TSEEQ: The Structured ETL Engine for Qlik

Version 3.1

Qlik Consulting Services

Document Author:	Jeff R. Robbins
Document Date:	December 24, 2017
Email:	Jeff.Robbins@qlik.com; jr@qlikperf.com

Ask, and it shall be given you; seek, and ye shall find; knock, and it shall be opened unto you.

(Matthew 7:7, King James Version)

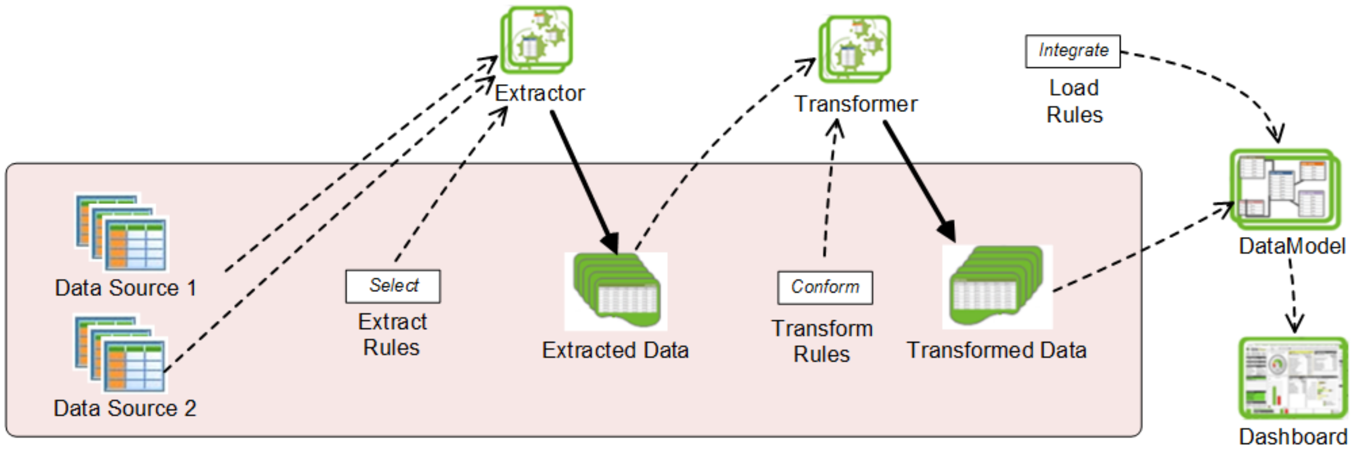
Contents

Please note that ***TSEEQ*** was formerly spelled ***SEEQ*** (no initial “T”); some screen shots, file names and variables may still reflect this older spelling in the near term.

Introduction

TSEEQ (pronounced “seek”), The Structured ETL Engine for Qlik, implements centralized management of Extract, Transform and Load (ETL) operations that provide data to QlikView and Qlik Sense applications. Primary benefits of ***TSEEQ*** are:

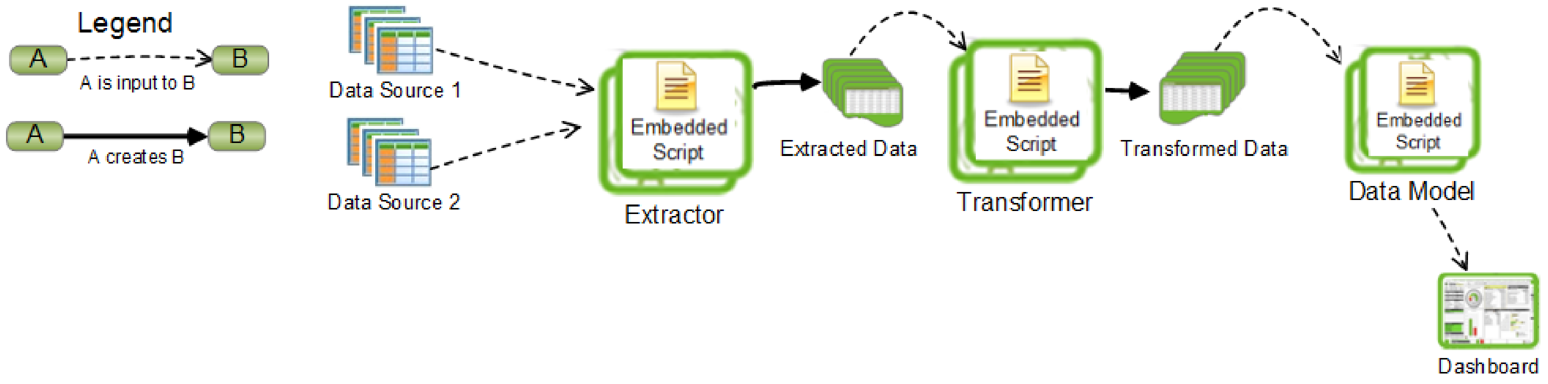
- **Governance:** ETL operations are defined in externalized (and therefore readily manageable) rule sets.
- **Self-Service:** Business users may easily define and modify ETL operations in sandbox environments.
- **Performance:** A profiler enables efficient ETL execution.
- **Migration:** ETL rule sets for QlikView can be used without modification for Qlik Sense (and vice versa).



- **Integration:** Data from multiple sources is conformed and integrated to create a consolidated data model.
- **Productivity:** A common code base promotes reuse and streamlines ETL development.

TSEEQ In Comparison to Traditional Embedded Scripting

The distinguishing characteristic of ***TSEEQ*** versus Traditional Embedding Scripting (**TES**) is that in ***TSEEQ***, externalized rules (diagram above) provide a structured source of ETL control; in **TES**, free-form textual ETL script is embedded within Qlik application files (**QVWs** in QlikView and **QVFs** in Qlik Sense). We can conceptualize **TES** as shown in the diagram below:



Note: ***TSEEQ*** and **TES** are not mutually exclusive; a hybrid approach is useful in many cases.

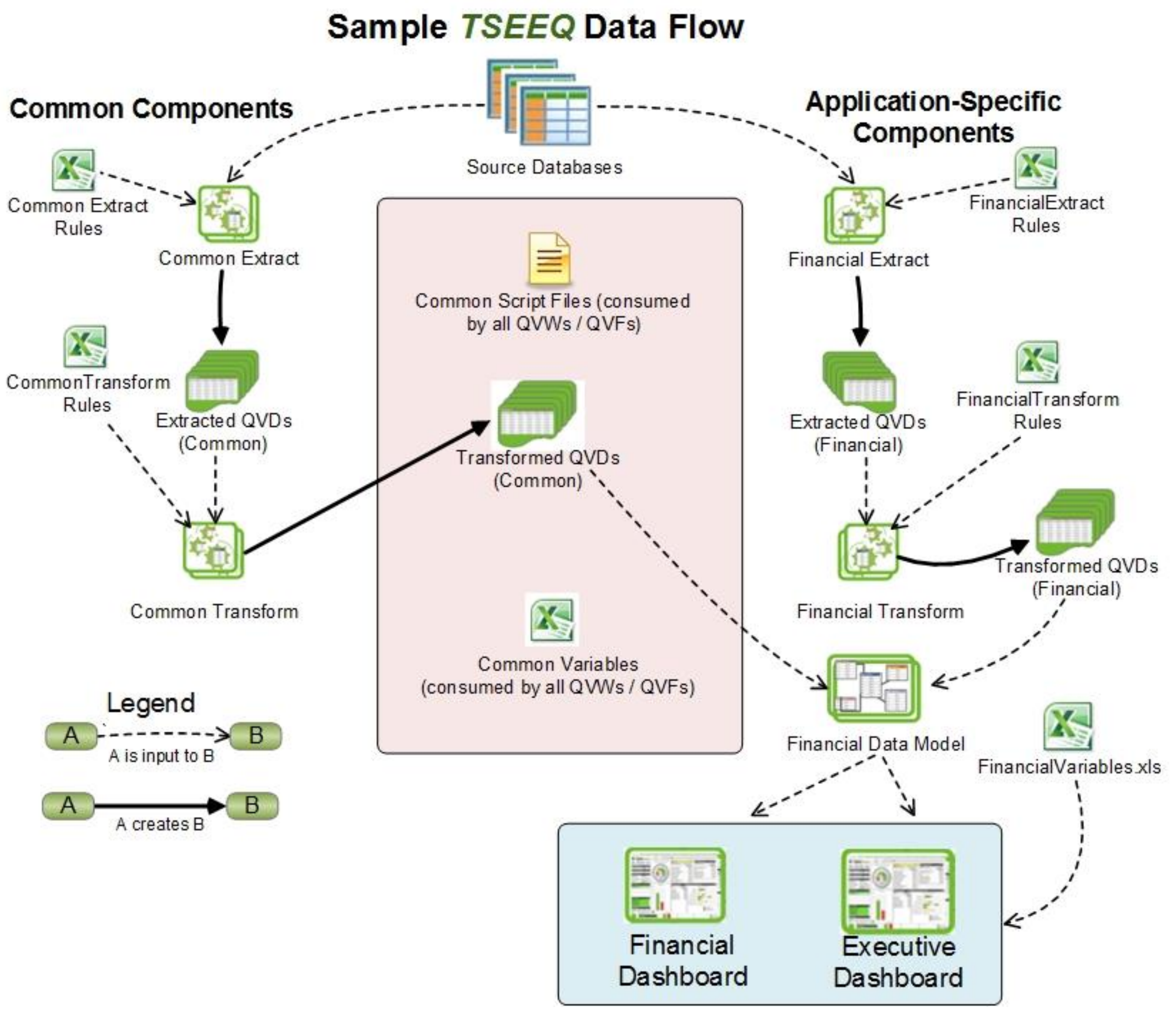
Technical Architecture

TSEEQ reads ETL rules from a metadata store and then automatically generates and executes Qlik script to build QVDs and QlikView or Qlik Sense data models.

1. ***TSEEQ*** implements the concept of ***common*** components, including both data and code, that are shared among multiple applications. Common components from a typical ***TSEEQ*** deployment are shown in the left and center of the diagram at right.

2. ***TSEEQ*** does not provide a graphical drawing tool for data flows; rather, the Qlik Developer defines ETL rules in the tabular metadata store.
   a. The metadata store is by default a set of Excel spreadsheets; a relational database may be used instead of Excel.
   b. Despite the lack of a drawing tool, ***TSEEQ*** ETL rules are easily created by modifying sample rules provided in the ***TSEEQ*** *Sales Sample* (detailed next page).

3. Field transformation rules are Qlik expressions; ***TSEEQ*** is an abstraction layer on top of the Qlik scripting engine.

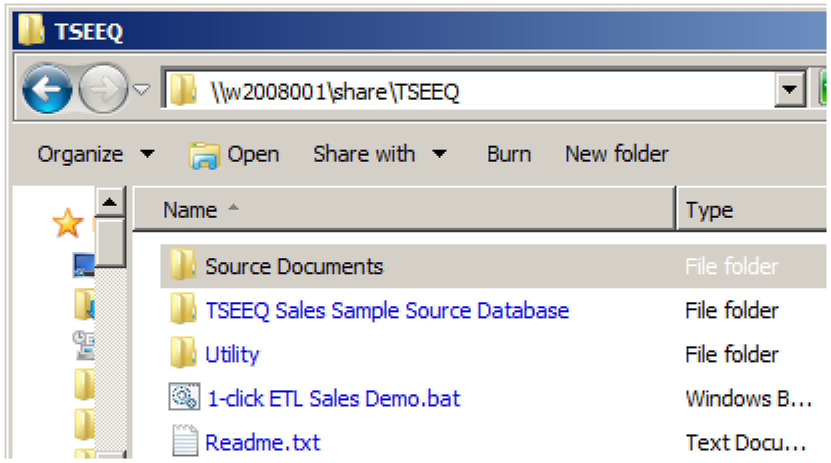


4. Since ***TSEEQ*** is a set of script routines executing in the context of a QVW or QVF, standard Publisher or Qlik Sense tasks are the scheduling mechanism.

Instructions for the *TSEEQ Sales Sample*:

The *TSEEQ Sales Sample*, contained within *TSEEQ.zip,* provides an end-to-end working example of a ***TSEEQ*** ETL flow that can be executed and modified for learning purposes, as well as serve as a template with which to implement ***TSEEQ*** ETL flows for additional applications.

File Extraction Instructions (Both Qlik Sense and QlikView)



TSEEQ

\\w2008001\share\TSEEQ

Name	Type
Source Documents	File folder
TSEEQ Sales Sample Source Database	File folder
Utility	File folder
1-click ETL Sales Demo.bat	Windows B...
Readme.txt	Text Docu...

1. Identify a network share to which your development team has read and write access.
   a. Note that for single developer scenarios (such as prototyping in a private sandbox using QlikView or Qlik Sense Desktop), a local path, ex *C:\TSEEQ*, can be used in lieu of a network share. However, the remainder of this document assumes the use of a network share.
   b. The screen shot at right shows a share named *\\w2008001\share*; your actual share name will likely differ. You can use a sub-folder within a share if preferred.
2. For the rest of this document, ~\ refers to the share or preferred sub-folder within a share that you identified.
3. Extract the attached ZIP file to ~\.
4. Under ~\, you should then see a **TSEEQ** folder with constituent sub-folders, as shown in the screen shot at right.

ETL Execution Instructions (QlikView)

1. For **QlikView**, to run the whole data flow for a Sales demo application, simply double-click the **1-click ETL - Sales Demo.bat** file that is in the top-level ~*TSEEQ* folder.
   a. The Extract, Transform and Load process for the *TSEEQ Sales Sample* will then automatically execute.
   b. (Hint: if prompted to with a Save As dialog, just click **Save** to overwrite the old file and click **Yes** when prompted if you want to replace the file.)

ETL Execution Instructions (Qlik Sense)

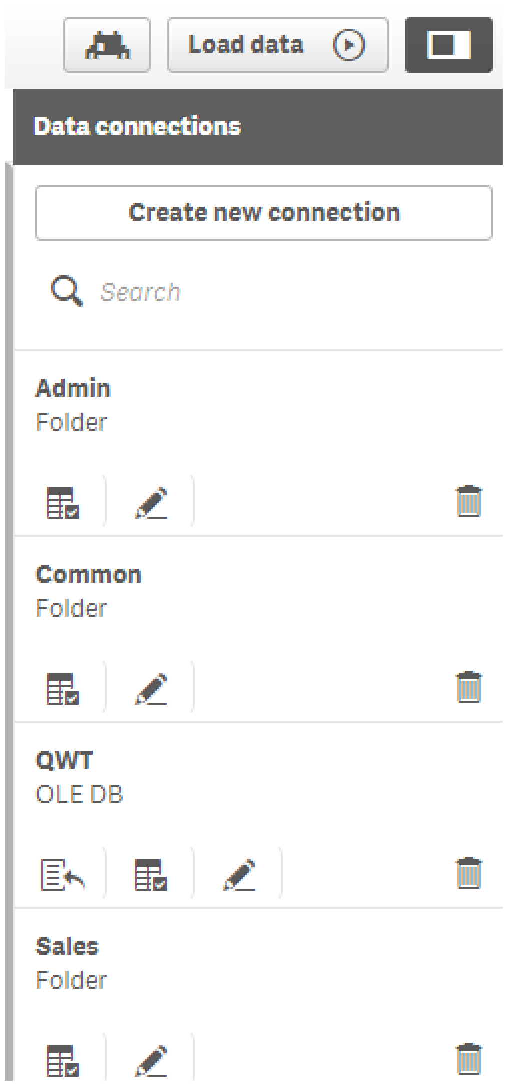
Below, you only need to complete Step 1 OR Step 2, depending on the Qlik Sense product you are using:

1. For **QlikSense Server**, use the QMC to import each of the 3 QVFs under *~\TSEEQ\Source Documents\TSEEQ Sales Sample*, from each of the step-specific sub-folders (*01_Extract*, *02_Transform* and *03_Load*).

2. For **Qlik Sense Desktop**, you can either:
   a. Copy the *~\TSEEQ* folder to *c:\users\<user id>\Documents\Qlik\Sense\Apps*.
   OR
   b. Point *c:\users\<user id>\user folder>\Documents\Qlik\Sense\Settings.ini* to *~\TSEEQ*.
      - Please see this link for instructions: https://community.qlik.com/thread/158503

   OR
   c. Use a junction point to re-direct *c:\users\<user id>\Documents\Qlik\Sense\Apps* to *~\TSEEQ*.
      - Please see this link for instructions: https://en.wikipedia.org/wiki/NTFS_junction_point#Creating_or_deleting_a_junction_point

3. Once you have completed step 1 (Qlik Sense Server) or step 2 (Qlik Sense Desktop), verify that the required data connections exist by simply opening the **Sales Extract** app from the **Hub**. Data connections will then show in the script **Data Load Editor** (screen shot at right).

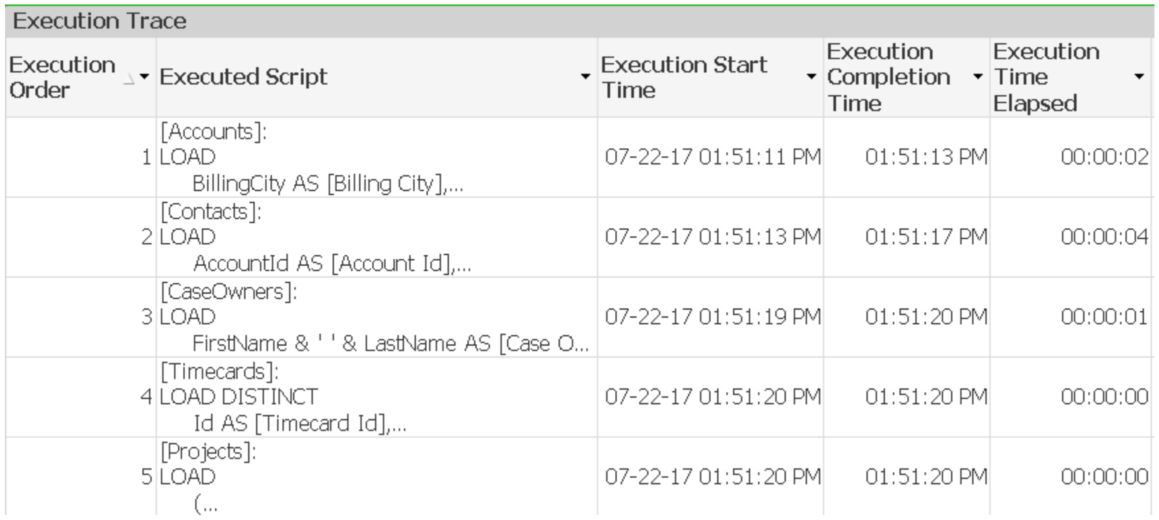


4. Next, edit the data connections. (For Qlik Sense Server, editing the connections from the **QMC** is preferred over editing the connections from the **Data Load Editor).** Edit as follows:
   a. **Admin**: modify it to point to *~\TSEEQ\Source Documents**Admin***.
   b. **Sales**: modify it to point to *~\TSEEQ\Source Documents**TSEEQ Sales Sample***.
   c. **Common:** modify it to point to *~\TSEEQ\Source Documents**Common***.
   d. **QWT:** modify it to point to *~\TSEEQ Sales Sample Source SourceDatabase**QWT**.mdb.*

5. Open the **Data Load Editor** and press **load data,** for each of the following apps in sequence:
   1. *Sales Extract*
   2. *Sales Transform*
   3. *Sales DataModel*

Validating Successful ETL Execution

Successful ***TSEEQ*** execution can be validated within both QlikView and Qlik Sense by viewing the **Performance Profile** and **Execution Trace** tables within the ETL QVWs and QVFs. The screen shot below shows the **Execution Trace** from the included **Sales Transform** QVW for QlikView (a similar table is provided in the **Sales Transform** QVF for Qlik Sense). Since the **Execution Trace** shown below is from a sandbox environment with a limited data set, all times shown are four seconds or less; times in a production environment with larger data sets may be higher.

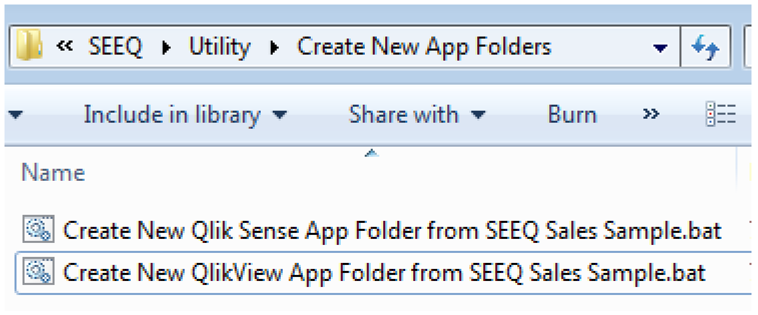


Execution Trace

Execution Order	Executed Script	Execution Start Time	Execution Completion Time	Execution Time Elapsed
1	[Accounts]: LOAD BillingCity AS [Billing City],...	07-22-17 01:51:11 PM	01:51:13 PM	00:00:02
2	[Contacts]: LOAD AccountId AS [Account Id],...	07-22-17 01:51:13 PM	01:51:17 PM	00:00:04
3	[CaseOwners]: LOAD FirstName & ' ' & LastName AS [Case O...	07-22-17 01:51:19 PM	01:51:20 PM	00:00:01
4	[Timecards]: LOAD DISTINCT Id AS [Timecard Id],...	07-22-17 01:51:20 PM	01:51:20 PM	00:00:00
5	[Projects]: LOAD (...	07-22-17 01:51:20 PM	01:51:20 PM	00:00:00

Creating New App Folders from the ***TSEEQ Sales Sample***:

After you have successfully run and validated the end-to-end ETL flow provided within the *TSEEQ Sales Sample* as discussed on the prior pages, you may wish to replicate the folder structure of the *TSEEQ Sales Sample* for use with additional applications[1]. To facilitate this folder structure replication, you may run the batch files under *~\TSEEQ\Utility\**Create New App Folders***.



Create New Qlik Sense App Folder from SEEQ Sales Sample.bat
Create New QlikView App Folder from SEEQ Sales Sample.bat

Please note that:

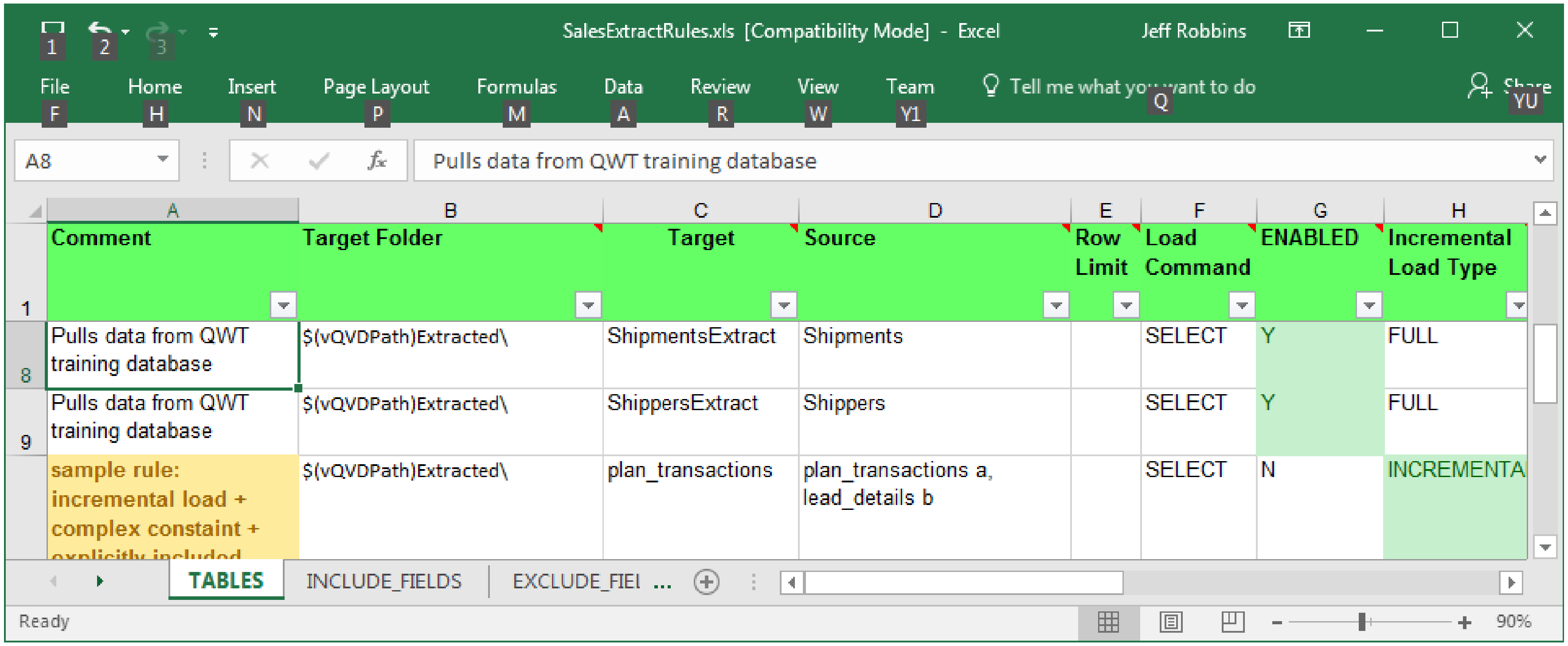
1. ***TSEEQ*** does not require any specific folder structure be used; ***TSEEQ*** can be adapted to an existing folder structure if one is already in place.
2. With Qlik Sense Server, the folder structure is less relevant than with QlikView; Qlik Sense Server stores all "apps" (QVFs) in a repository and therefore the concept of a folders does not apply for QVFs (with the one potential exception being a BINARY LOAD of a QVF from a folder connection). In Qlik Sense, the concept of folders only applies to data files (such as XLSX and QVD) and externalized script that is brought in with an **INCLUDE** statement. fix

[1] A later section of this document defines terms such as "app" and "application" a bit more explicitly.

Enabling Additional Rules & The GENERATE ONLY Mode

Please note the included sample rules files under the following folder: ~*Source Documents\TSEEQ Sales Sample\ETL_Rules*.

By default, ***TSEEQ*** generates and executes ETL script for those rules where **ENABLED** is set to **Y**. The currently enabled rules are those which operate upon the included sample database (~*TSEEQ Sales Sample Source Database\QWT.mdb*).


SalesExtractRules.xls [Compatibility Mode] - Excel
Jeff Robbins
File
Home
Insert
Page Layout
Formulas
Data
Review
View
Team
Tell me what you want to do
Share
A8
Pulls data from QWT training database
TABLES
INCLUDE_FIELDS
EXCLUDE_FIEI ...
Ready
90%

	Comment	Target Folder	Target	Source	Row Limit	Load Command	ENABLED	Incremental Load Type
8	Pulls data from QWT training database	$(vQVDPath)Extracted\	ShipmentsExtract	Shipments		SELECT	Y	FULL
9	Pulls data from QWT training database	$(vQVDPath)Extracted\	ShippersExtract	Shippers		SELECT	Y	FULL
	sample rule: incremental load + complex constaint + explicitly included	$(vQVDPath)Extracted\	plan_transactions	plan_transactions a, lead_details b		SELECT	N	INCREMENTA

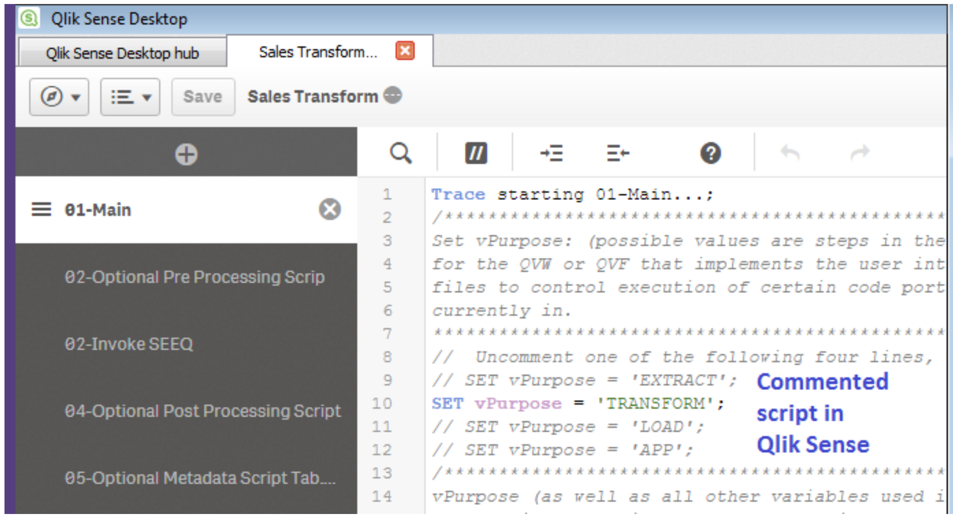
TSEEQ will not generate and execute ETL script rules where **ENABLED** is set to *N*, since a corresponding sample database is not included in the ***TSEEQ*** zip file. However, you may optionally enable those rules by

1. changing **ENABLED** to **Y**, and then
2. uncommenting the following line in on the *01-Main* script tab: `// SET vTSEEQ_Mode = 'GENERATE ONLY';`

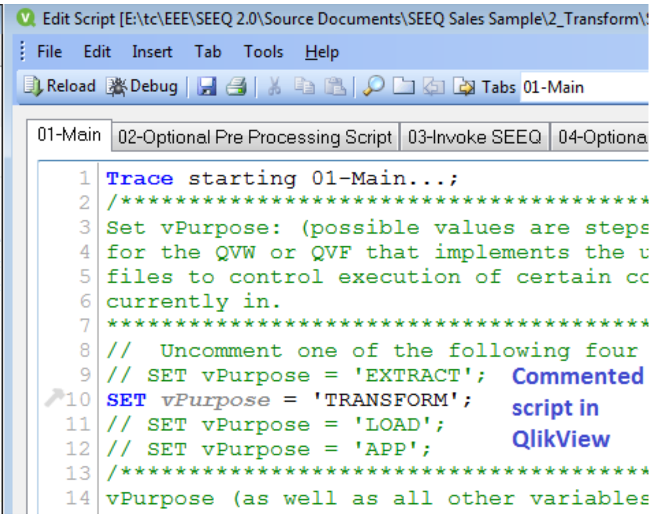
TSEEQ will then generate, but **not e**xecute, ETL script, thereby allowing you to see how ***TSEEQ*** creates ETL script for a wider variety of ETL rules.

Documentation Within in the ETL QVFs and QVWs

As shown in the screen shot, the script within each of the ***TSEEQ*** *Sales Sample* QVFs (Qlik Sense) and QVWs (Qlik View) is extensively commented (comment lines outnumber actual code lines by over 3-to-1). Information from those intra-QV* comments is not replicated in its entirety within this document. As such, developers are referred to the intra-QV* comments for additional information on the script within the QV* files.



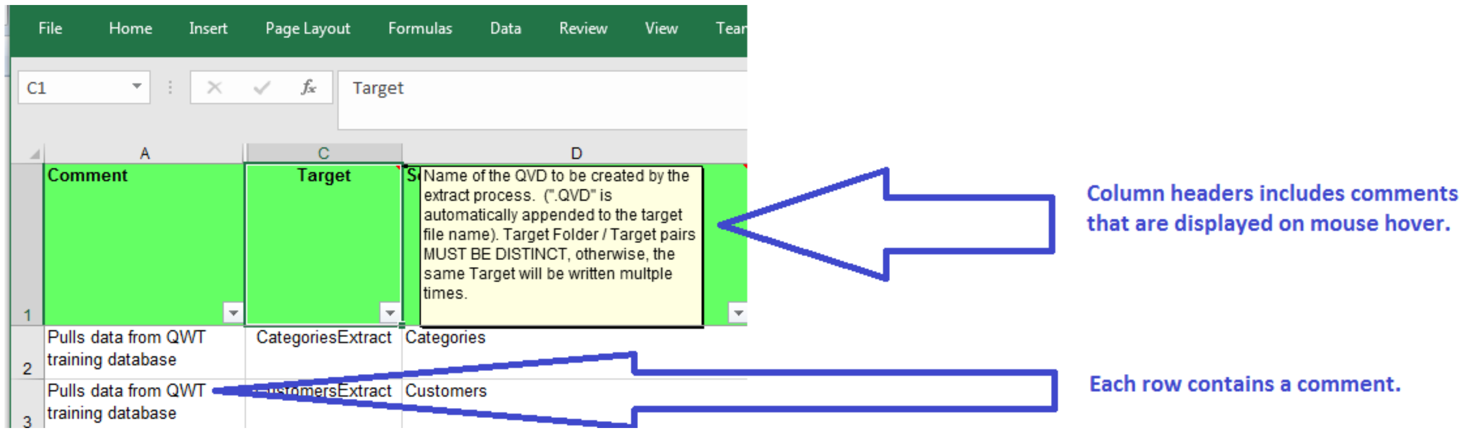
```
Trace starting 01-Main...;
/*******************************************
Set vPurpose: (possible values are steps in the
for the QVW or QVF that implements the user int
files to control execution of certain code port
currently in.
*******************************************
//  Uncomment one of the following four lines,
// SET vPurpose = 'EXTRACT';
SET vPurpose = 'TRANSFORM';
// SET vPurpose = 'LOAD';
// SET vPurpose = 'APP';
/*******************************************
vPurpose (as well as all other variables used i
```


Edit Script [E:\tc\EEE\SEEQ 2.0\Source Documents\SEEQ Sales Sample\2_Transform\
File
Edit
Insert
Tab
Tools
Help
Reload
Debug
Tabs 01-Main
01-Main
02-Optional Pre Processing Script
03-Invoke SEEQ
04-Optiona
Commented
script in
QlikView

```
Trace starting 01-Main...;
/**************************************
Set vPurpose: (possible values are steps
for the QVW or QVF that implements the u
files to control execution of certain cc
currently in.
**************************************
//  Uncomment one of the following four
// SET vPurpose = 'EXTRACT';
SET vPurpose = 'TRANSFORM';
// SET vPurpose = 'LOAD';
// SET vPurpose = 'APP';
/**************************************
vPurpose (as well as all other variables
```

Documentation Within in the XLS Files

As shown in the screen shot, each column and row within the XLS files in the ***TSEEQ*** *Sales Sample* contains embedded documentation. This documentation is not replicated in its entirety within this document. As such, Qlik developers are referred to the intra-XLS documentation within the ***TSEEQ*** *Sales Sample*:


File
Home
Insert
Page Layout
Formulas
Data
Review
View
Tear
C1
Target
Column headers includes comments that are displayed on mouse hover.
Each row contains a comment.

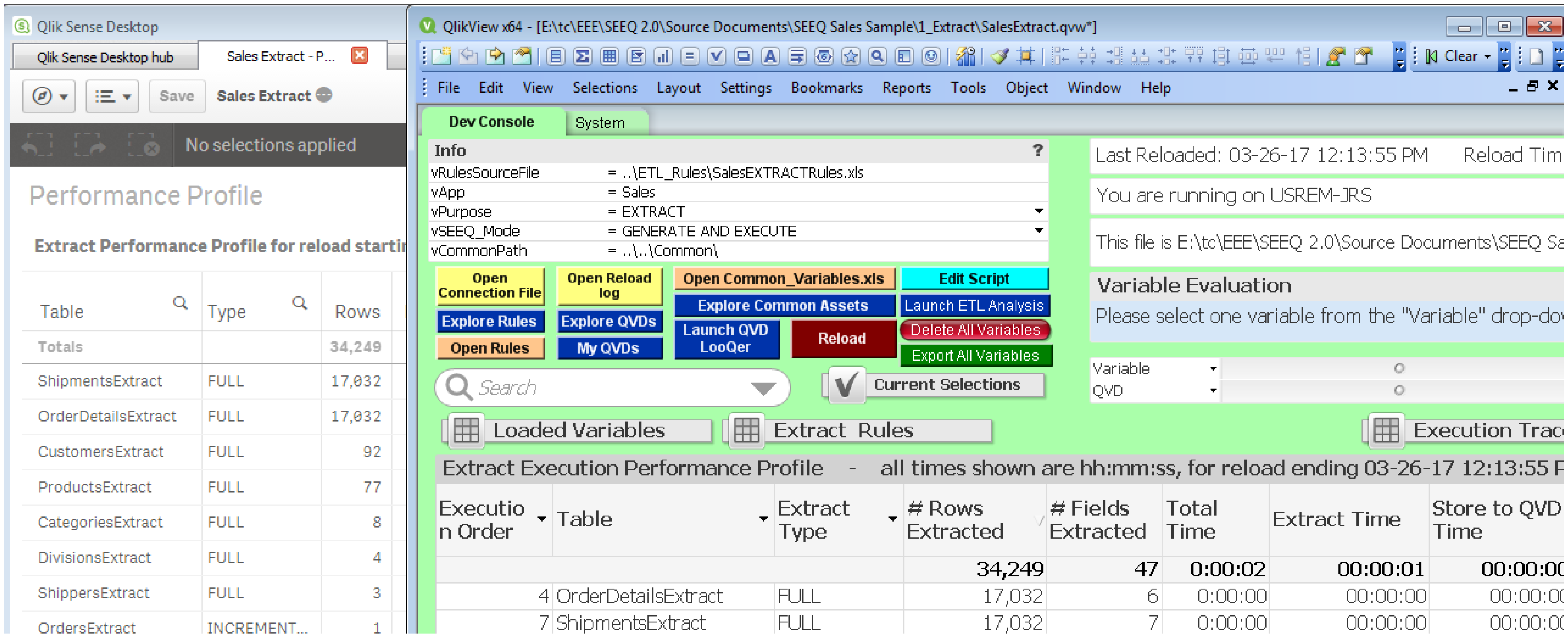
	A	C	D
1	Comment	Target	S
2	Pulls data from QWT training database	CategoriesExtract	Categories
3	Pulls data from QWT training database	CustomersExtract	Customers

Name of the QVD to be created by the extract process. (".QVD" is automatically appended to the target file name). Target Folder / Target pairs MUST BE DISTINCT, otherwise, the same Target will be written multple times.

ETL Statistics (Most Recent Execution)

In the ***TSEEQ** Sales Sample*, ETL Statistics for the most recent execution shown in within the respective **Extract, Transform and Load** QVFs (Qlik Sense) and QVWs (QlikView).

The screen shot immediately below shows a view of the Qlik Sense statistics on the left and the QlikView statistics on the right. Since the **Performance Profiles** shown below is from a sandbox environment with a limited data set, all times shown are two seconds or less; times in a production environment with larger data sets may be higher.



Performance Profile

Extract Performance Profile for reload starti

Table	Type	Rows
Totals		34,249
ShipmentsExtract	FULL	17,032
OrderDetailsExtract	FULL	17,032
CustomersExtract	FULL	92
ProductsExtract	FULL	77
CategoriesExtract	FULL	8
DivisionsExtract	FULL	4
ShippersExtract	FULL	3
OrdersExtract	INCREMENT...	1

Info

vRulesSourceFile	= ..\ETL_Rules\SalesEXTRACTRules.xls
vApp	= Sales
vPurpose	= EXTRACT
vSEEQ_Mode	= GENERATE AND EXECUTE
vCommonPath	= ..\..\Common\

Last Reloaded: 03-26-17 12:13:55 PM Reload Tim

You are running on USREM-JRS

This file is E:\tc\EEE\SEEQ 2.0\Source Documents\SEEQ Sa

Variable Evaluation

Please select one variable from the "Variable" drop-do

Variable

QVD

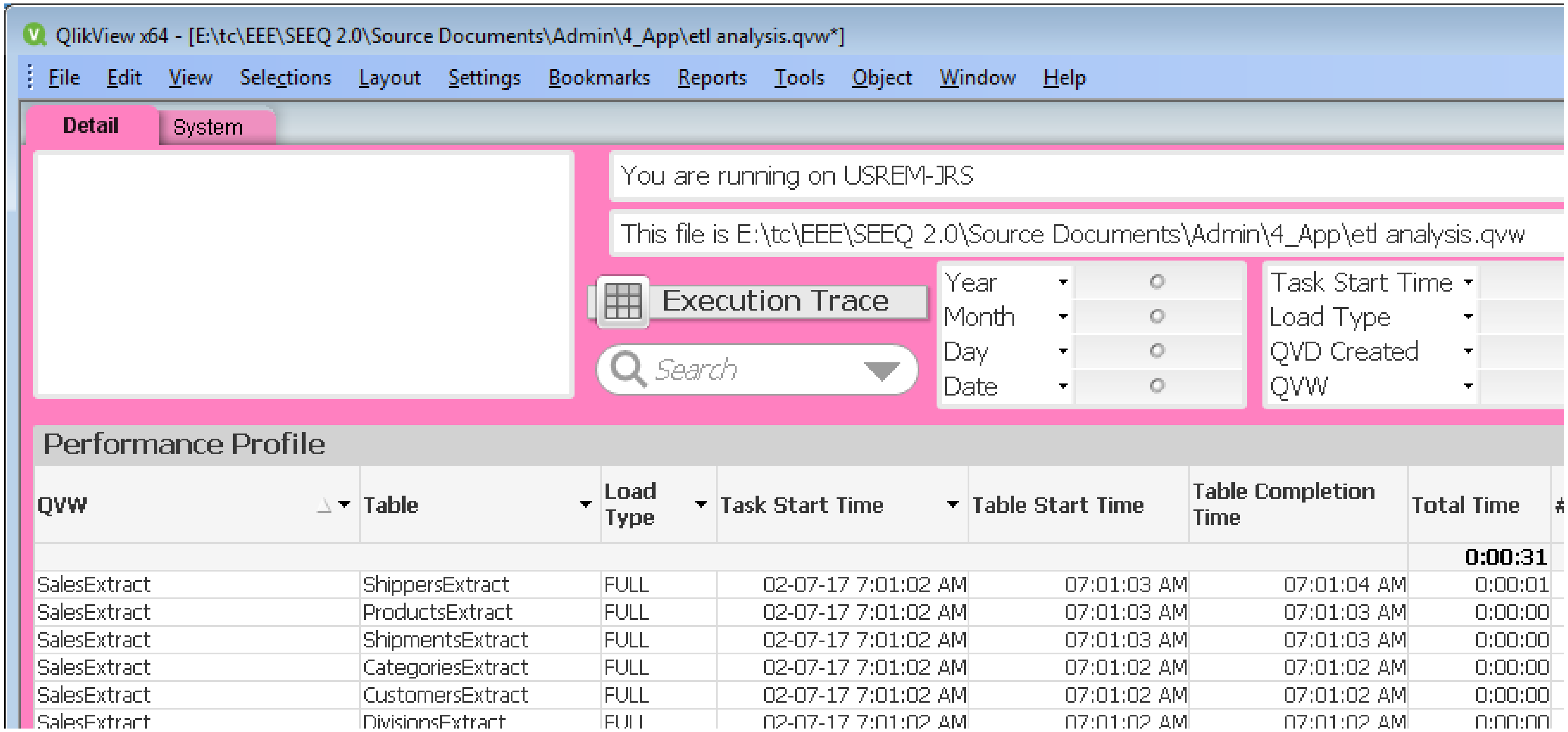
Extract Execution Performance Profile - all times shown are hh:mm:ss, for reload ending 03-26-17 12:13:55 P

Execution Order	Table	Extract Type	# Rows Extracted	# Fields Extracted	Total Time	Extract Time	Store to QVD Time
			34,249	47	0:00:02	00:00:01	00:00:0
4	OrderDetailsExtract	FULL	17,032	6	0:00:00	00:00:00	00:00:0
7	ShipmentsExtract	FULL	17,032	7	0:00:00	00:00:00	00:00:0

ETL Statistics (Historical)

ETL statistics are for the past 100,000 ETL operations are reported in the **~\Admin\4_App\ETL Analysis QVW** and **QVF.**

The screen shot shown below is from the **ETL Analysis** QVW QlikView; a similar sheet is provided in Qlik Sense by the **ETL Analysis** QVF.


QlikView x64 - [E:\tc\EEE\SEEQ 2.0\Source Documents\Admin\4_App\etl analysis.qvw*]
File Edit View Selections Layout Settings Bookmarks Reports Tools Object Window Help
Detail
System
Execution Trace
Search
Year
Month
Day
Date
Task Start Time
Load Type
QVD Created
QVW

You are running on USREM-JRS

This file is E:\tc\EEE\SEEQ 2.0\Source Documents\Admin\4_App\etl analysis.qvw

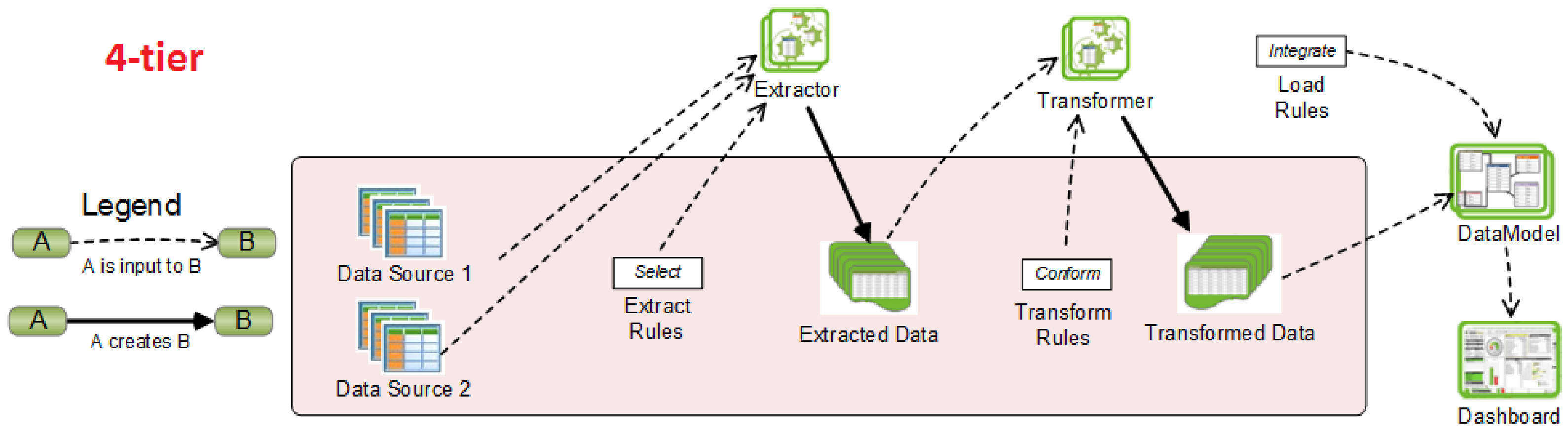
Performance Profile

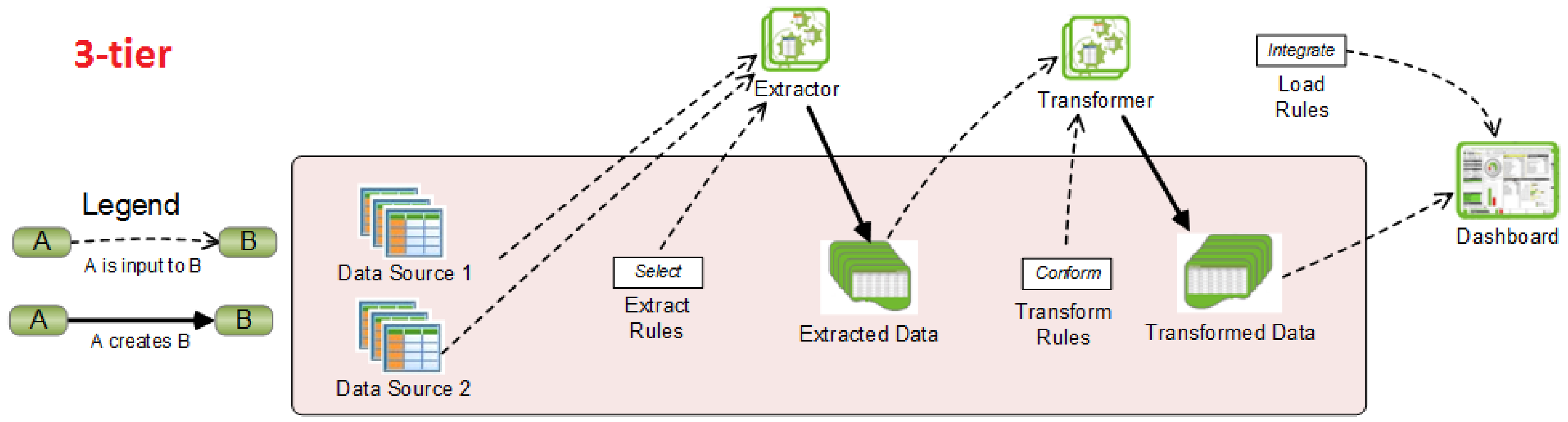
QVW	Table	Load Type	Task Start Time	Table Start Time	Table Completion Time	Total Time
						0:00:31
SalesExtract	ShippersExtract	FULL	02-07-17 7:01:02 AM	07:01:03 AM	07:01:04 AM	0:00:01
SalesExtract	ProductsExtract	FULL	02-07-17 7:01:02 AM	07:01:03 AM	07:01:03 AM	0:00:00
SalesExtract	ShipmentsExtract	FULL	02-07-17 7:01:02 AM	07:01:03 AM	07:01:03 AM	0:00:00
SalesExtract	CategoriesExtract	FULL	02-07-17 7:01:02 AM	07:01:02 AM	07:01:02 AM	0:00:00
SalesExtract	CustomersExtract	FULL	02-07-17 7:01:02 AM	07:01:02 AM	07:01:02 AM	0:00:00
SalesExtract	DivisionsExtract	FULL	02-07-17 7:01:02 AM	07:01:02 AM	07:01:02 AM	0:00:00

Considerations on 3-Tier vs 4-Tier Data Architectures

TSEEQ allows for both 4-tier (separate files for data model and dashboard) and 3-tier (single file for data model and dashboard) architectures. The 4-tier approach provides more modularization, but in Qlik Sense, 4-tier typically requires that the repository **Apps** folder be mapped to a folder data connection.

As such, the 3-tier approach may be preferred in some cases. Note that in the 4-tier approach, any single data model is a re-usable asset that can consumed by multiple dashboards. In the 3-tier approach, the consumable data model concept is not used; however, a single set of **Load Rules** (which define a data model) may be consumed by multiple dashboards. All dashboards consuming any specific **Load Rule** set will contain identical table structures and data sets.



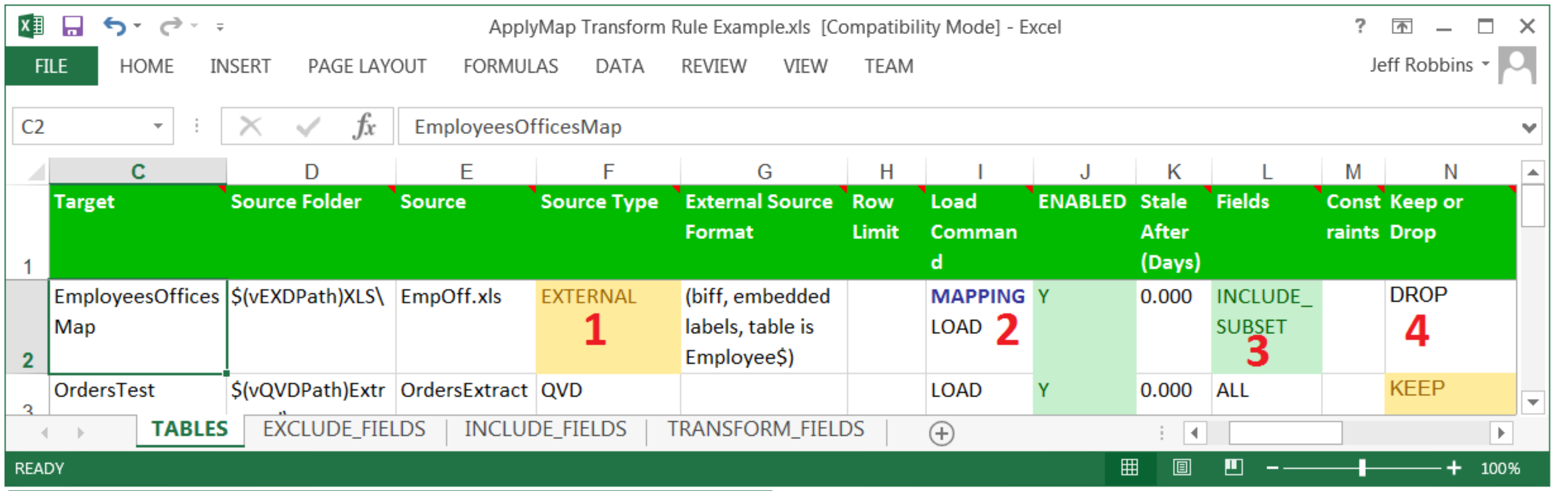


ApplyMap Example

TSEEQ includes an example Transform rule set implementing the [Qlik ApplyMap()] functionality:

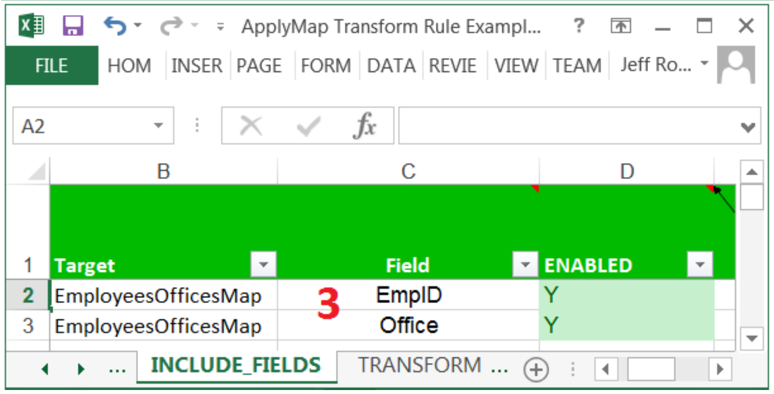
~*Source Documents\TSEEQ Sales Sample\ETL_Rules\ApplyMap Transform Rule Example.xls*

1. The mapping table can be loaded from either a QVD or from an EXTERNAL file (XLS*, CSV, etc). The example uses an XLS file (*EmpOff.xls*).
2. **MAPPING LOAD**, not simply **LOAD**, is the **Load Command** for the mapping table.
3. **INCLUDE_SUBSET** is required in the **Fields** column, if the source has more than 2 fields. If the source only has two fields, you could just use **ALL** in the **Fields** column, if those two fields are ordered with the key first and the mapped value second.
4. The **Keep or Drop** option does not apply, since Qlik will automatically drop the mapping table. You can specify **KEEP**, but Qlik will still drop the table.
5. A **TRANSFORM_FIELDS** rule applies the map.


ApplyMap Transform Rule Example.xls [Compatibility Mode] - Excel

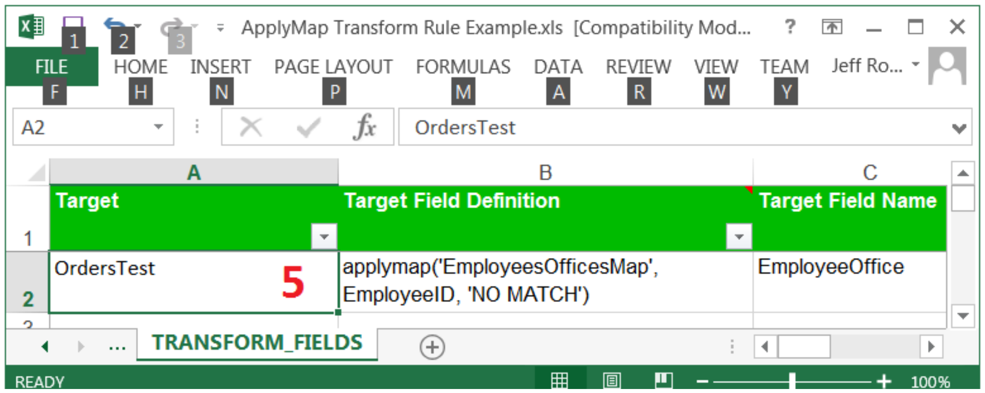
C2 EmployeesOfficesMap

	C	D	E	F	G	H	I	J	K	L	M	N
1	Target	Source Folder	Source	Source Type	External Source Format	Row Limit	Load Command	ENABLED	Stale After (Days)	Fields	Constraints	Keep or Drop
2	EmployeesOfficesMap	$(vEXDPath)XLS\	EmpOff.xls	EXTERNAL 1	(biff, embedded labels, table is Employee$)		MAPPING LOAD 2	Y	0.000	INCLUDE_SUBSET 3		DROP 4
3	OrdersTest	$(vQVDPath)Extr	OrdersExtract	QVD			LOAD	Y	0.000	ALL		KEEP


ApplyMap Transform Rule Exampl...

A2

	B	C	D
1	Target	Field	ENABLED
2	EmployeesOfficesMap 3	EmpID	Y
3	EmployeesOfficesMap	Office	Y


ApplyMap Transform Rule Example.xls [Compatibility Mod...

A2 OrdersTest

	A	B	C
1	Target	Target Field Definition	Target Field Name
2	OrdersTest 5	applymap('EmployeesOfficesMap', EmployeeID, 'NO MATCH')	EmployeeOffice

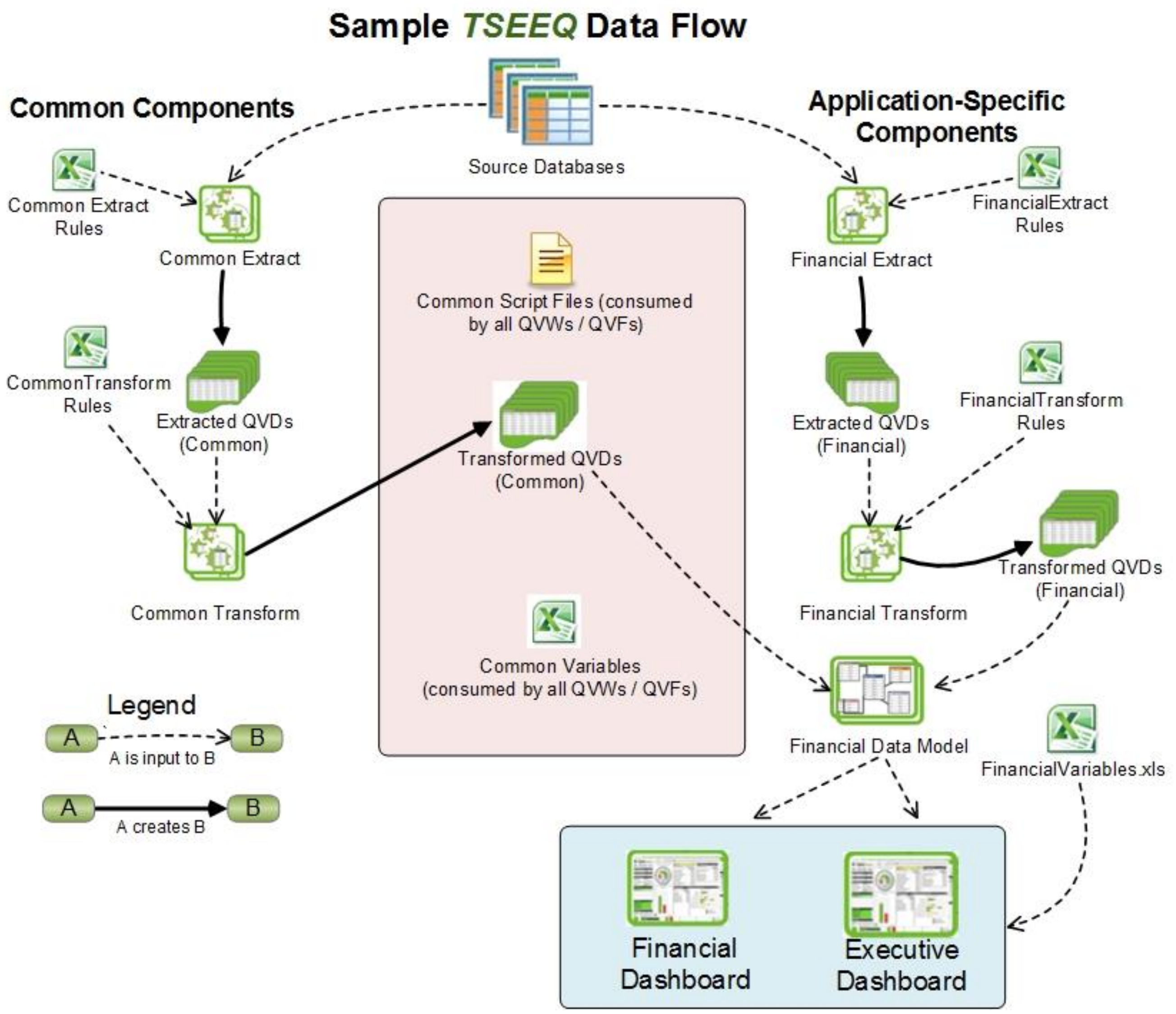
Notes on the *vApp* Variable

Qlik Sense developers should note that the ***TSEEQ*** *vApp* variable is not equivalent to the Qlik Sense concept of "app".

- In **Qlik Sense,** an "app" corresponds to a single QVF on disk. In Qlik Sense Desktop, this QVF is visible in the developer's local file system; in Qlik Sense Server, the QVF is within a centralized repository and therefore somewhat hidden.

- We should precisely define what "application" refers to in the following paragraphs: "application" is deployed software that the end user interacts directly with, for example, the **Financial Dashboard.** Technically speaking, an "application" is implemented by a set of files (including one or more QVFs or QVWs), but the end user does not have file-level visibility.

- In ***TSEEQ***, *vApp* is not a single file, but rather the prefix to multiple file names. *vApp* does not refer to an "app" in the sense of a QVF; rather, *vApp* refers to the set of files that implement a group of applications.



In the **Sample *TSEEQ* Data Architecture** (diagram at right), *vApp* = 'Financial'; the corresponding group of applications includes the **Financial Dashboard** and the **Executive Dashboard.** All back-end file names in the build chain (which end users do not see) are prefixed with '**Financial',** the value of *vApp*.

In this example, the **Financial** and **Executive** dashboards were initially conceived of as two separate initiatives; after some analysis, we realized that the **Executive** dashboard had high requirements overlap with the **Financial** and therefore based the **Executive** on the **Financial** build chain**.**

Note that the **Extract**, **Transform,** and **DataModel** QVFs in the **Example *TSEEQ* Data Architecture** (diagram on previous page) are all persisted as QVFs and are therefore considered “apps” from the Qlik Sense perspective. However, these back-end QVFs are not exposed to end users and are therefore not what we would refer to as “applications”.

So, to put this to code: suppose that there is a set of variables that every Financial-related QVF must read in, regardless of the QVF’s *vPurpose* (**EXTRACT**, **TRANSFORM**, data model **LOAD**, or front-end dashboard **APP**). Then, we could create a file, **FinancialVariables.xls,** and each of the financial QVFs could read in the variables with the following code:

```
/*****************************************************************************
   Load financial-specific variable definitions:
*****************************************************************************/
call Load_Variables_from_XLS('$(vVariablePath)FinancialVariables.xls', 'Sheet1');
```

We could further genericize this code:

```
/*****************************************************************************
   Load app-specific variable definitions:
*****************************************************************************/
call Load_Variables_from_XLS('$(vVariablePath)$(vApp)Variables.xls', 'Sheet1');
```

And the same single line of code immediately above could conceivably be used in every QVF that needs to read in *vApp*-specific variables: All Sales QVFs would read the Sales variables; all Financial QVFs would read the Financial variables.

We wish your success with ***TSEEQ***; please contact us with any questions!