null
```

The next step is to place the new source field keys and integer key relationship into the keys help table so that for the next batch run the ETL subsystem knows that these relationships exist in the target table.

```
insert into bi4all_01.admin.z01_vf_sale_txn_01_k1
(
 pk_tf_sale_txn
,product_number
,customer_number
,account_number
,transaction_type
,sale_tstamp
)
select
 z01_vf_sale_txn_01_t1.pk_vf_sale_txn
,z01_vf_sale_txn_01_t1.product_number
,z01_vf_sale_txn_01_t1.customer_number
,z01_vf_sale_txn_01_t1.account_number
,z01_vf_sale_txn_01_t1.transaction_type
,z01_vf_sale_txn_01_t1.sale_tstamp

from bi4all_01.admin.z01_vf_sale_txn_01_t1 z01_vf_sale_txn_01_t1

left join bi4all_01.admin.z01_vf_sale_txn_01_k1 z01_vf_sale_txn_01_k1 on 1=1
and z01_vf_sale_txn_01_k1.product_number                                    = z01_vf_sale_txn_01_t1.product_number
and z01_vf_sale_txn_01_k1.customer_number                                   = z01_vf_sale_txn_01_t1.customer_number
and z01_vf_sale_txn_01_k1.account_number                                    = z01_vf_sale_txn_01_t1.account_number
and z01_vf_sale_txn_01_k1.transaction_type                                  = z01_vf_sale_txn_01_t1.transaction_type
and z01_vf_sale_txn_01_k1.sale_tstamp                                       = z01_vf_sale_txn_01_t1.sale_tstamp

where 1=1
and z01_vf_sale_txn_01_k1.product_number is null
and z01_vf_sale_txn_01_k1.customer_number is null
and z01_vf_sale_txn_01_k1.account_number is null
and z01_vf_sale_txn_01_k1.transaction_type is null
and z01_vf_sale_txn_01_k1.sale_tstamp is null
;
```

This page intentionally left blank.