DecisionExcelerator User Guide

Decision Resources, Inc.

Table of Contents

Back To Top

Back To Top

Introduction

DecisionExcelerator is an application sold by Decision Resources, Inc. (DRI) that provides bidirectional transfer and manipulation of data between Mongoose-based ERPs, such as Infor CloudSuite Industrial (SyteLine), and Microsoft Excel. DecisionExcelerator can communicate equally well with both multi-tenant SaaS and on-premises editions of Mongoose.

Data is transmitted between Excel and Mongoose through the Mongoose IDO interface. Data transmission operations (queries) are defined on a configuration tab in an Excel file. These queries will either download data from Mongoose into the same Excel file, create new Mongoose records, or update existing Mongoose records using data from that same Excel file.

As of this writing, DecisionExcelerator is only available on MS Windows.

A common approach is to run the application as you would any other application (interactively). However, it is also possible to schedule the tool to process Excel files at regular intervals. This is particularly useful when exporting Mongoose data for data analysis and reporting purposes.

The application installer provides other things in addition to the application. A selection of sample data files demonstrating capability can be referred to for example purposes. An Excel Add-In is provided, which offers tools to help with the creation and scheduling of queries. A separate tool named IDOQuery is also provided. It can be used to browse data and is provided to help with the formulation of query definitions. The IDOQuery tool is analogous to the Infor .NET Web Service Test Utility and can perform the same basic functions.

Back To Top

Support

A demo overview of DecisionExcelerator can be viewed on the DecisionExcelerator product page (link here).

Please direct all support requests to: DecisionExceleratorSupport@Decision.com.

Support Disclaimer

The data files included in the DecisionExcelerator installer are provided for example purposes only and are not guaranteed to work. The values and settings in these sample data files are for demonstration purposes only.

All data files are separate from the DecisionExcelerator product and are not supported as part of the product. **Support for data files is available as a billable service.** Please contact DecisionExceleratorSupport@decision.com if you need assistance using any data file.

Back To Top

Installation

DecisionExcelerator runs on any Windows 7+ or Windows Server 2008+ machine with access to a local or SaaS deployment of Mongoose. Microsoft Excel is not required for DecisionExcelerator to run. However, the Excel Add-In will only run if Excel has previously been installed on the system.

NOTE: Please refer to Infor KB 2320037 if using DecisionExcelerator with SyteLine 10. Following the November 2023 maintenance update, DecisionExcelerator will only operate normally for SyteLine 10 if ran from Windows 10 (or newer) or Windows Server 2016 (or newer).

An installer application is provided in a zipped file. Save and unzip this file then run the installer application.

The installer will not run correctly from inside the zip file.

The installer will install the Excel Add-in and will also create a folder named "DecisionExcelerator" in your "My Documents" folder. That folder will include a copy of this help file as well as a selection of sample data files which can be referred to for example purposes.

Upon launching Excel for the first time following the installation of DecisionExcelerator, you will be prompted to allow the DecisionExcelerator Add-In to run. Thereafter, the add-in will run immediately.

Back To Top

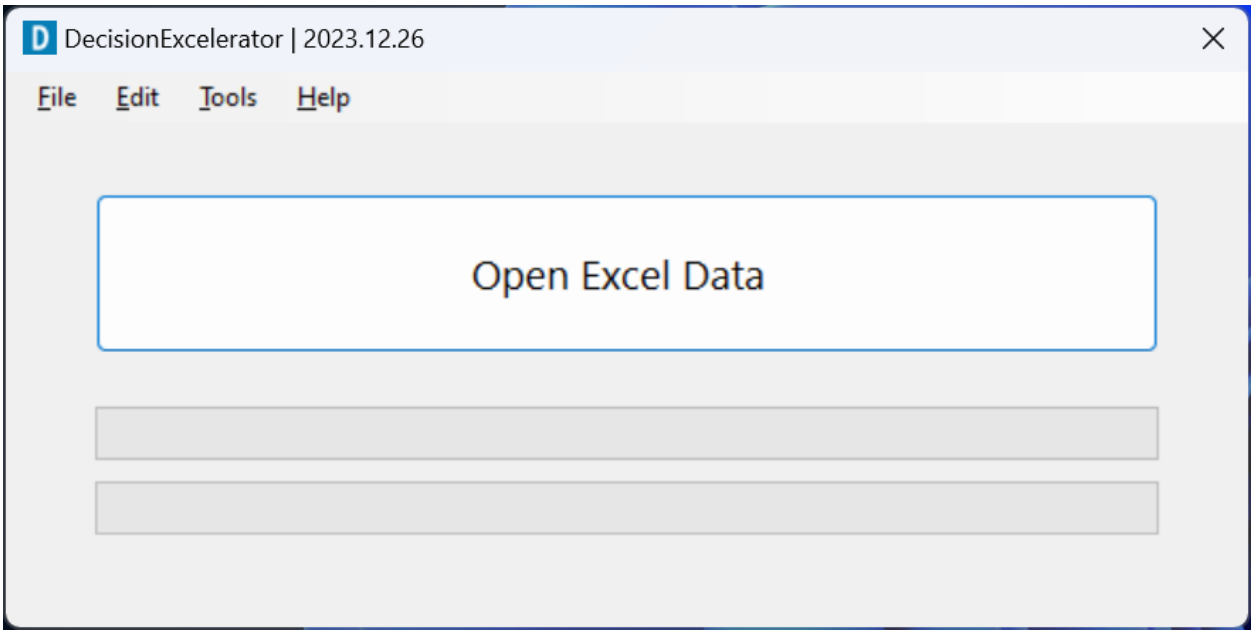
Running DecisionExcelerator

When the installer runs, it will place a shortcut on the desktop. You can also find the executable in the “Documents/DecisionExcelerator” folder. This folder will be created by the installer.

DecisionExcelerator can be launched directly from the shortcut, or it can be called from the command line with command line arguments.

Look for this icon:

This is what the DecisionExcelerator looks like when it runs. **The version number on the title bar will reflect the current installed version.**



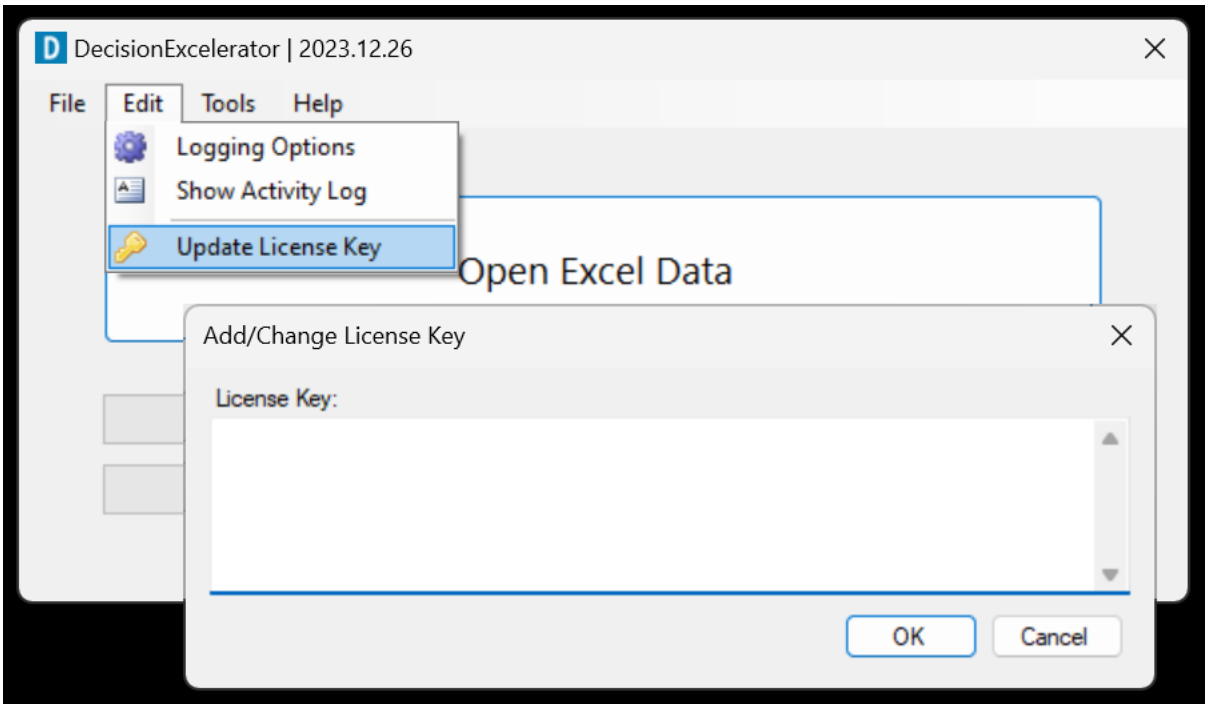
As DecisionExcelerator runs, the text on the application will change to keep you updated with the program’s progress. Pressing the Show Log button in the lower right corner will display this same information in a pop-up dialog. There are also two progress bars that will fill up as the program moves towards completion. The top progress bar tracks the progress of an entire file while the bottom progress bar tracks the progress of the current query. Additional information

Back To Top

will be populated in log files that are placed into the same directory as the data file. Log files provide valuable information in case the active queries do not behave as expected. The checkboxes below the progress bars control when log files are created.

License Key

When you obtain a license key from DRI, use the Update License menu to apply it. You will need to update the key file prior to processing a file. When processing a file without a key entered, DecisionExcelerator will limit the number of processed data rows to twenty, regardless of the row count indicated by the StartRow and EndRow settings in the ConfigInfo table.

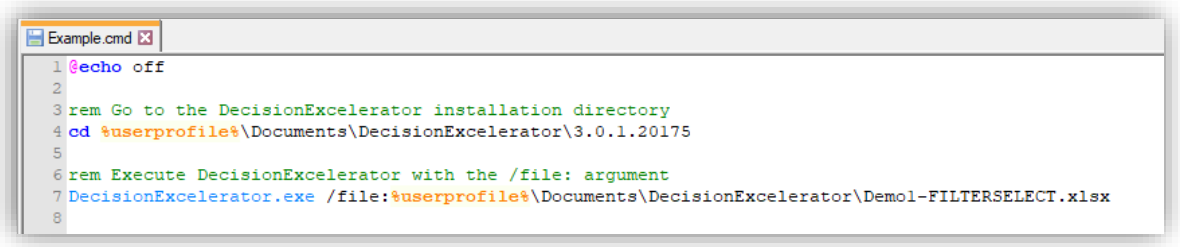


The license key is stored in user-specific settings on the computer. Therefore, if DecisionExcelerator is used by another user or on another machine, then the license key will have to be reapplied.

Command Line Arguments

It is possible to pass the name of the Excel file to be processed as a command line argument. When doing so, the normal User interface does not display, and no user interaction is required. One example where the command line is useful is when scheduling DecisionExcelerator to run as a Windows Scheduled Task. The command line argument is **/file:** followed immediately with the path to the Excel file. For example:

Back To Top



Here is the full list of commands.

/file:<file name>	DecisionExcelerator only runs in headless mode when this parameter is supplied. The other parameters listed below only take effect if this parameter is also supplied. If this parameter is missing, DecisionExcelerator runs in normal visible mode. Supplying this parameter is the same as pressing the Open File button on the UI. Supply a file path immediately following the colon. If the file path contains spaces, wrap the file path in quotes. The file must exist and the user running DecisionExcelerator must have read/write access to that file. None of the other parameters take effect unless this parameter is supplied.
/key:<license key>	This parameter allows you to update the license key. Supplying this parameter is the same as pressing the Add License Key button on the UI.
/nokey	Supplying this parameter clears the license key (if one was previously added).
/nolog	This parameter prevents the creation of status logs as a performance enhancement measure. Supplying this parameter is the same as unchecking the Create Status Logs checkbox on the UI. If you have already unchecked the Create Status logs checkbox on the UI then this parameter has no effect.
/clearlog	This parameter causes DecisionExcelerator to overwrite existing status and error logs for the current file every time DecisionExcelerator processes that file. Supplying this parameter is the same as unchecking the Append Logs checkbox on the UI. If you have already unchecked the Append Logs checkbox on the UI then this parameter has no effect.

Back To Top

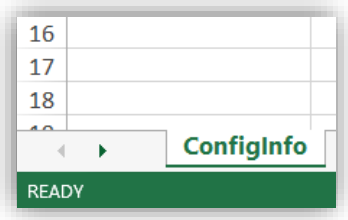
Excel Data Files

As noted previously, Excel is not required for DecisionExcelerator to run. However, DecisionExcelerator only works with Excel files, and requires that they be saved in the XLSX format (Excel 2007 and newer). The Excel file must reside in a folder accessible to DecisionExcelerator both for reading and writing.

DecisionExcelerator actively edits the file during processing so the file must be closed prior to processing to prevent blockages due to file locking. Also, the folder permissions must be set so that the file can be written to and modified by DecisionExcelerator.

ConfigInfo Tab

A tab named "ConfigInfo" is required and stores the configuration information for queries.



This tab contains essential parameters for the processing of a query. The table contained on this tab (example shown below) is used by the application to determine the active queries to be processed and what actions should be taken. Each row defines a separate query. Passwords are visible on this tab and therefore this file should be treated with care.

Active	Sheet Name	Action dropdown	IDO	Syteline Web ServiceURL	Config	User
	Jobs	FILTERSELECT	SLJobs	https://csi901t.inforcloudsuite.com/I	DECISION_DEM_DALS	DRIAutomation
	EDIPricingInSL	FILTERSELECT	SLItemprices	https://csi901t.inforcloudsuite.com/I	DECISION_DEM_DALS	DRIAutomation
	MatlTran	FILTERSELECT	SLMatltrans	https://csi901t.inforcloudsuite.com/I	DECISION_DEM_DALS	DRIAutomation
	SLLedgers	FILTERSELECT	SLLedgers	https://csi901t.inforcloudsuite.com/I	DECISION_DEM_DALS	DRIAutomation
	EDIContractPricing	INSERT	SLItemCustPrices	https://csi901t.inforcloudsuite.com/I	DECISION_DEM_DALS	DRIAutomation

Each column in the configuration table represents a setting that defines how the query behaves. If optional settings are blank or missing, then they are safely ignored by DecisionExcelerator. The "Blank ConfigInfo" button in the Excel Add-In provides information about what each ConfigInfo setting does.

Review **ConfigInfo User Guide** for detailed information on various ConfigInfo settings.

Back To Top

Data Tabs

The first row of the tab must contain the names of the IDO properties in use on that tab. These property names must exactly match how they appear in Mongoose (case sensitive). Begin at the first column (A1) and proceed to the right. DecisionExcelerator stops processing at the first empty or blank column. You may place unused property columns after a blank column.

Naming must precisely match (case sensitive) for both the IDO name and IDO property names.

UETs can be accessed with DecisionExcelerator but property name definitions require special formatting. UET's must follow the default naming convention of appending the "Uf_" prefix to the property name. In addition, the table alias must be included in the property name. For example, a UET field on the Items table would be included in the template with a property name of: itmUf_fieldname.

If Extended IDO properties are included in the list of properties to update or retrieve, the log will record that an IDO Property defined in the data file is not part of the standard base IDO and that these extended IDO properties and should be closely reviewed.

The data in certain fields are space padded or fully expanded to the full field length. Examples are Customer Order Number (CoNum) and Customer Number (CustNum). DecisionExcelerator manages those well, but careful consideration should be made when working with these key properties. If you are loading data with an alpha prefix that will be fully expanded in the client form, you should carefully prepare and pad the data. It is best to format these fields as text and space pad or fully expand your source data – A000001 instead of A1.

The general rule to follow is to ensure your data conforms to the data type as specified in the IDO or returned by the SELECT process.

Writing To A Text File

A new setting Write To Text File will cause DecisionExcelerator to write the results of a FILTERSELECT query to a tab-delimited text file instead of writing to the data tab. The data tab is still required and is used to supply the list of IDO properties that will be retrieved by the FILTERSELECT query. The Write To Text File setting honors the RecordCap setting. The resulting file will be overwritten to, or appended to, based on the Append setting. StartRow and EndRow settings do not apply to text files. No snapshot actions will occur for a FILTERSELECT query where the Write To Text File setting is enabled.

Writing to a text file offers performance benefits in certain high-volume reporting scenarios. The targeted use case is where huge tables are being exported for use in a separate reporting platform. In this scenario, the most noticeable advantage of a text file over an Excel file is that the text file does not have a one million row record cap.

Back To Top

Snapshot Exports

When running DecisionExcelerator, the snapshot activity will run after all active queries are processed. The following criteria must be met for snapshot activity to occur:

- Snapshot activity is only available for queries that are marked Active in ConfigInfo settings.
- Snapshot activity is only available for queries that use the FILTERSELECT action.
- Snapshot activity is only available when the data tab corresponding to Sheet Name exists in the file.
- Snapshot activity will only occur when the data tab has at least one row of data.

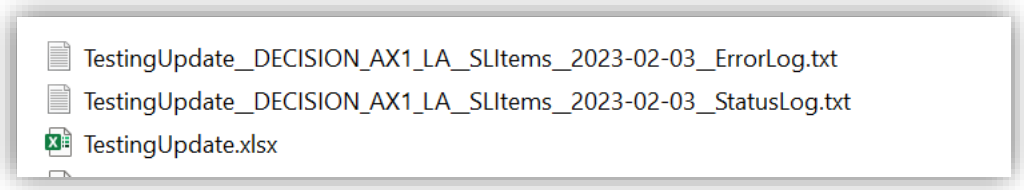
Note that snapshots will copy the entire file to the specified location. The ConfigInfo tab will be removed from the exported copy. However, all other tabs will remain included in the file. Consequently, you may desire to use the Hide Sheet action to make one or more data tabs invisible in the exported copy.

FILTERSELECT removes all existing data from a data tab prior to running the new query unless the Append action is selected. Without using Append, you will only get exported snapshots when the query returns data. However, you may get inconsistent results when using the Append action.

Back To Top

Log Files and Reprocessing Errors

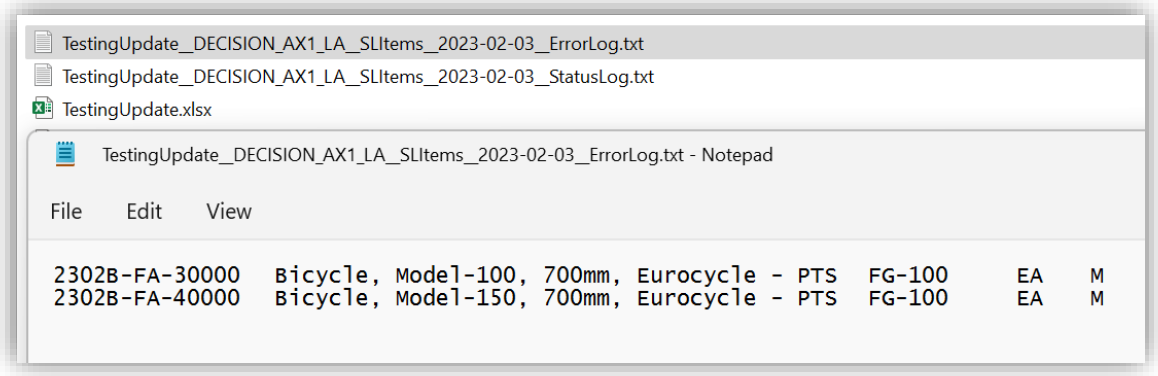
When DecisionExcelerator processes an Excel file, it has the ability to produce two kinds of log files. The naming convention for these log files includes the original Excel file name as well as the name of the data tab being processed. If an Excel file contains multiple active queries using separate data tabs, DecisionExcelerator will produce multiple log files in a single session.



The status log files contain a chronological history of actions taken by DecisionExcelerator while processing the file. The information contained in this log may be useful when looking for performance enhancement opportunities with regard to the way specific queries are configured.

The error log file contains error messages. These are helpful when troubleshooting problems. Additionally, the error log file contains a tab-delimited copy of any records from the data tab where errors were encountered while processing. For those actions that are limited by StartRow and EndRow, the records added to the error log will be taken from that range on the current data tab.

Only records with errors are added to the error log. They are added to the log as tab-delimited fields.



Reprocessing these error records can be achieved by copying these records into a new range of rows on the data tab, and then modifying StartRow and EndRow to process just these records.

Back To Top

DecisionExcelerator Menu Bar

The Excel Add-In has been discontinued. Several of the tools that were provided in the Add-In have been moved to the DecisionExcelerator menu bar.

Blank ConfigInfo

If you wish to create a new data file, this button will create a blank ConfigInfo tab in the current Excel file. The ConfigInfo User Guide provides information about what each ConfigInfo setting does.

SyteLine Forms & Fields

When designing queries in data files, it is necessary to understand the IDO collections and Properties with which you are working. This button adds tabs to the current Excel file that display lists of SyteLine forms and form fields, showing the IDO collection and property bindings for each form and field. The information gleaned here can help guide you as you construct a query.

This same ability is available in the DecisionExcelerator application through the SyteLine Forms & Fields menu.

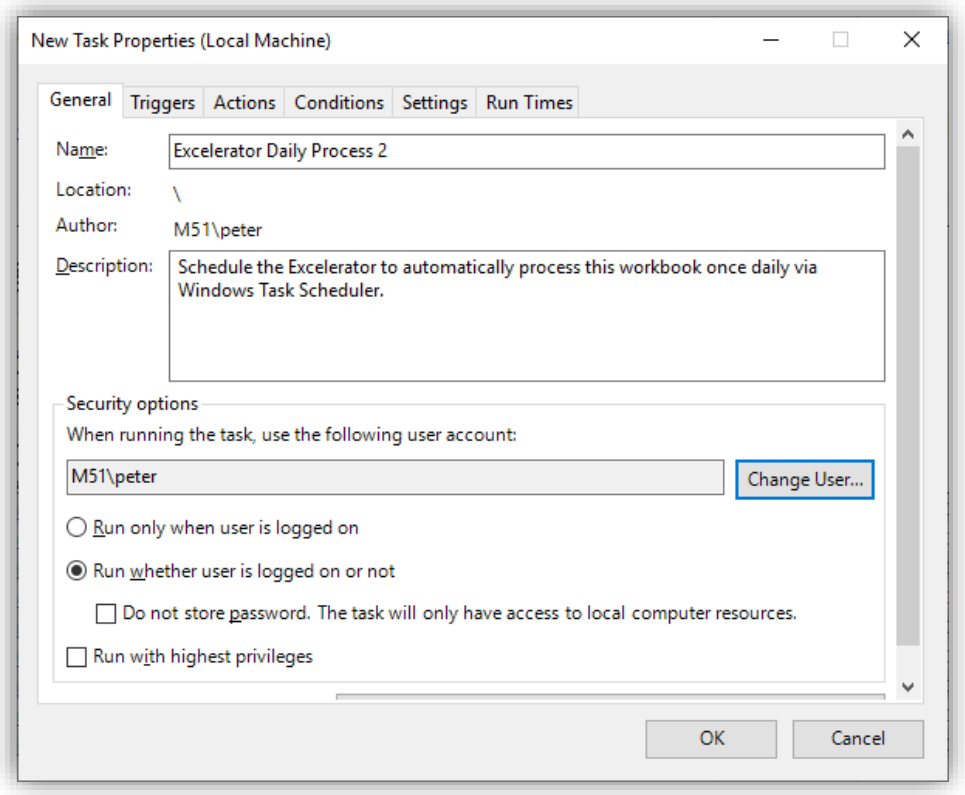
Launch IDOQuery

DecisionExcelerator ships with an IDOQuery tool that is similar in nature to the Infor .NET IDO Web Service Test Utility and can be used to validate IDO requests and the resulting data sets.

The IDOQuery tool can also be launched through the IDO Query menu in the DecisionExcelerator application.

Back To Top

Schedule Queries



When running DecisionExcelerator with command line arguments (see Appendix 4), DecisionExcelerator does not require user interaction to run. As a result, it can be scheduled to run at regular intervals via Windows Task Scheduler. Such a feature might be valuable if you wanted to download a table of data daily where that data could serve as the foundation for a dashboard. Pressing the Schedule Queries button will open a task scheduling dialog. Confirming your scheduling options will cause DecisionExcelerator to generate a Windows Scheduled Task which will launch DecisionExcelerator at the specified interval. Note that by default the scheduled task will cause DecisionExcelerator to process the data file that you have open currently.

Task scheduling can also be configured using the Schedule Background Task menu in the DecisionExcelerator application.

Check the error log for the selected file when the scheduled task does not run as expected.

User Permissions for Scheduled Tasks

Please pay special attention to these notes regarding user permissions!

Back To Top

Windows UAC is known to prevent programs from accessing files on a network file share or mapped drive during certain circumstances. The best approach is to make sure that DecisionExcelerator and the file to be processed both exist on the same computer. Snapshot settings can be used to send the results of FILTERSELECT queries to other locations.

Also, due to Windows permissions regulating scheduled tasks, the user scheduling the task must be a member of one of the following local user groups:

- Administrators
- Backup Operators
- Performance Log Users

Back To Top

Intelligent Data Objects (IDOs)

IDOs, or Intelligent Data Objects, provide the backbone to Mongoose's middleware. The external touch points offered by Mongoose provide access to Mongoose data via IDOs. Information about IDOs, their use, and their schema definitions can be found on the Infor Xtreme user support portal.

IDO User Access

Access to an IDO requires the use of a Mongoose user account, wherein a user ID, password, and config name are required to access data just like when logging into WinStudio, the WebClient, or Ming.le as a normal user. The Automation Module is required to access the IDO middleware regardless of any other security and permission settings on the user account.

Security Considerations

The Excel file used by DecisionExcelerator stores the configuration name, user ID, and password information in human readable text. This should be considered when providing access to the Excel files used by this application. Methods for access include:

- Log In as "SA" (Major Security Issues)
- Create a new login for the sole and specific use in this application. (Less Security Issues)
- Use the login of the person running the application. (User is responsible)

A password may be blank, however, that is not an ideal security scenario, and the use of passwords is recommended.

User Modules and Permission

User Setup may be required. Please work with the system administrator or person responsible for user setup and permission.

In certain cases, the user account may need to be configured to Allow Remote WinStudio. This can be configured either for the individual user account or can be changed globally in the Process Defaults form. When making changes to the Process Defaults form, you must also discard the IDO cache via the Unload Farm IDO Metadata form.

If you observe a message that refers to LCDT IDO's, then ensure the user account has the DocTrack module assigned. This is necessary due to the number of IDO's extended by DocTrack and the IDO extension methodology.

Back To Top

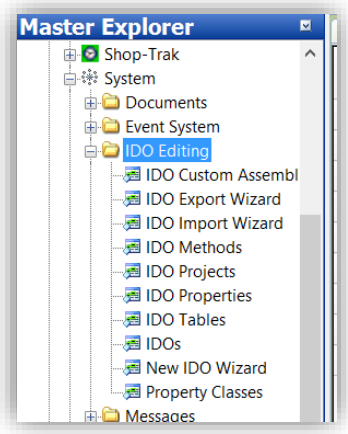
Users may be Super Users or limited to specific middleware object (IDO). While checking the Super User check box on the Users form is easy, this user can read and write to most areas of the ERP using the DecisionExcelerator. If the DecisionExcelerator files are to be processed by other than administrative users, or are to be distributed, it is advised to create users with limited authorizations.

In order to create new data tabs, or to validate properties, the user must have access to the IdoTables and IdoProperties MIDDLEWARE on the Object Authorizations for User form, as well as the Object Name (IDO), and of course, the appropriate privileges.

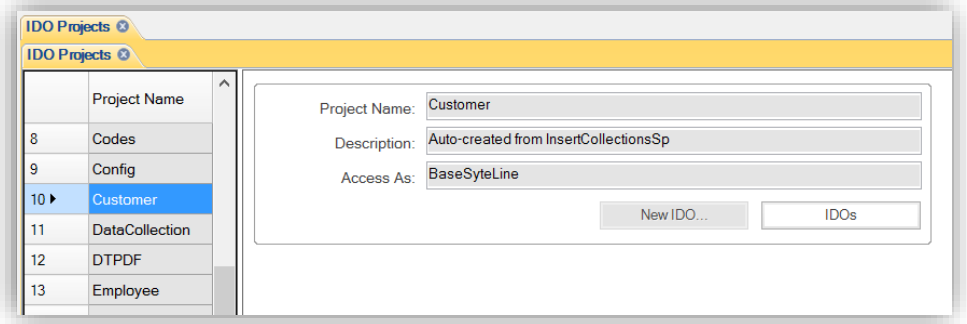
IDO Forms

These forms can be found under “System” and then “IDO Editing” in the Master Explorer window.

The hierarchy presented in the forms is IDO Projects -> IDOs -> IDO Properties. Although IDO Properties does appear as a sub-collection in IDOs, the full form provides more detail and filtering options.

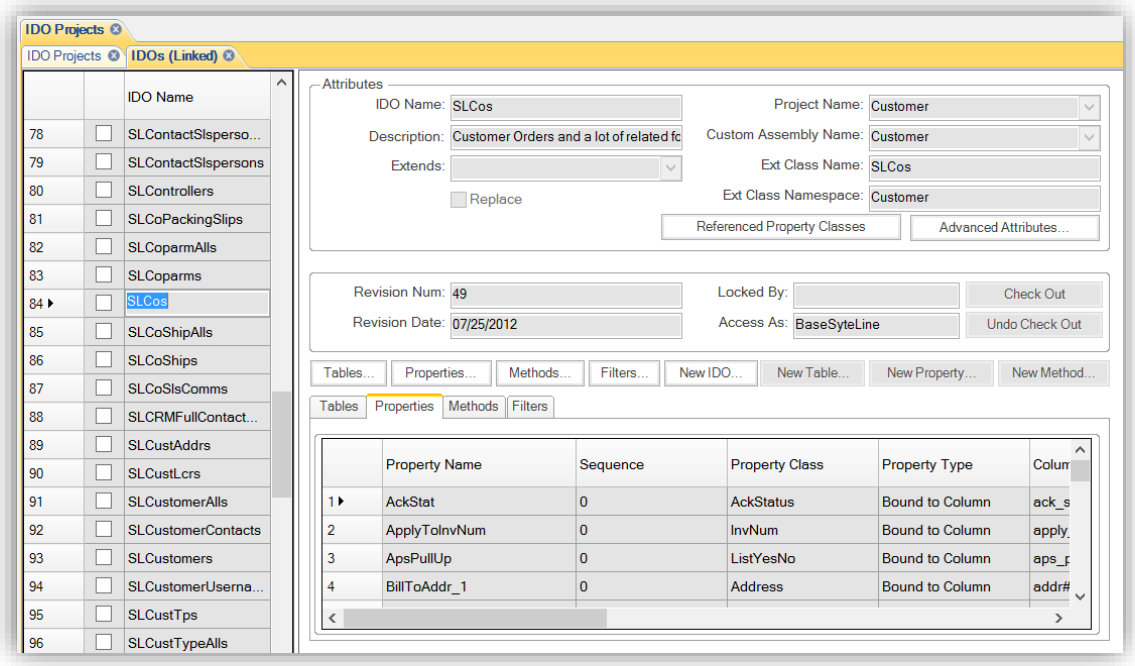


IDO Projects

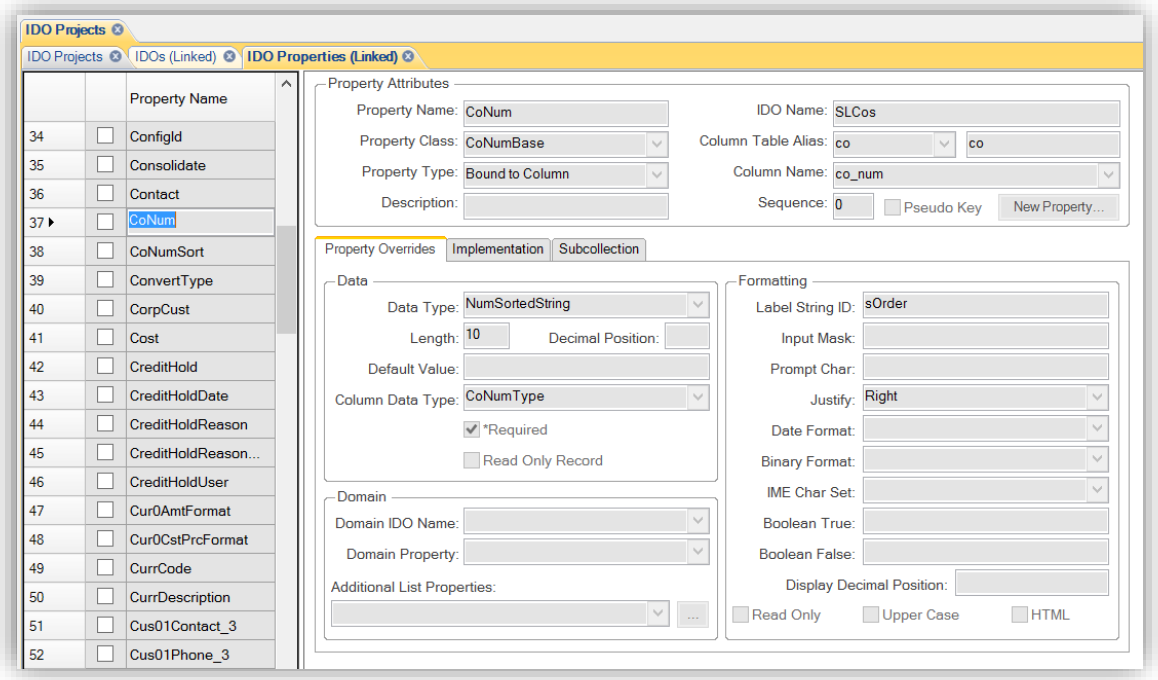


Back To Top

IDOs



IDO Properties



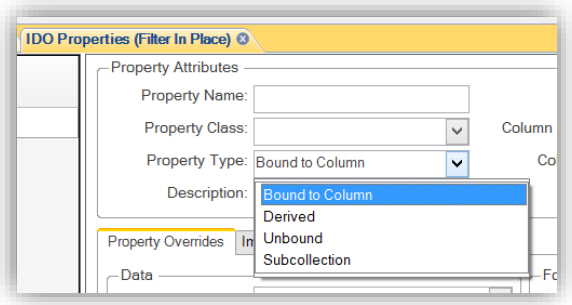
Back To Top

Property Types

Four types of IDO Properties can be accessed through Mongoose.

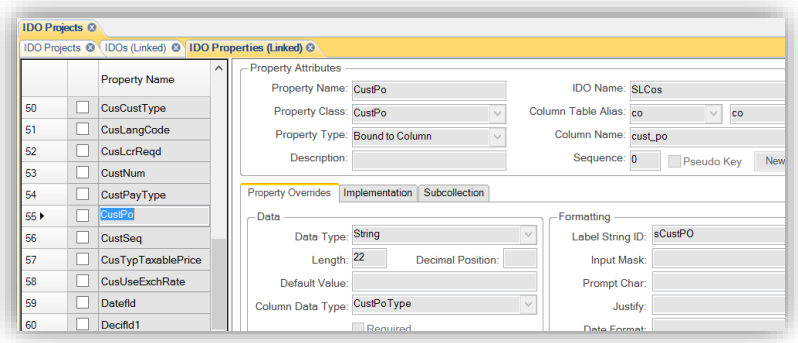
- Bound to a column in a table.
- Unbound.
- Derived from a SQL expression (the SQL expression can be used to perform calculations, subqueries, or calls to SQL functions).
- A sub-collection.

DecisionExcelerator typically will be accessing only fields that are bound to a column in a table. A case may exist where there may be a requirement to access and read one of the derived fields thereby loading a calculated value into the target Excel file. Unbound and sub-collections are unlikely to be used. There is value in understanding the difference in these types with a focus on bound versus derived.



Bound

The example below shows the property CustPo in the SLCos IDO. Notice the Property Type is *Bound to Column*. Information on the top right of Property Attributes shows the table alias and the column name. This information is the binding of the property to the column. The tables and field names at the database level may differ slightly from the convention used at the IDO layer.

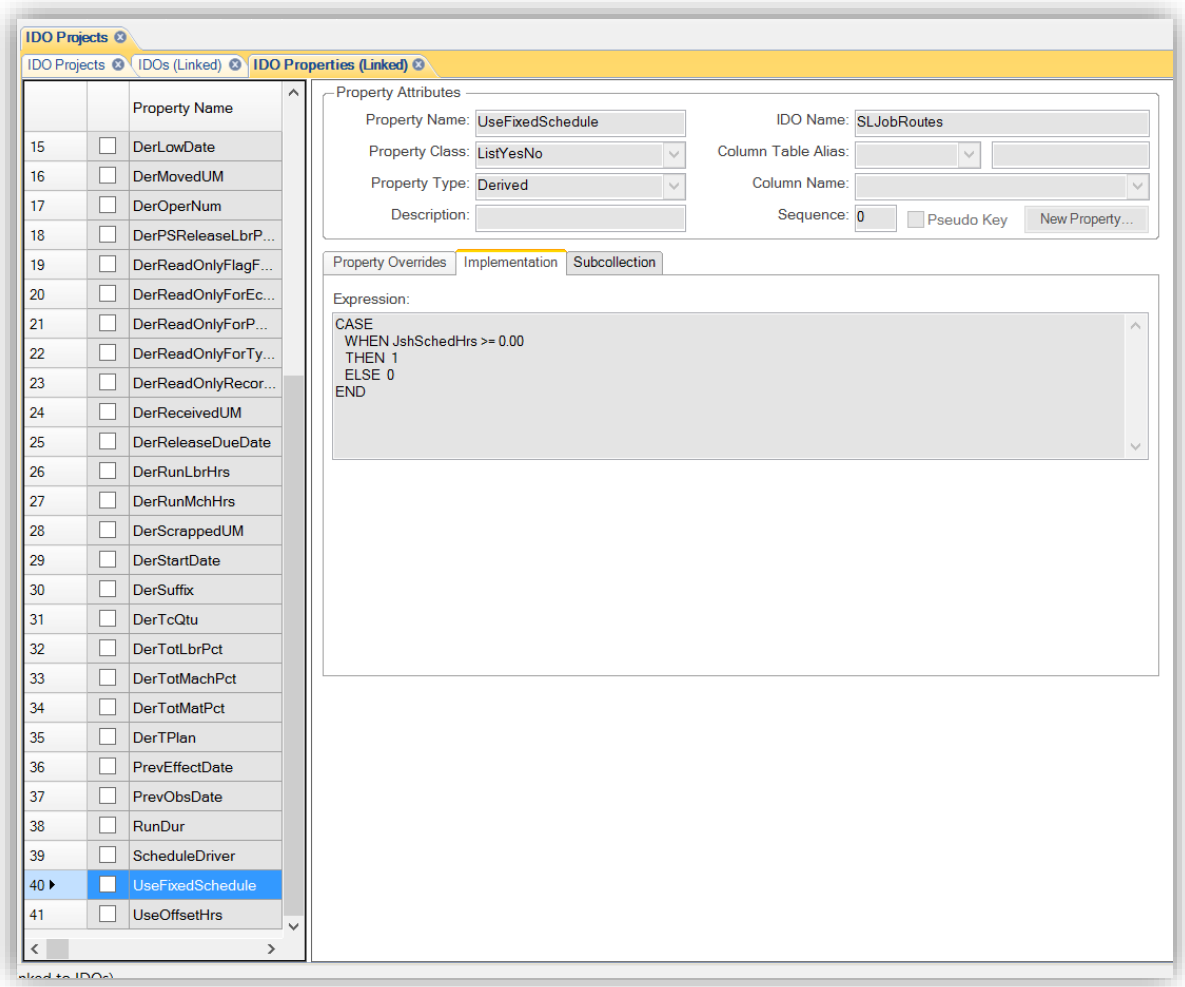


Back To Top

Derived

The example below shows the property UseFixedSchedule in the SLJobRoutes IDO. Notice the Property Type is *Derived*. Also note that most derived fields start with *Der*. Information on the top right of Property Attributes shows no table alias or column name but instead the tab labeled “Implementation” is enabled. On that tab we can see the expression used to calculate the field.

In this example we know that, in the client form, the field for Used Fixed Schedule Hours is a checkbox which enables another field where a value can be added for hours. On this form we see how, however, this field is calculated based off the Schedule Hours field. This may be unexpected and appear the opposite from the user interface. Note that this same functionality does exist throughout the ERP.



Back To Top

Layers of Logic – User Interface, IDO, and Database

Mongoose is a multi-tier application. Processing logic exists at more than one tier or layer. The IDO interface is an external touch point to a middle tier. Evaluation of fields and calculations may happen at this level, but this is not always the case. What is seen from the user experience point of view may be occurring at the database level or at the form level. For most forms, logic happens in all three layers.

An important example of this to consider when loading data through DecisionExcelerator can be seen in the forms for Customer Order Lines and Purchase Order Lines. In both forms, the field labeled Item Description appears when a stocked Item is chosen. The Item's description is pulled from the Item information, and this happens at the form level reacting to user input. This information does not automatically populate at the database level, nor does it populate automatically at the IDO level. The DecisionExcelerator operates at the middle layer and must therefore collect and upload this information or the field will be left blank (NULL) and would need subsequent correction.

IDORequestService Runtime URL

DecisionExcelerator communicates with the IDO middleware using the IDO Request Service URL. The main points to note are:

- Is the service running securely or unsecure (HTTP or HTTPS)?
- What is the name of the server hosting the IDO service?
- Is this a multi-tenant SaaS environment, where the ConfigGroup query string parameter is required?

Knowing the answers to these questions will help with providing the correct URL.

DecisionExcelerator prefers to use Mongoose's SOAP endpoint (IDOWebService.asmx) but will use the ASP.Net endpoint (RequestService.aspx) when the SOAP endpoint is unavailable. Shown below is an example of where to find the IDOWebService URL.

Back To Top

The target server may reside on a protected network segment not accessible from a user's computer. To evaluate access, open a web browser on the computer that is to be running DecisionExcelerator and attempt to browse to the address identified previously.

Back To Top

Inserting Current Operations

A multi-step process is required to insert Current Operations. You can query existing Current Operations by running a FILTERSELECT query on the SLJobRoutes collection. However, writing to the SLJobRoutes collection is not sufficient by itself to create new Current Operations.

DecisionExcelerator can create Current Operations in a single step using a custom IDO method.

An IDO Custom Assembly named DRI_SLJobRoutes can be found in the same zip archive folder as the DecisionExcelerator installer. After installing DecisionExcelerator, a copy of this IDO Custom Assembly can be found in the "Documents/DecisionExcelerator" folder.

Installing the IDO Custom Assembly will create an IDO collection of the same name. This IDO will contain an IDO method named *InsertCurrentOperations*.

An example for calling this IDO method can be found in the

Demo10-InsertCurrentOperations.xlsx

file that is installed into the "Documents/DecisionExcelerator" folder. This file is provided by the DecisionExcelerator installer.

Because DRI_SLJobRoutes.InsertCurrentOperations is an IDO method, the fields on the "Operations" data tab represent IDO method arguments. Consequently, all those fields must remain present in the data tab to preserve the index sequence of method arguments. Those fields that are populated in the demo file should be considered required and should always have a value. The remaining fields must be present, but only need to have values populated in them as needs dictate.

This IDO Custom Assembly is provided as-is and is only known to work with SyteLine version 10.

Support for importing, maintaining, and using the DRI_SLJobRoutes IDO is available as a billable service.

Please contact DecisionExceleratorSupport@decision.com if you need assistance importing or using the DRI_SLJobRoutes IDO.

Back To Top