ONE Desktop Core

v15.4.x

Agenda

Day 1

- Introduction to ONE Desktop
- Quick Demo
- Read / Write Operations
- Data Profiling
- Flow Control

Day 2

- Best Practices & Conventions
- Expressions
- Data Transformation
- Step Debugging
- Scores & Explanations
- Lookup & Dictionaries

There are 8 practical lab exercises on this course.

ataccama

Introduction to ONE Desktop

ataccama

ONE Desktop – Introduction

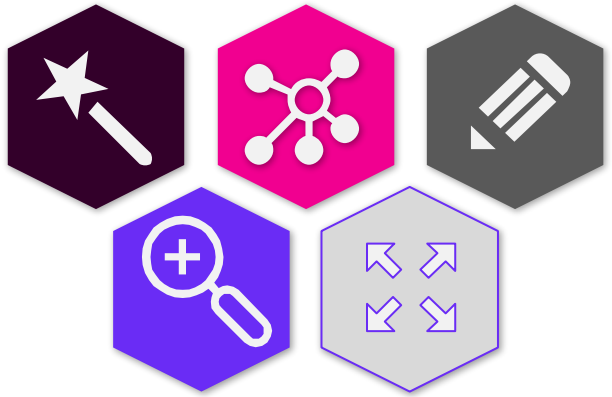
ONE Desktop is a powerful engine for improving data quality coming from various data sources.

Key features:

CLEANSING

MATCHING

MERGING



DATA ANALYSIS

STANDARDIZATION

VALIDATION

Highlights:

- Available ready-to-use DQ modules
- Flexible and customizable component architecture
- Works in both online and batch modes

Typical use case scenarios:

- Assessment: Data Profiling
- Data Cleansing
- Data Matching and Enrichment
- Prevention: Online Services & DQ Firewall
- Address Cleansing and Identification
- Rule-based engine for complex Data Quality Monitoring and Reporting

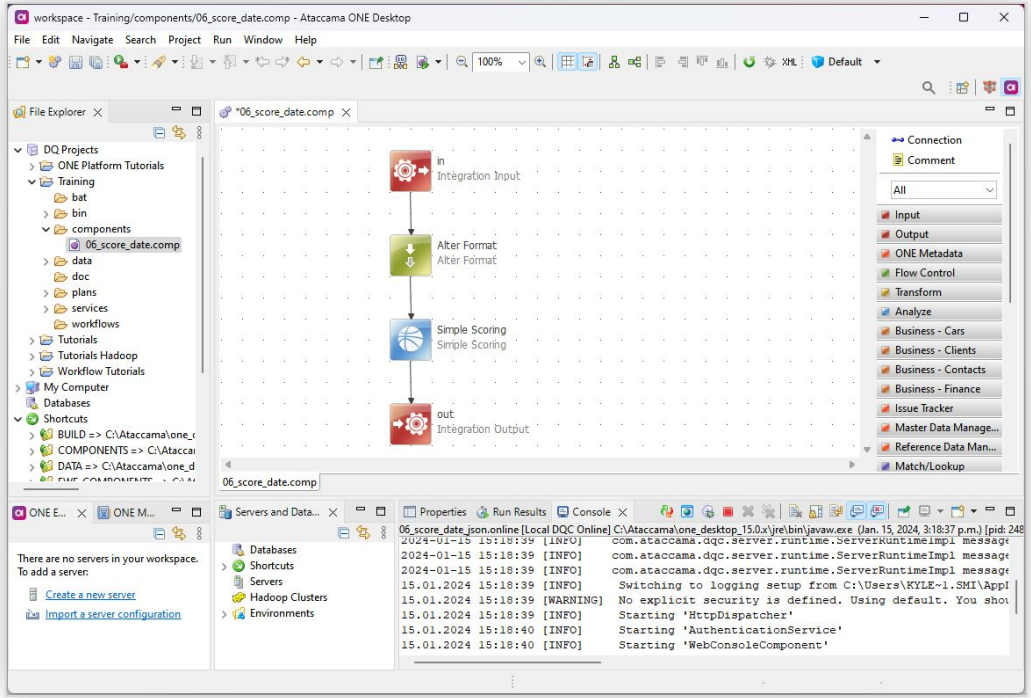
ataccama

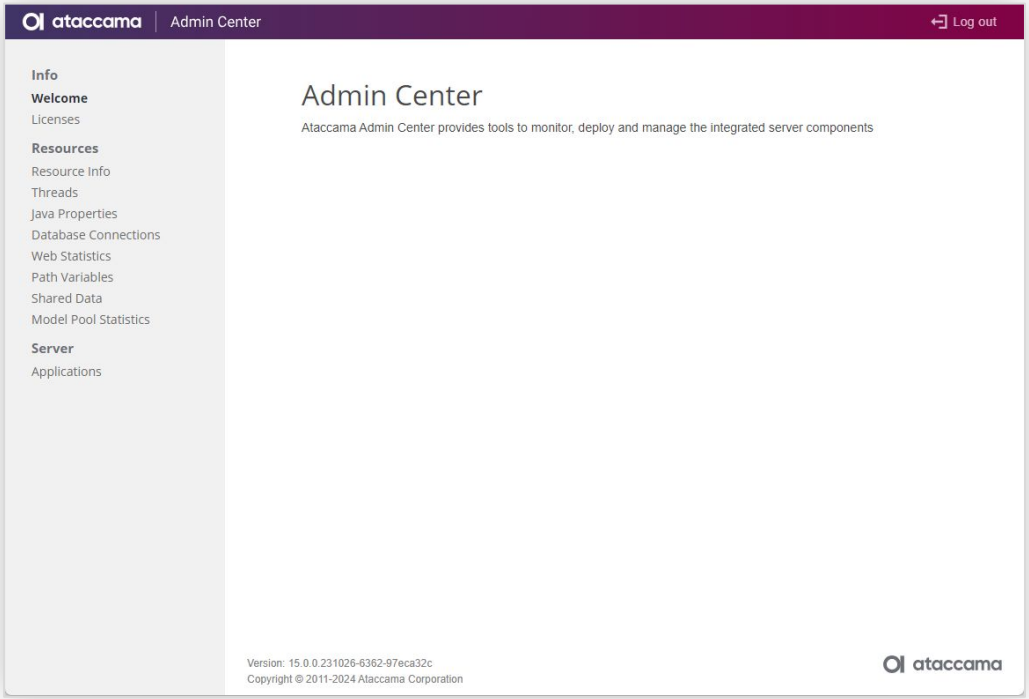
ONE Desktop – Architecture

ONE Desktop
Integrated Development Environment
(**IDE**)

Ataccama Server

Admin Center





ataccama

ONE Desktop – Architecture

ONE Desktop components and requirements:

Server engine (core)

+ System requirements

- Server engine (runtime core):
  - Executes plans and services
  - Contains various tools and documentation
- Platform-independent
  - Can be run on almost any platform (combination of OS and architecture)
- Requires a suitable Java Runtime Environment (JRE) on the platform
  - Server engine requires JRE 1.8 or higher

GUI / ONE Desktop

- Eclipse-based application
  - Bundled with an updated JRE
  - No additional packages need to be preinstalled
- ONE Desktop GUI
  - Interprets the XML code from plans and displays it in an easy-to-use form
  - Calls the runtime whenever a plan or service is executed
  - Shows console and monitoring, as well as additional information
  - Allows the user to configure sets of runtime variables for the execution of each specific plan
  - Includes debugging, content assists, and searchable help for ease of development

ONE Desktop: Other Requirements

License Keys

License keys have PLF extension.

- Created and managed by Ataccama.
- Version-dependent.
- Can be restricted to only allow the use of a certain subset of modules and components.
- Can be generic or detail-specific (e.g. user, architecture, validity).

How to install a license?

- Place the PLF license file in the USER folder or in {ONE_Desktop}/runtime/license_keys.
- Install the license directly from the GUI/IDE using the menu **Help > License Information > Install License**.

Database Connections

For connectivity with a DB, a JDBC driver must exist.

- Most DBs - and all the popular ones - have JDBC drivers available.

ONE Desktop bundles come with popular JDBC drivers that can be found in {ONE_Desktop}/runtime/lib/jdbc.

How to Install a driver?

- Server: Place the JDBC driver as a JAR file in the {ONE_Desktop}/runtime/lib folder.
- GUI (Desktop): Navigate to the menu **Window > Preferences** and choose **Ataccama ONE_Desktop > DB drivers**.

Creating a new project

Before you begin doing any transformations or analysis of data, you need to create a project. A project is a set of multiple plans, components, data files, workflows, and custom scripts, which are usually focused on solving a specific task and are logically organized into folders.

Empty project

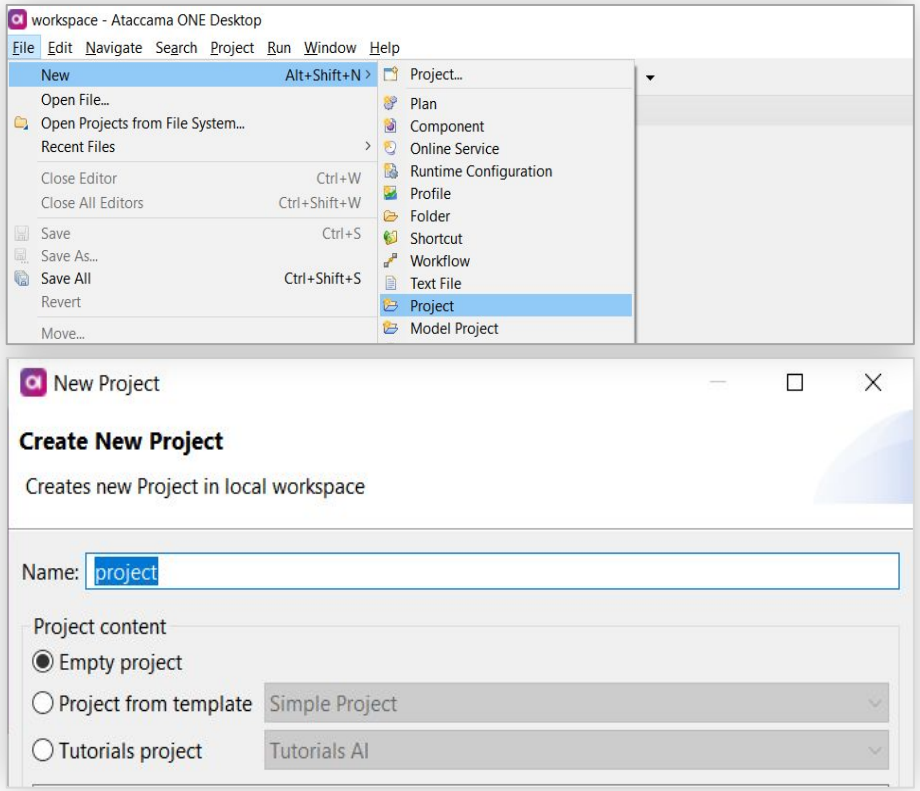
- No predefined folder structure

Project from template

- **Simple Project** – begin with two default directories
- **Complex Project** – begin with a full folder structure

Tutorials Project

- Tutorials
- Tutorials Hadoop
- Workflow Tutorials
- ONE DQM Server Example

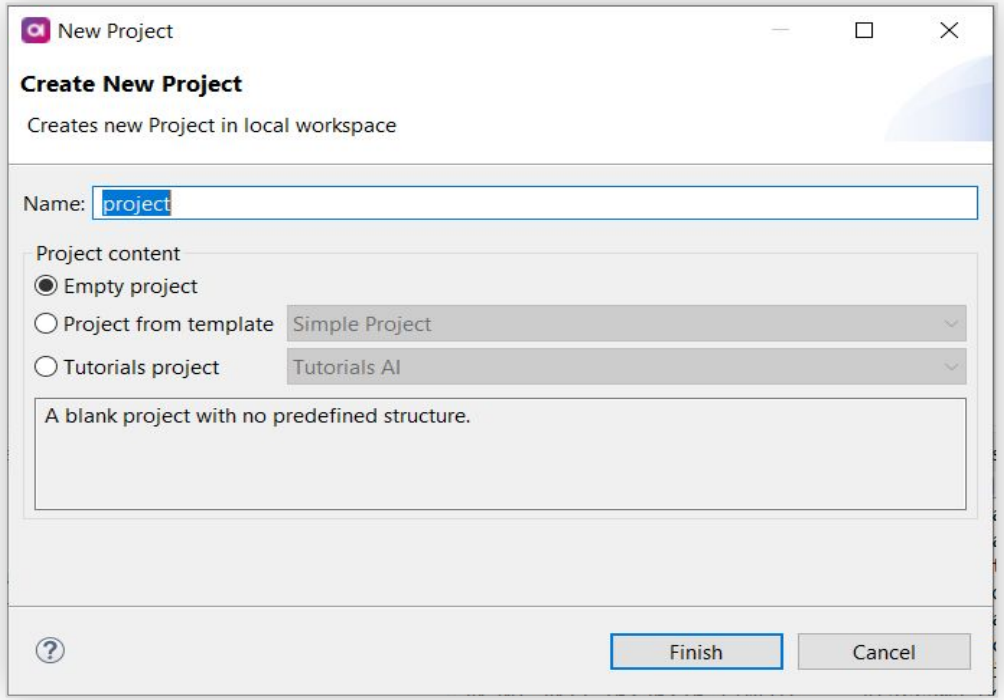


ataccama

Project Types: Projects

Projects

- Empty Project
  - No predefined folder structure
- Project from template
  - Simple Project – two default directories
  - Complex Project – full folder structure
- Tutorials Project
  - Tutorials
  - Tutorials Hadoop
  - Workflow Tutorials
  - ONE Platform Tutorial

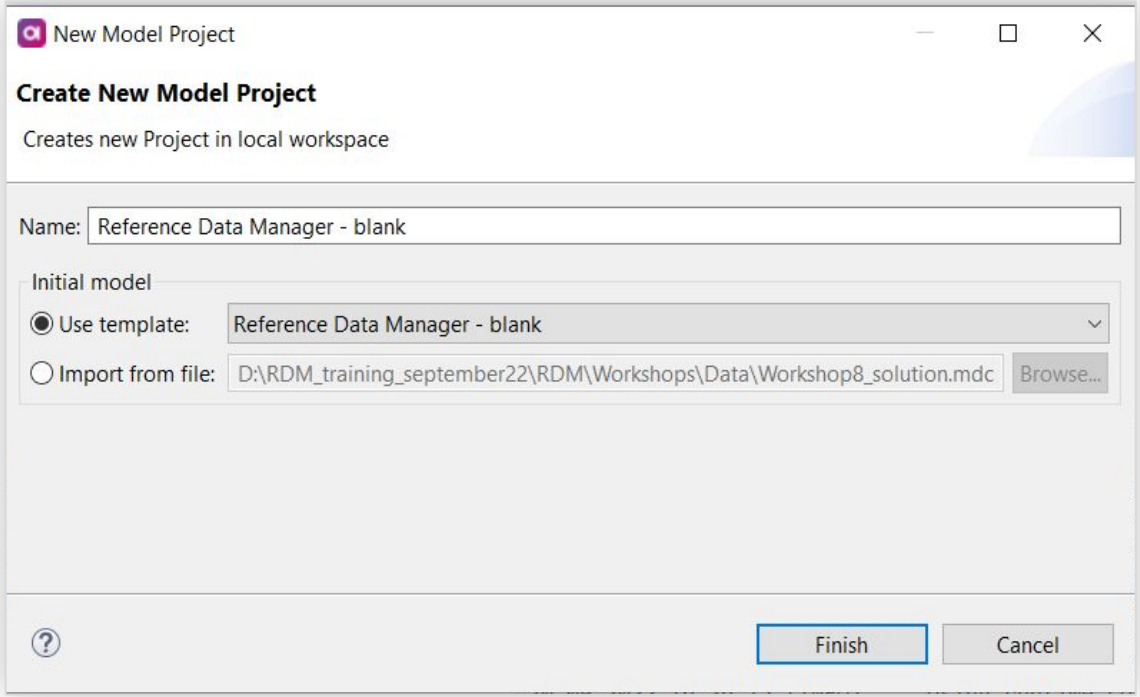


ataccama

Project Types: Model Projects

Model Projects

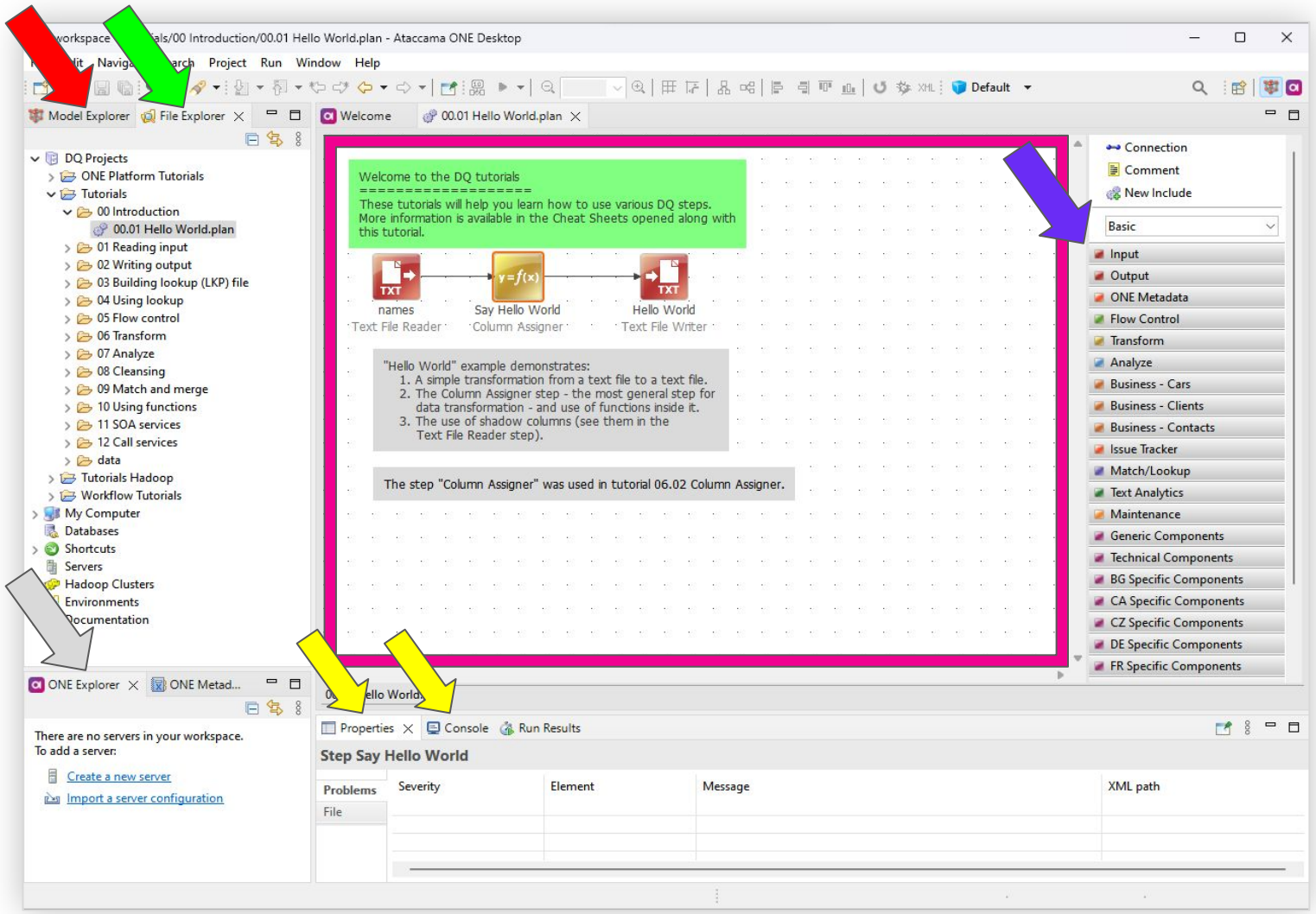
- File for import
  - Reuse or modify existing projects
- Templates Configured to a specific Ataccama module:
  - MDM
  - Profiling
  - DQIT
  - RDM



ataccama

ONE Desktop: Interface

- **Model Explorer**
- **File Explorer**
  - DQ Projects
  - Databases
  - Shortcuts
  - Servers
- **Canvas**
- **Palette**
- **Console**
- **Properties**
- **ONE Integration**



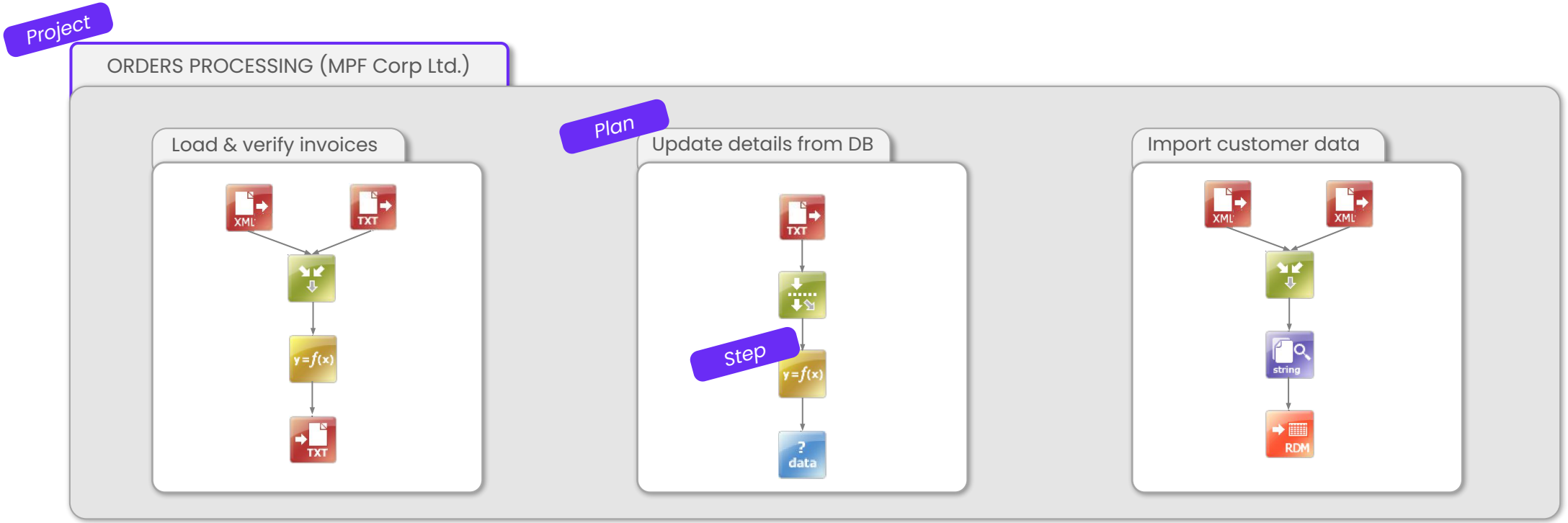
ataccama

ONE Desktop: How it works

Projects A set of multiple **Plans** and data files focused on a specific overall goal

Plans Definition of logic and rules that are applied to data in order to produce a desired output

Steps Perform a specific function or algorithm with relation to other **Steps**.

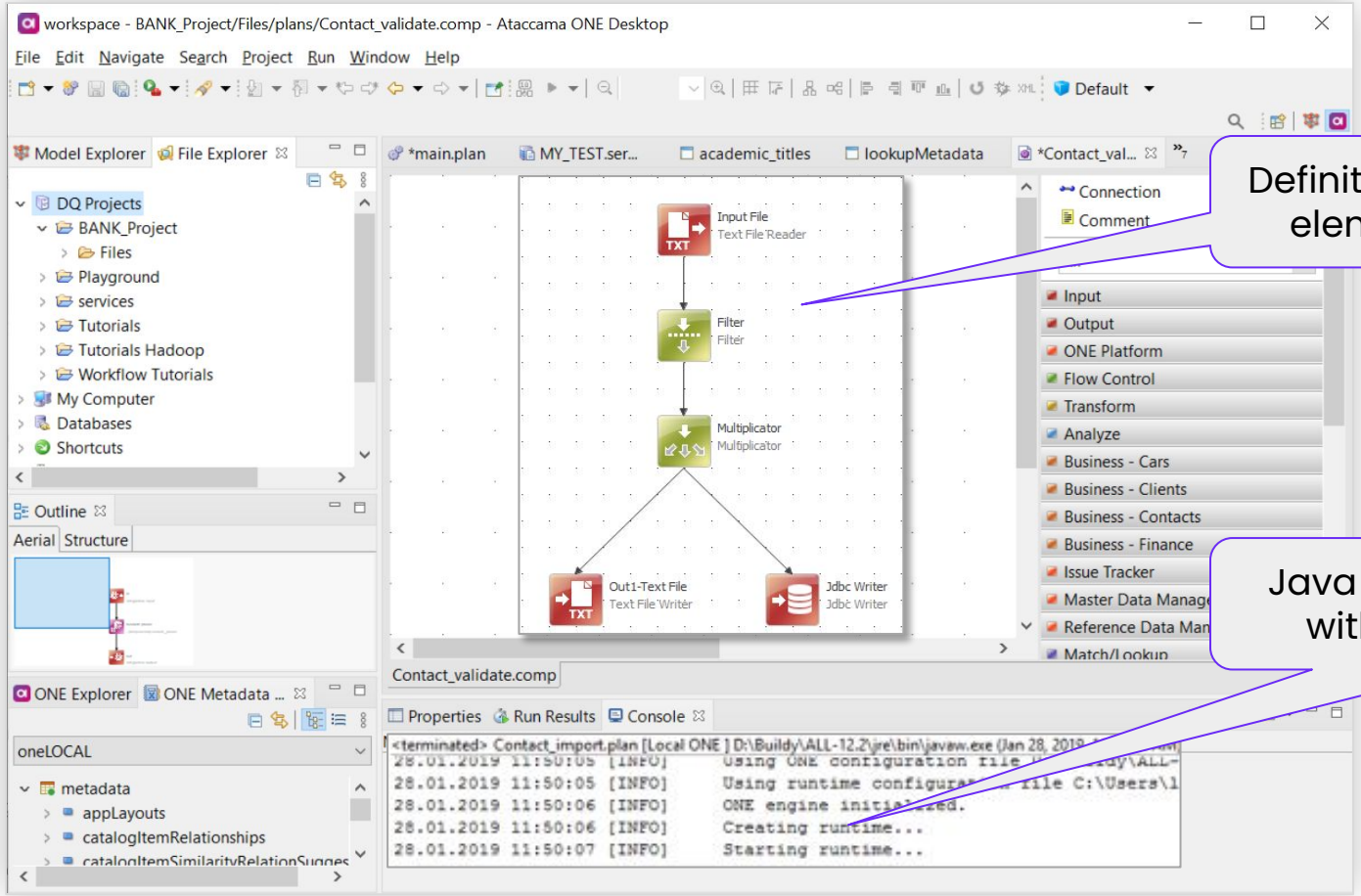


ataccama

ONE Desktop:
Plans

Plans can be described as a canvas where all Steps are placed and linked together.

main.**plan**



Definition of the plan's logical elements and their order

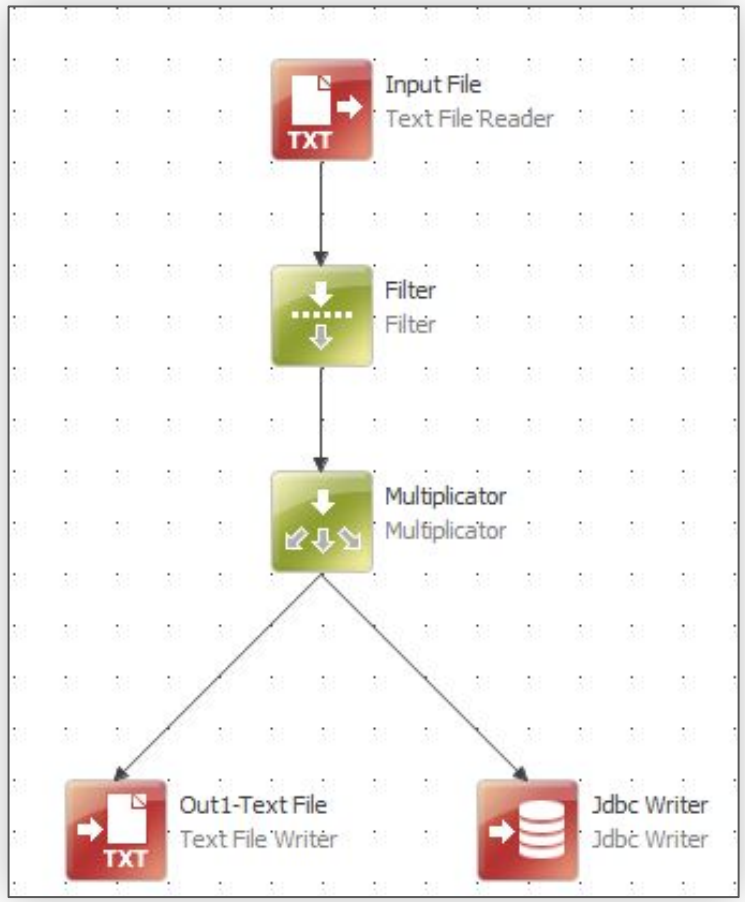
Java runtime execution log with processing details

ataccama

ONE Desktop: Step Types

Steps are divided into the following categories:

- Input & Output
- Flow Control
- Transform
- Analyze
- Business rules
- Issue Tracking
- Matching & Lookup
- Text analytics
- Maintenance



ataccama

ONE Desktop: Commonly Used Steps

The most useful steps to begin with:

Input / Output

- Text File Reader/Writer
- JDBC Reader/Writer
- Excel File Reader/Writer
- Integration Input/Output

- XML Reader/Writer
- Random Record Generator

Flow Control

- Alter Format
- Multiplicator
- Filter
- Extract Filter
- Condition
- Union & Union Same
- Join

Transformations

- Column Assigner
- SQL Select / Execute
- Matching Values
- Apply Replacements
- Regex Matching

- Splitter
- JSON & XML Parser
- Pattern Parser

Analysis

- Profiling
- Simple Scoring
- Data Quality Indicator
- Group Aggregator

- Schema Discovery

Matching/Lookups

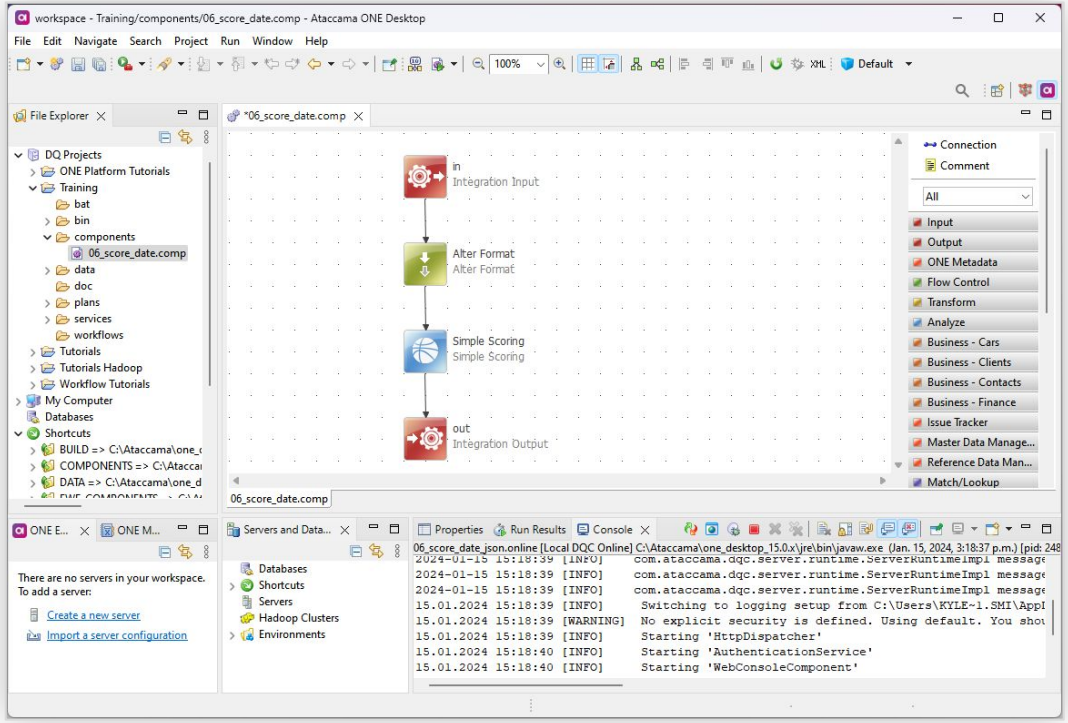
- Lookup & Lookup Builder
- Representative Creator
- Matching
- SOAP / JSON call

- WEB Lookup
- Canopy Clustering

ataccama

Topic Highlights

- **ONE Desktop** is an **Eclipse-based IDE** designed for creating DQG solutions based on Ataccama products.
- **Projects** are sets of plans, components, workflows, and data, which are logically organized into folders.
- **Plans** define the logic and rules that are applied to data in order to produce the desired output.
- **Steps** perform specific functions or algorithms to data with relation to other **Steps** within plans, components, or workflows.



ataccama

Memory Refresher #1 Introduction to ONE Desktop

ataccama

Which of the following statements about ONE Desktop is True?

1. Simple, Complex, and Empty projects are the main project types.
2. There are two types of projects: project and model project.
3. ONE Desktop can not be used without any web-applications.

Match the terms with their meanings:

A. Plans

B. Steps

C. Components

C. Components

B. Steps

A. Plans

1. Algorithms that can be reused in different plans.

2. Algorithms for reading, transforming, and analyzing data.

3. Defines the logic by placing elements and connecting them together.

ONE Desktop Demonstration

ataccama

Lab Exercise #1 Environment Preparations

ataccama

Read / Write Operations

ataccama

Data Sources: Traditional

Steps to read data from several sources:

- **Text File Reader** — **csv and txt (incl. compressed files)**
- **JDBC Reader** — **common DB systems, anything with a JDBC Driver**
- **Excel File Reader** — **.xls and .xlsx files**
- **XML File Reader** — **flat or hierarchical XML files**
- **JSON File Reader** — flat or hierarchical JSON files
- …and more

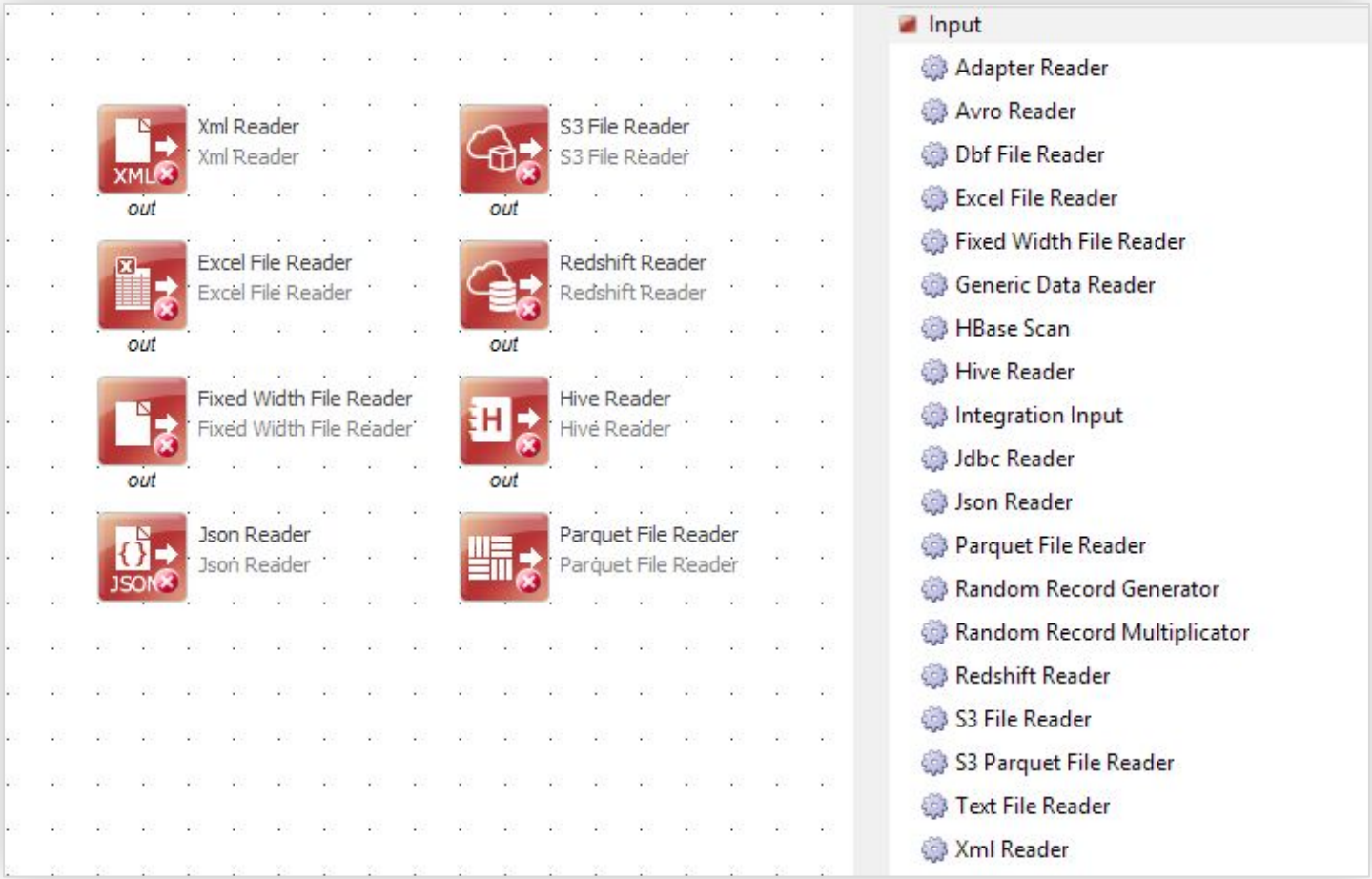
Steps to interact with the ONE Web Application:

- **ONE Metadata**


Steps to use in Components and Web Services:

- **Integration Input** and **Integration Output** can be used instead of other, more specialized, input steps to create a plan file that is usable as either a component or an online service.

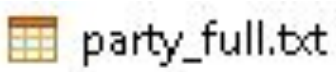
Other Reader Steps



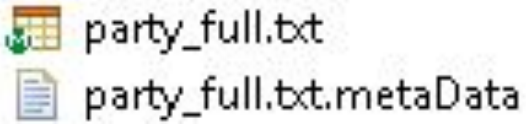
ataccama

Flat Files: Metadata

In ONE Desktop, a flat file (csv, txt) has no metadata by default.



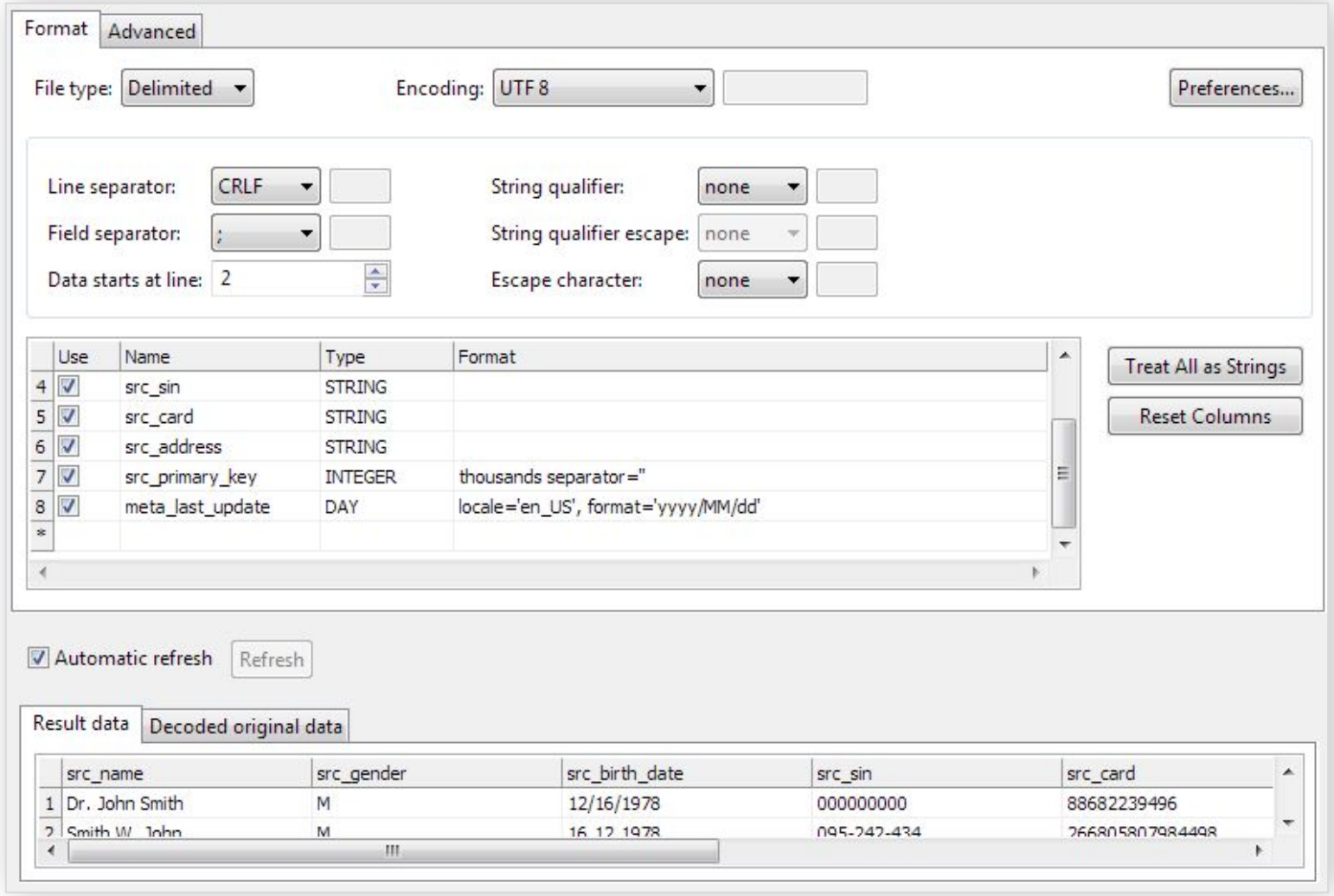
Once the metadata is created, the file gets a green „**M**“ (metadata) icon.



Metadata is stored by default as a hidden XML file.

To create or edit metadata, **double-click the file or use right-click > Edit Metadata** in the File Explorer.

Flat Files:
Metadata Editor

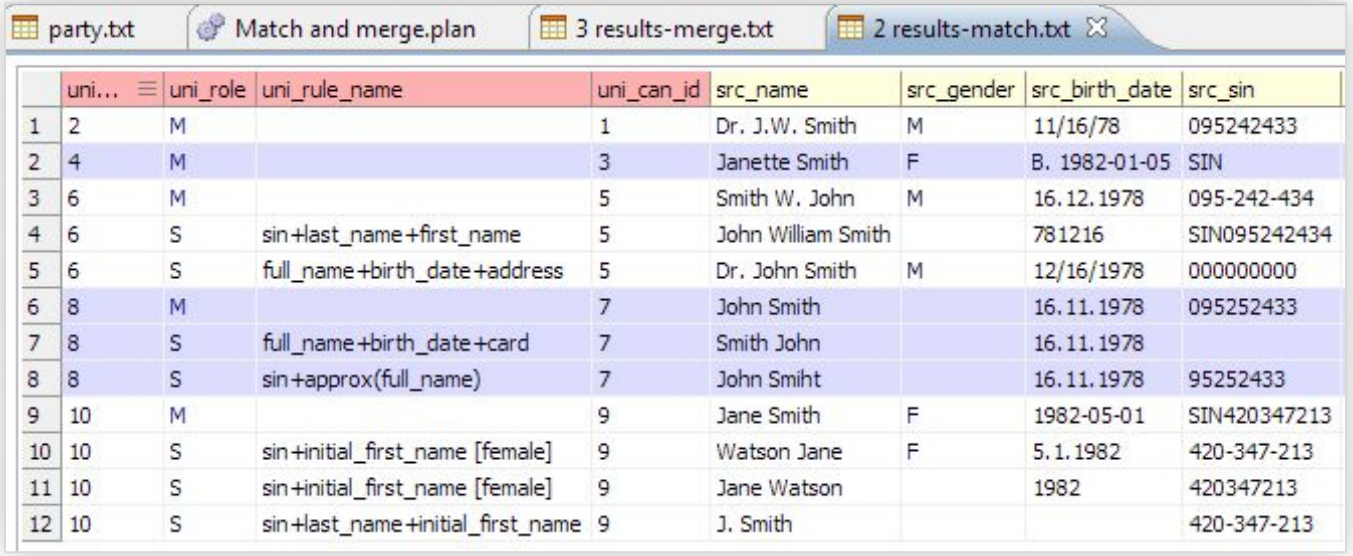


ataccama

Flat Files:
CSV Viewer

Flat files are opened in the CSV Viewer which allows sorting, filtering, coloring.

See “Getting started with Ataccama DQC” in **Help > Help Contents** for details.


party.txt | Match and merge.plan | 3 results-merge.txt | 2 results-match.txt

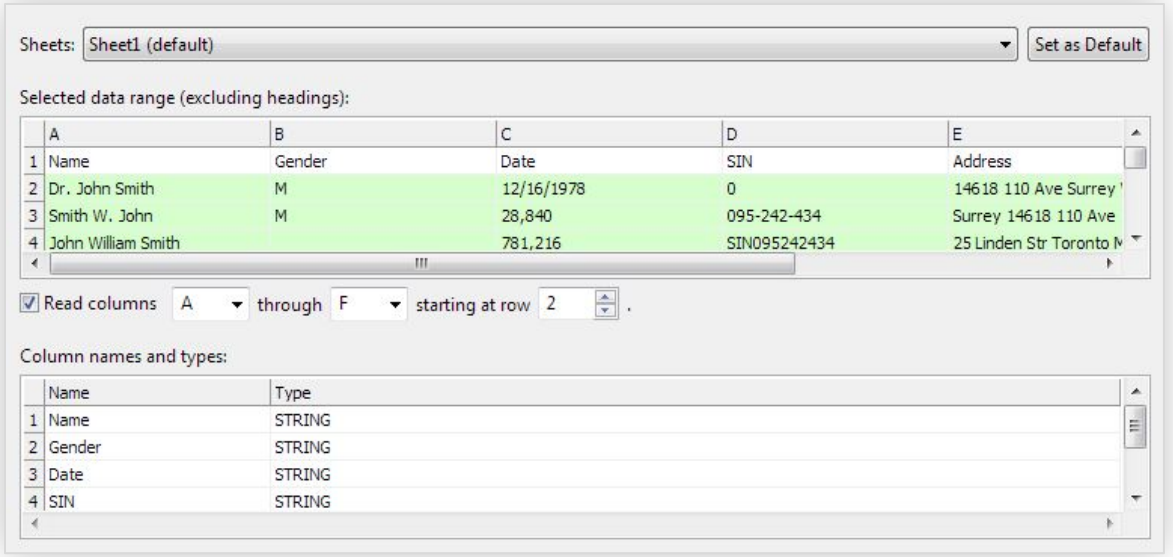
	uni...	uni_role	uni_rule_name	uni_can_id	src_name	src_gender	src_birth_date	src_sin
1	2	M		1	Dr. J.W. Smith	M	11/16/78	095242433
2	4	M		3	Janette Smith	F	B. 1982-01-05	SIN
3	6	M		5	Smith W. John	M	16.12.1978	095-242-434
4	6	S	sin+last_name+first_name	5	John William Smith		781216	SIN095242434
5	6	S	full_name+birth_date+address	5	Dr. John Smith	M	12/16/1978	000000000
6	8	M		7	John Smith		16.11.1978	095252433
7	8	S	full_name+birth_date+card	7	Smith John		16.11.1978	
8	8	S	sin+approx(full_name)	7	John Smiht		16.11.1978	95252433
9	10	M		9	Jane Smith	F	1982-05-01	SIN420347213
10	10	S	sin+initial_first_name [female]	9	Watson Jane	F	5.1.1982	420-347-213
11	10	S	sin+initial_first_name [female]	9	Jane Watson		1982	420347213
12	10	S	sin+last_name+initial_first_name	9	J. Smith			420-347-213

ataccama

Excel Files: Metadata

To create metadata for an Excel file:

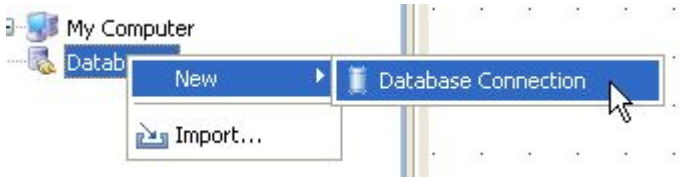
- Right-click the file and choose '**Edit Metadata**'.
- Drag & drop the file onto a plan.

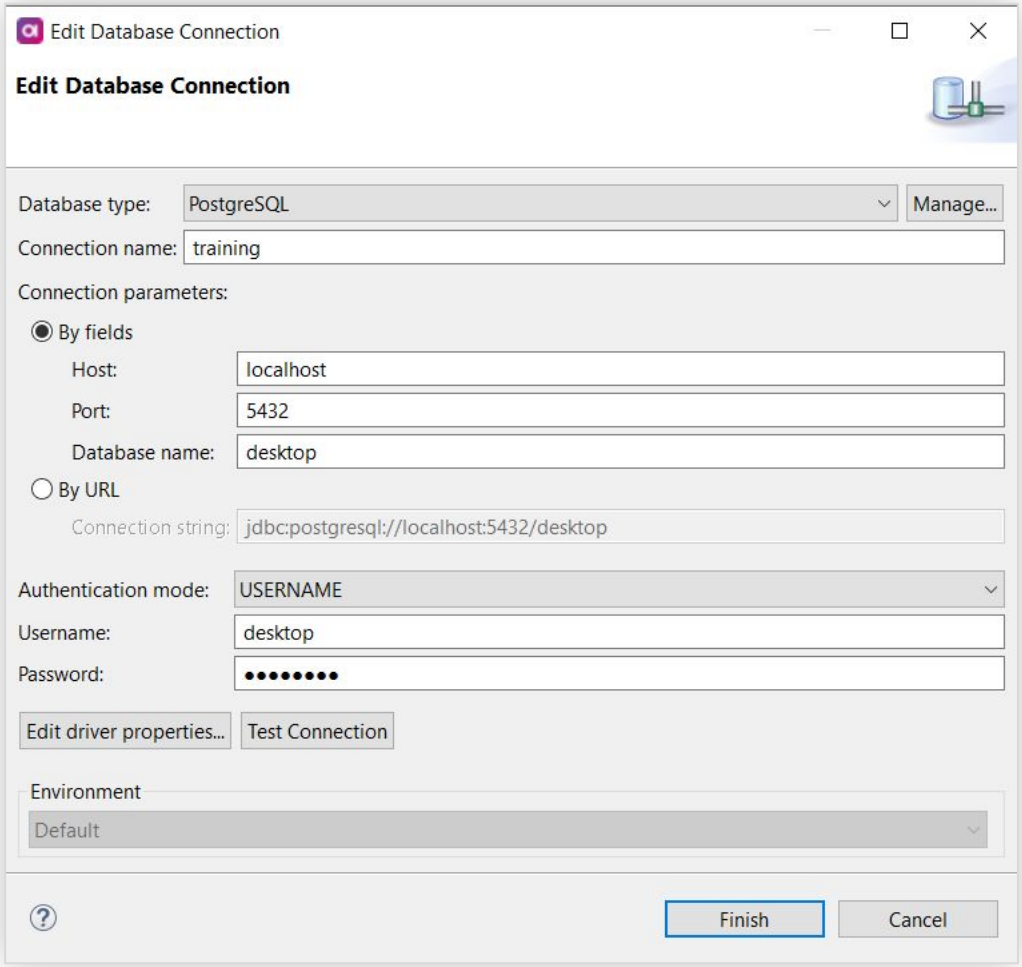


ataccama

Database Connections: Creation

ONE Desktop supports direct connection to supported DBs via the File Explorer.

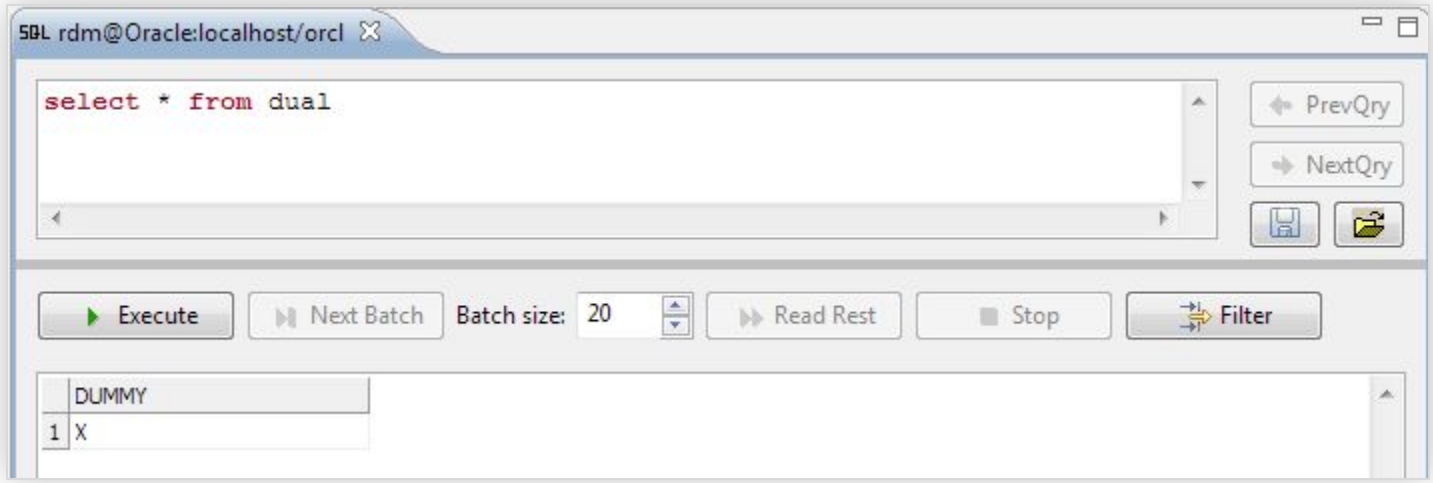


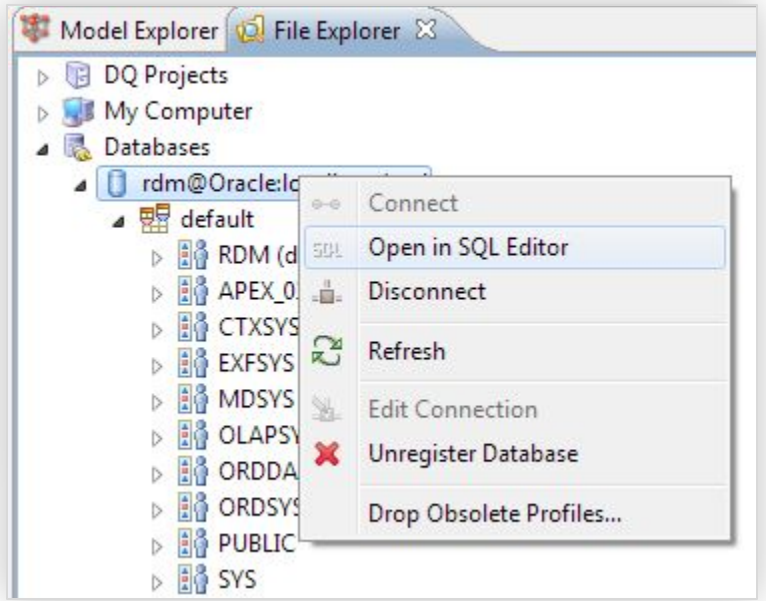


ataccama

Database Connections: SQL Editor

- Simple SQL queries can be executed directly within the ONE Desktop.
- Double-click on a table to open it in SQL Editor (or right-click > Open in SQL Editor).
- The SQL query field offers content assist (CTRL + Space).

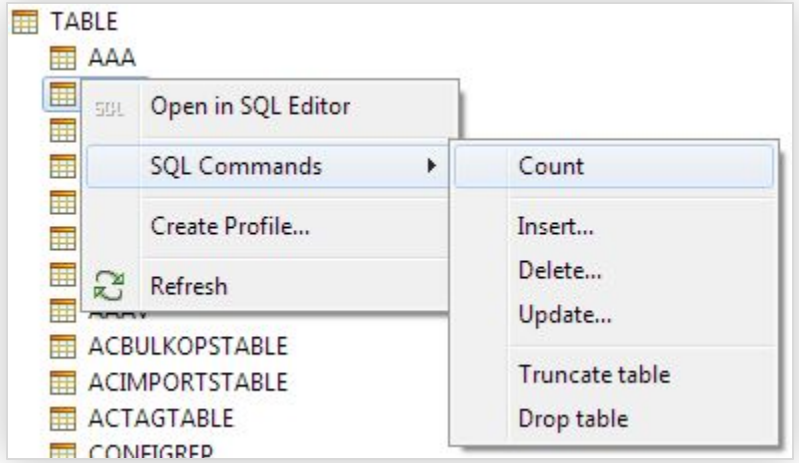




ataccama

Database Connections: Custom Commands

- Right-click a table and choose **SQL Commands** to execute common queries.
- Custom SQL Commands can be configured in **Preferences > ONE Desktop > Database > Table Commands**.



ataccama

Data Sources:
Creating Reader Steps

Drag & drop a file (text or Excel) or a database table into a plan to create the appropriate **Reader Step** (Text File/Excel/Jdbc).

- For **files**, the reader step is configured based on the metadata.
- For **DB tables**, a configuration wizard is shown to help set up the reader step.

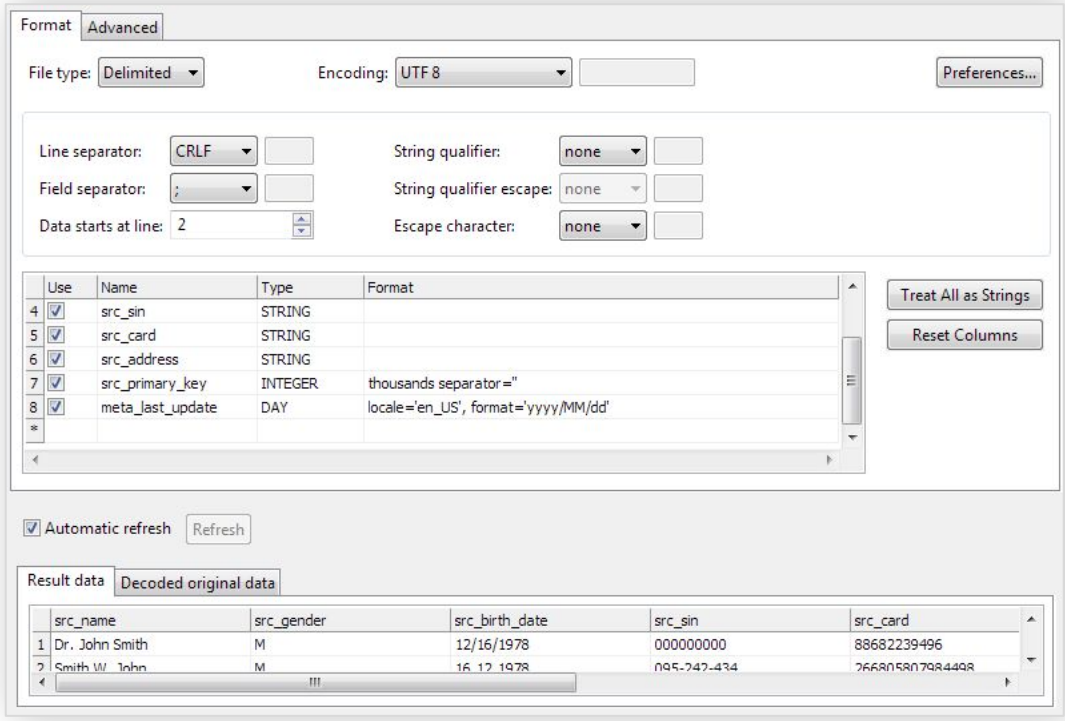
Data Output

ONE Desktop can write data in several ways:

ataccama

Topic Highlights

- **Metadata** describes the format of the data, primarily for **CSV**, **TXT**, and **XLSX** files, for properly using in plans.
- Text files can be opened in the **CSV Viewer** by double-clicking. Clicking the name of the column in the header row will sort that column.
- A data file can be directly added to a plan by **dragging & dropping** it into the plan's canvas, where the corresponding reader step will be generated.



ataccama

Memory Refresher #2 Read / Write Operations

ataccama

Metadata must be created before a data file can be added to a plan.

- TRUE
- FALSE

ataccama

There are Reader steps that will allow input from and output to the ONE Web Application.

- TRUE
- FALSE

ataccama

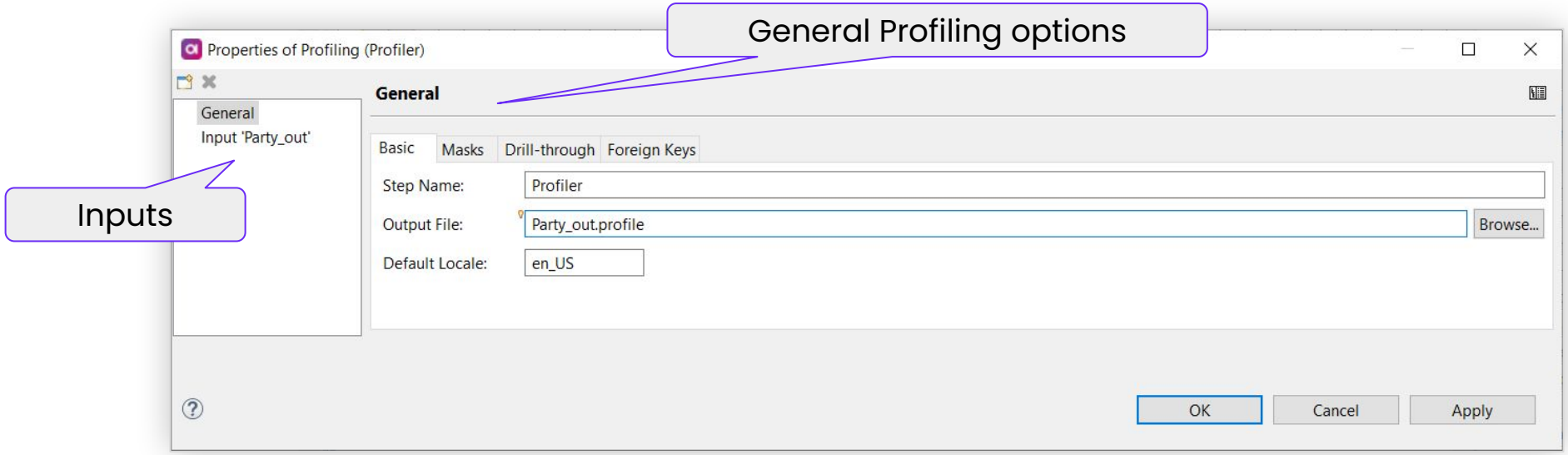
Data Profiling

ataccama

Profiling Step

The Profiling step provides its outputs directly to the ONE Desktop Environment. It serves primarily for data analysis.

- Generates a special file '*filename.**profile***' which contains the profiling results.
- Besides common statistical analysis, the **Profiling** step also allows highly advanced analysis.
- Columns can be automatically mapped with the '**Fill Columns**' button.
- **Drill-through** enables quick access to sliced data; limits can be set for working with large volumes of data.



ataccama

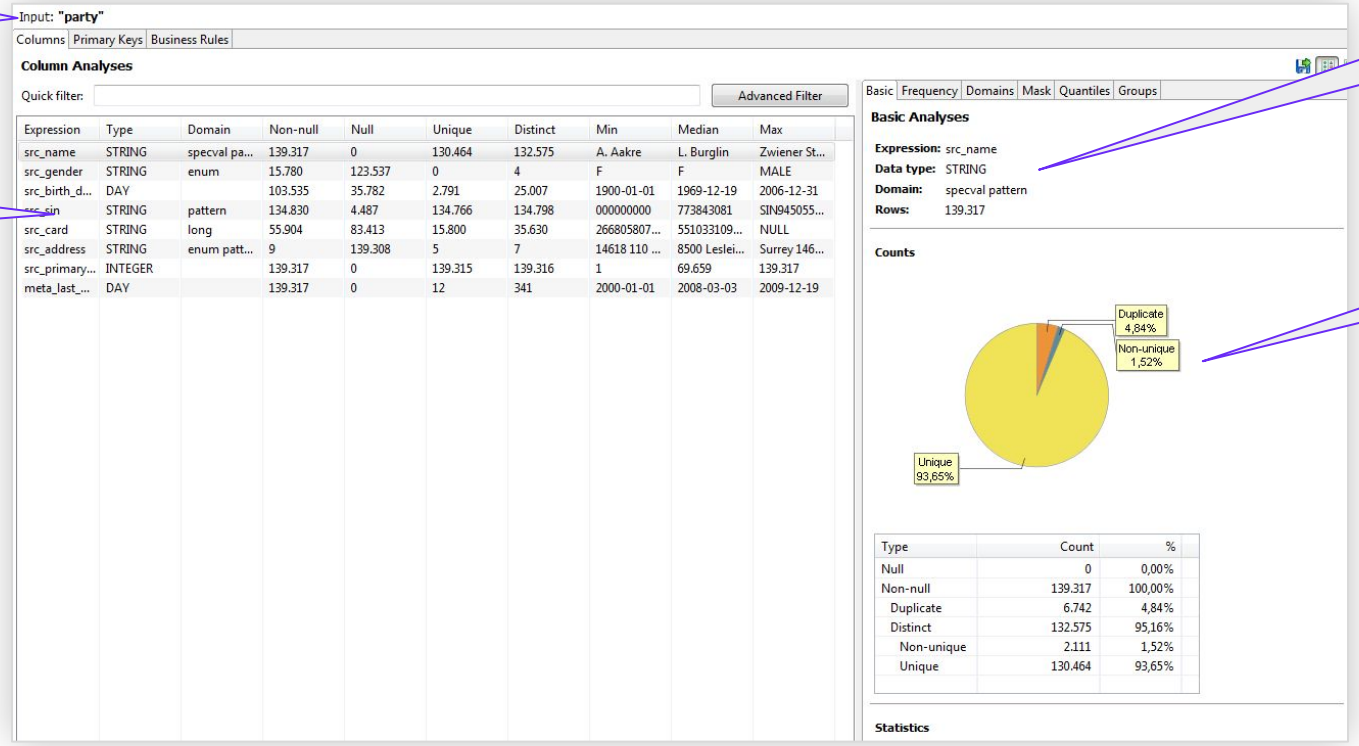
Statistical Analysis

The ONE Desktop IDE provides common statistical analysis as a simple, one-click operation.

- Profile files can be created for inputs from a text file or a DB table.
- Right-click the source file/table and choose Create Profile in the contextual menu.

Source-specific analysis

Field-specific analysis



Other statistical values

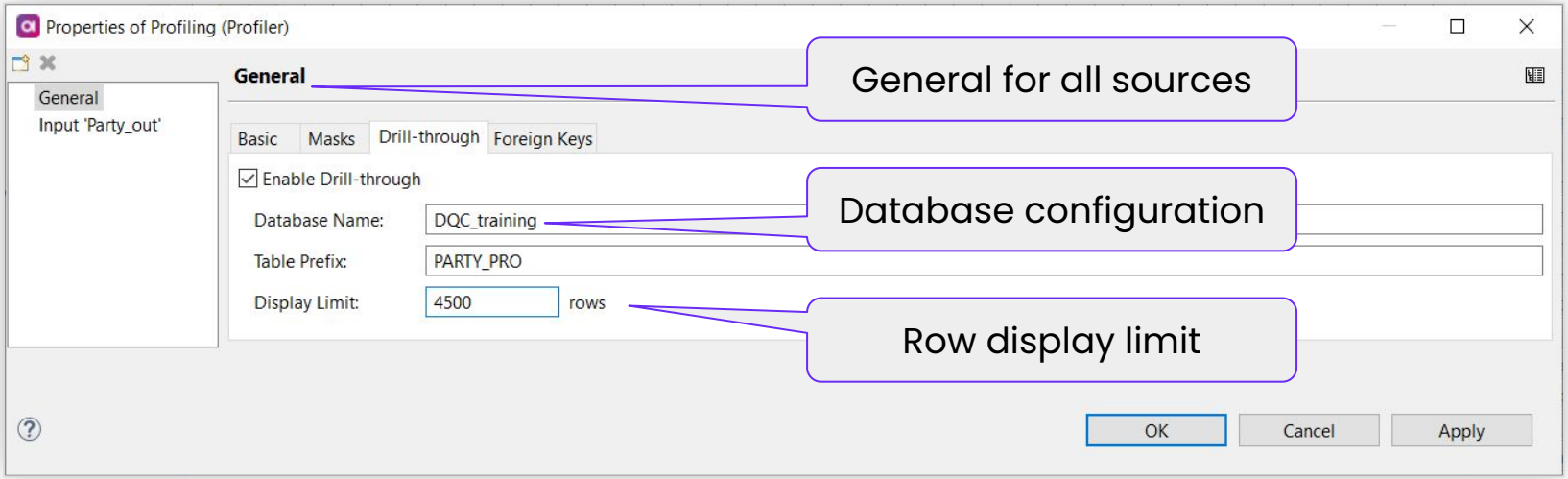
General statistics

ataccama

Drill-Through Profiling

The drill-through feature can be enabled for any profile.

- Tables will be created in a DB with different prefixes for different profiles.
- Row limit can be set to avoid overloading the DB with huge data volumes.



Right-clicking on any value (analysis result) from statistics, business rules, primary key analysis, dependency analysis, etc. will offer you the possibility to drill-through and display the corresponding values.

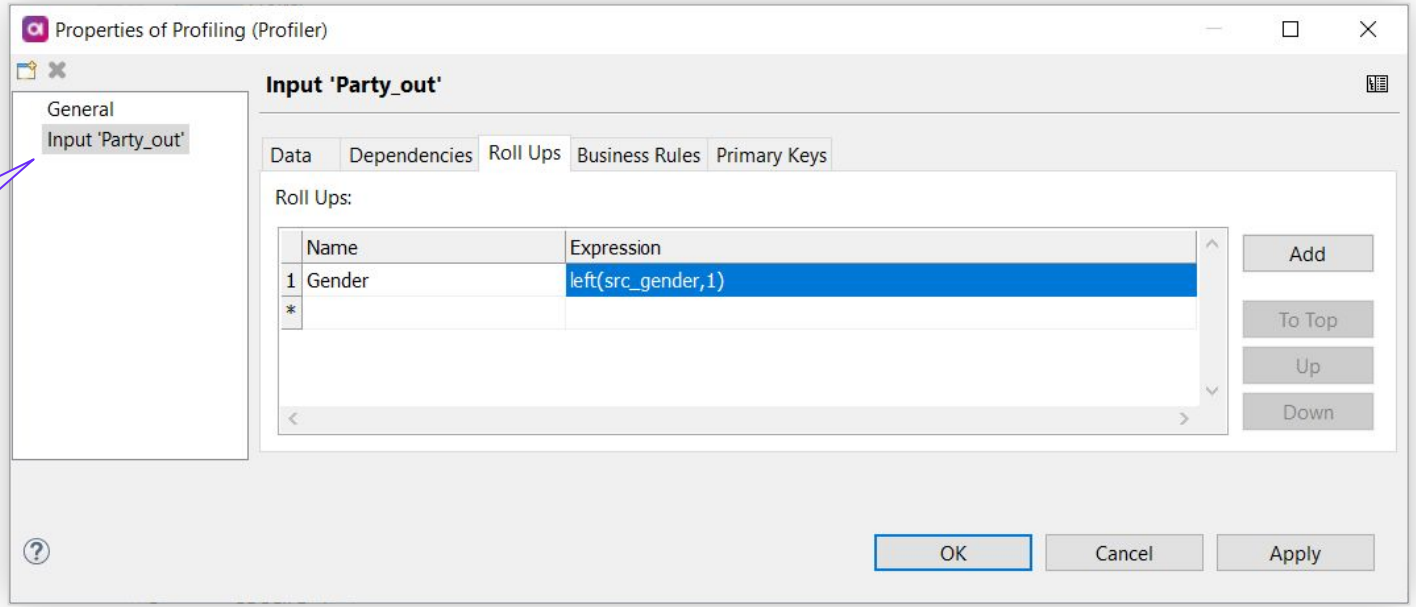
- EXAMPLE: This feature can be used to find all invalid (FALSE) results of a business rule

ataccama

Roll-Ups Profiling

Roll-ups allow you to create a kind of sub-profile within a profile.

- Example: for a group of people, you can create a roll-up by gender (if this value is available). A sub-profile for males and females will then be created within the common profile.
- A roll-up can be a particular column value or any expression, such as a function applied to a column value
- Hierarchies of roll-ups can be created

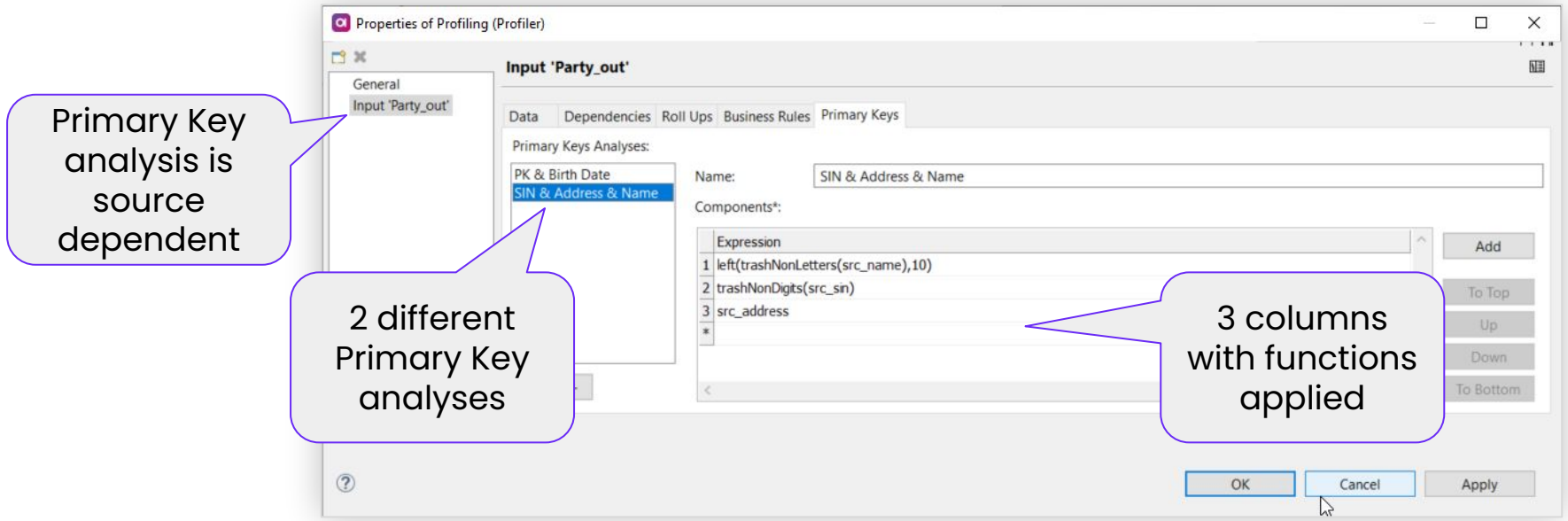


ataccama

Primary Key Analysis

Various primary key analyses can be created:

- Primary key can be a column, set of columns, or any expression or function applied to columns.



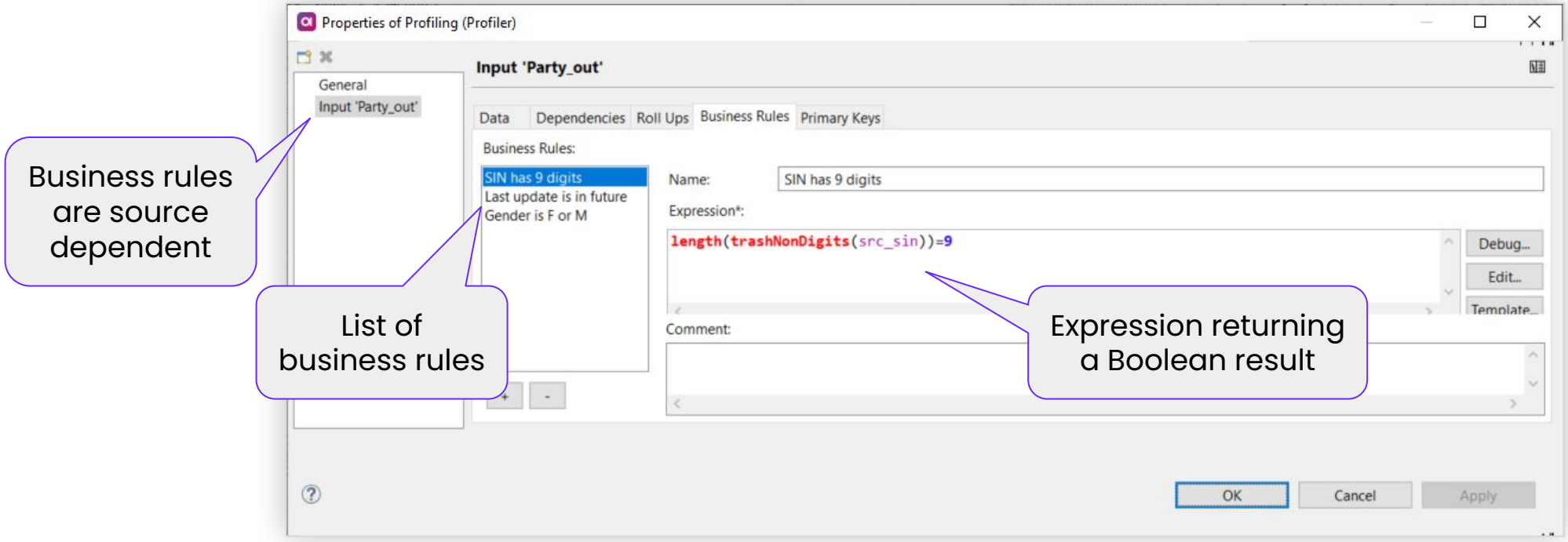
Primary key analysis ensures the uniqueness of identifiers that might be primary keys.

- i.e. reading from a DB without constraints.

ataccama

Business Rules

- Business rules are statements which can be evaluated as either **true** or **false**.
- All business rules are evaluated for each record.
- There is no limit to the complexity of business rules expressions, as long as they return a Boolean value at the end.

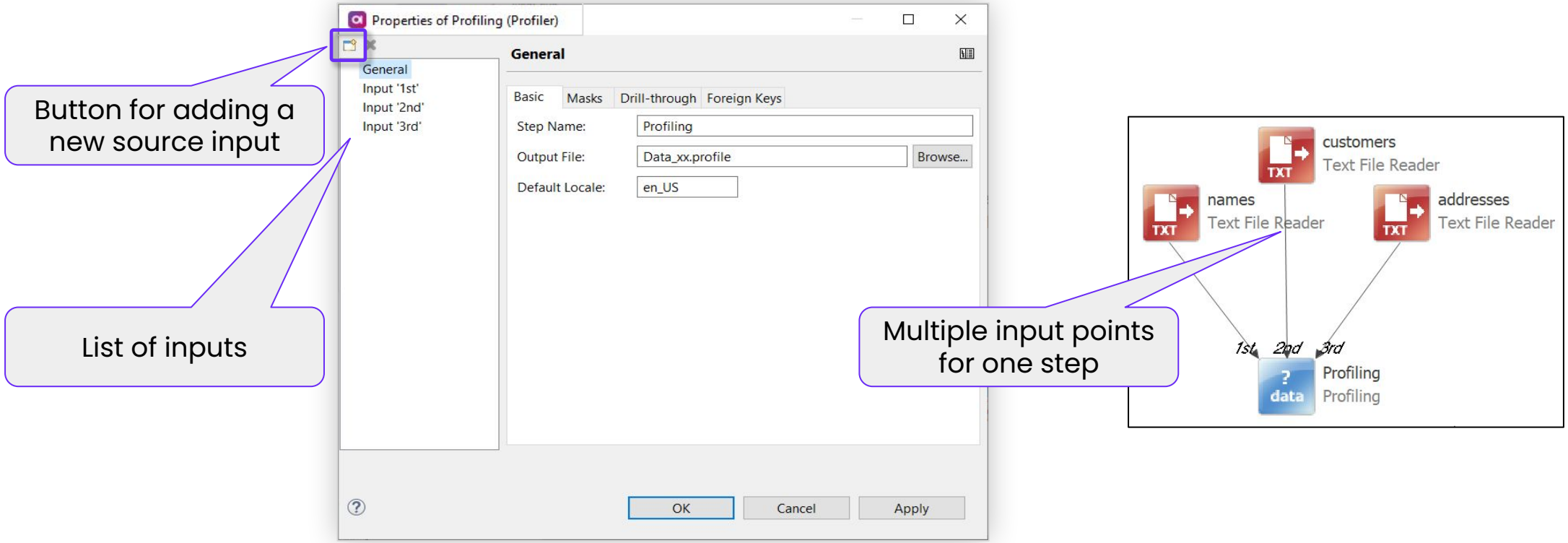


ataccama

Multi-Table Profiling

The Profiling step can read from multiple input sources.

- Each source gets analyzed independently, but cross-source analysis is possible as well (e.g. foreign key analysis)

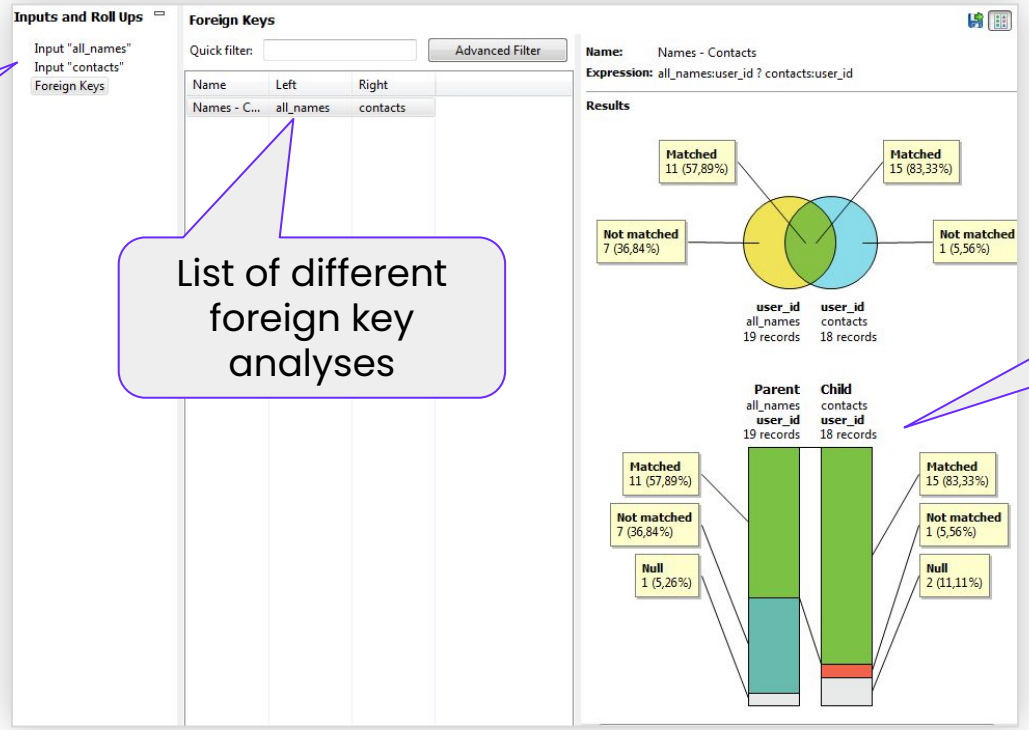


ataccama

Foreign Key Analysis

Analysis between two of the sources going into the Profiling step.

- Requires definition of both the parent key and the child key.
- Graphic representation of relationship statistics (matched, non-matched, null in the parent, etc.).



Dependent on two different sources

List of different foreign key analyses

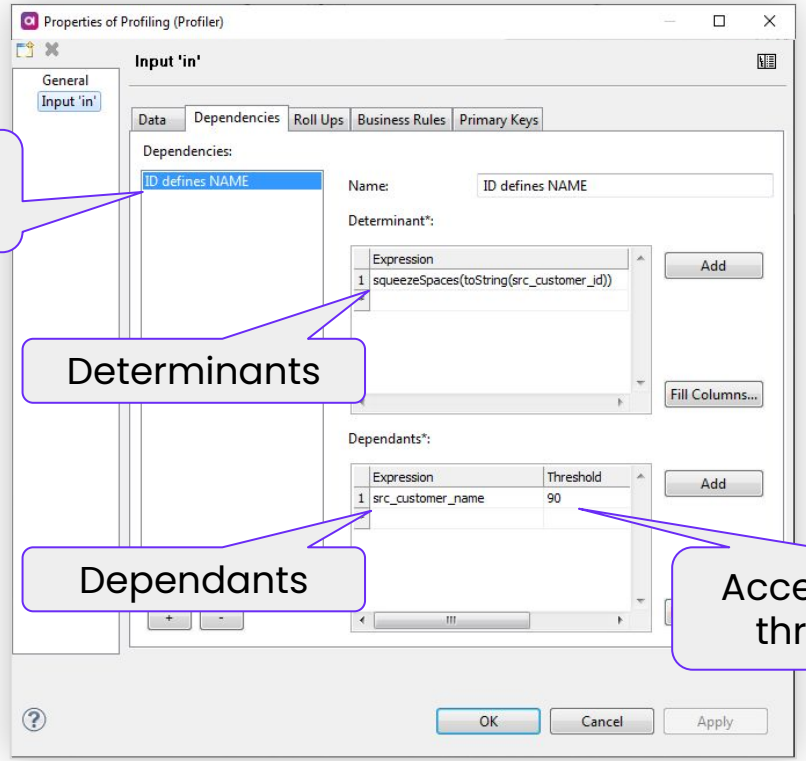
Graphical representation of the key values

ataccama

Dependency Analysis

Searches for dependencies between determinant and dependent values.

- **Determinant** and **Dependant** can be column values or any expression.
- You can set thresholds for what is accepted as a dependency and what isn't.

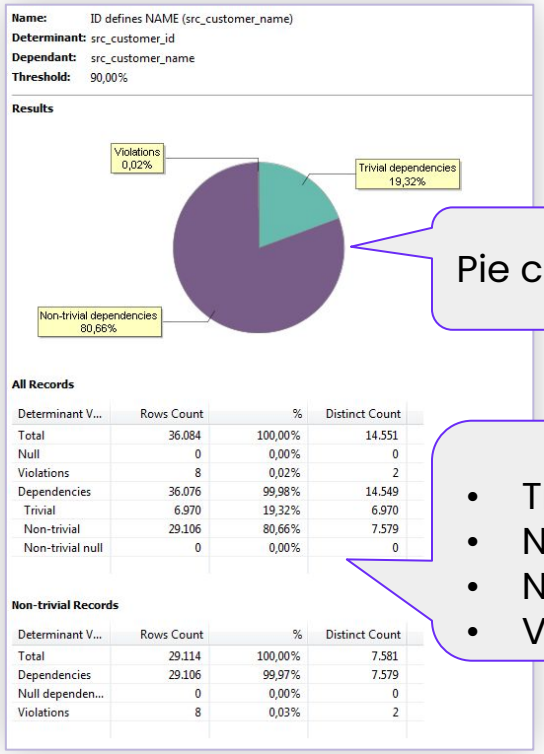


Dependencies list

Determinants

Dependants

Acceptability threshold



Pie chart with results

Counts on:

- Trivial
- Non-trivial
- Null
- Violations

ataccama

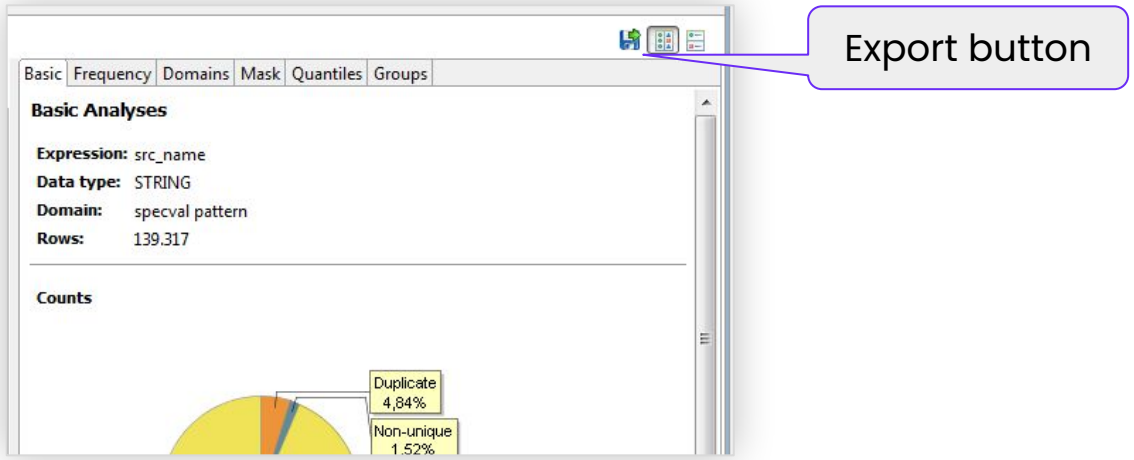
Sharing Results

Results can be simply shared via the profile files.

- Profile files have a very small footprint as they only store aggregations and statistical information, not the data itself.

ONE Desktop plans can also be shared.

- Plans are abstractions of the ONE Desktop GUI constructed from an XML file.
- XML files can be stored in a VCS and allow access to all parties.
- Because it is a plain text file, it is easy to keep track of changes and document them.



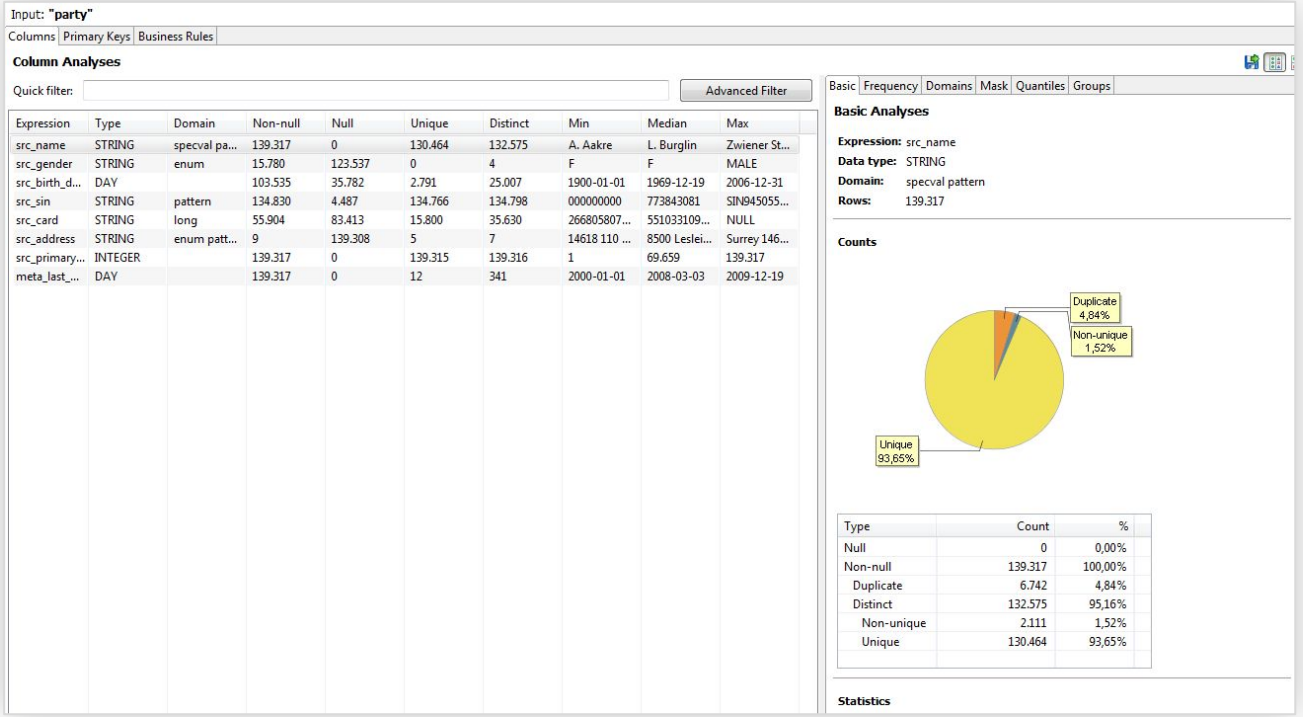
Profiles can also be exported to HTML via the Export button

- Allows for partial export
- Export to HTML or XML
- Can be auto-zipped

ataccama

Topic Highlights

- Profiling provides the user with a **summary of basic statistical information**: unique values, duplicates, dependencies, frequency analysis, masks, and patterns.
- Allows the evaluation of business rules and observation of patterns in data, **to determine future steps for data cleansing**.
- Provides a **drill-through** feature, allowing quick access to sliced data samplings.



ataccama

Memory Refresher #3 Data Profiling

ataccama

Match the terms with their meanings:

A. Frequency Analysis

B. Mask Analysis

C. Quantiles

D. Groups

D. Groups	*1. Shows the number of times that each non-null frequency count is repeated in the selected column.*
C. Quantiles	*2. Displays the data values that occur at designated intervals in the ordered data set.*
B. Mask Analysis	*3. Shows the syntactic patterns of the data.*
A. Frequency Analysis	*4. Shows the number of times each value occurs in the data.*

Lab Exercise #2
Data Profiling

ataccama

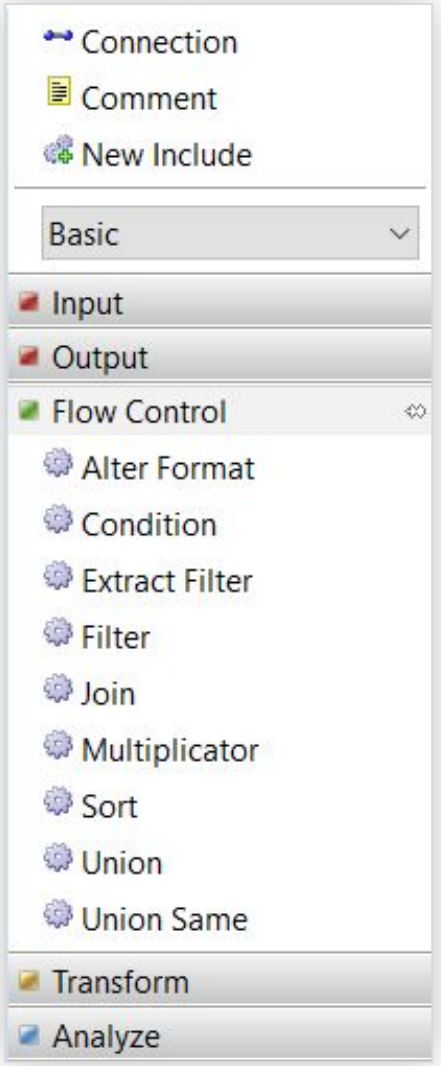
Flow Control

ataccama

Flow Control Steps

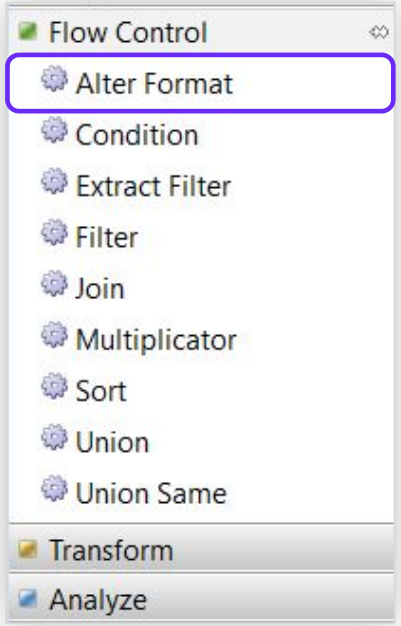
Flow control steps can be used to:

- Add or remove columns from the data flow.
- Split the data flow.
- Extract or segregate certain data sets.
- Join two or more data flows.



ataccama

Alter Format Step

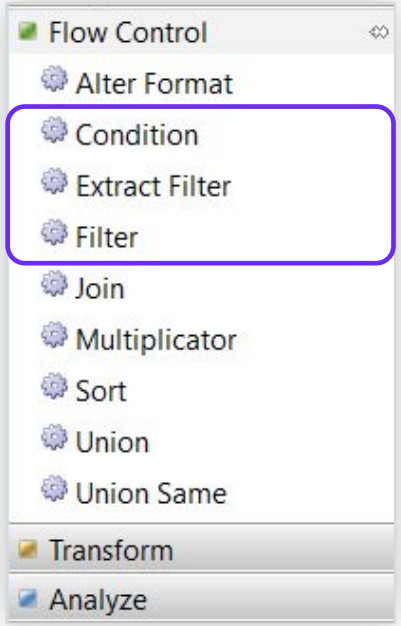


Alter Format	
	Adds or removes columns from the data flow. This step is used to include additional or temporary columns that are not present in the original source input data. Common usage is to have two **Alter Formats** in a plan – one at the beginning to create new columns and one at the end to remove the unwanted ones not designed for output. Typically used in every plan.

Alter Format Step

ataccama

More Flow Control Steps



Condition	
	Allows branching of the data flow – divides into '*true*' and '*false*' outputs based on the result of specified condition. Usually used to extract certain data set (such as bad records) for manual verification.
Filter	
	Filters out records which **do not satisfy** a given condition. The remaining records are lost.
Extract Filter	
	Conditional branching of the data flow (same as condition), but the original flow is left untouched (all records are still there). Useful when you need to leave the original flow intact.

ataccama

Condition Step

src_name	src_tier
John Smith	BR
Jane Watson	PT
J. Jones	AG

Condition*: `src_tier = "PT"`

src_name	src_tier
John Smith	BR
J. Jones	AG

src_name	src_tier
Jane Watson	PT

ataccama

Extract Filter Step

src_name	src_tier
John Smith	BR
Jane Watson	PT
J. Jones	AG

Condition*:

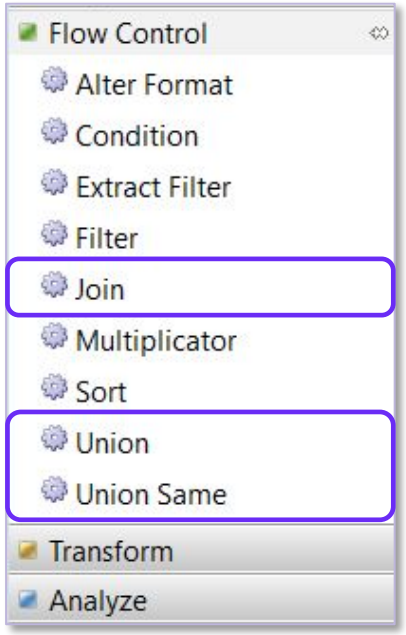
```
src_tier = "PT"
```

src_name	src_tier
John Smith	BR
Jane Watson	PT
J. Jones	AG

src_name	src_tier
Jane Watson	PT

ataccama

Other Flow Control Steps



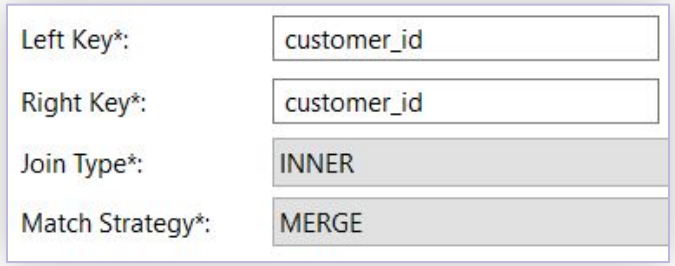
Join	
	Works in the same way as SQL JOIN. You must specify a key column and type of join (LEFT, RIGHT, INNER or OUTER).
Union	
	Works in the same way as SQL UNION. Apply a vertical merge of selected columns.
Union Same	
	Special type of **Union** step for working with identical data streams - all columns must have the same name and type.

ataccama

Join Step

customer_id	src_name
1	John Smith
2	George Kent
3	Jane Watson

customer_id	src_address
1	8500 Leslei St
3	20 St John Ave
3	1221 Queens Blvd
4	49 River Ave



customer_id	src_name	src_address
1	John Smith	8500 Leslei St
3	Jane Watson	20 St John Ave
3	Jane Watson	1221 Queens Blvd

+

ataccama

Union Step

src_name	src_address
John Smith	8500 Leslei St
George Kent	27 Town Square

src_name	residence
Jane Watson	20 St John Ave

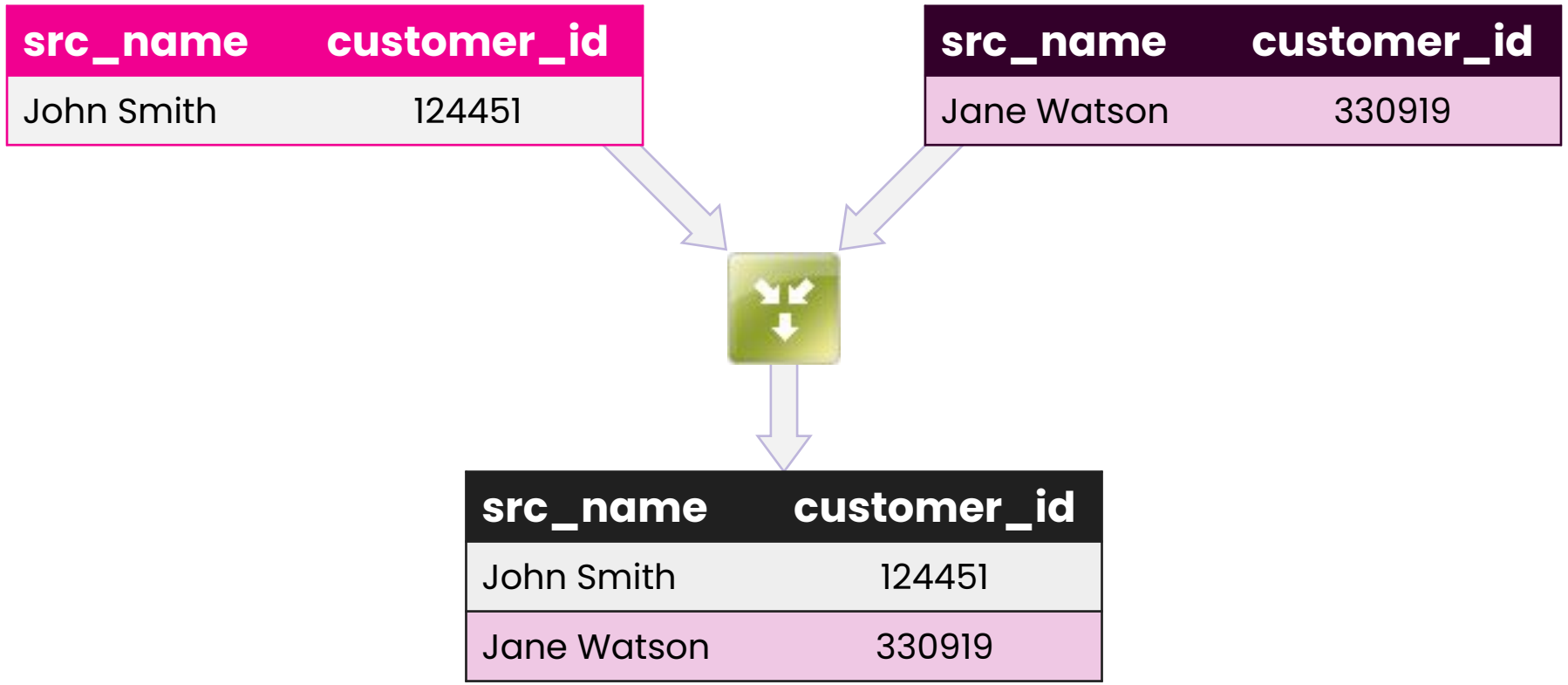
Column Mappings:

	Dest	In_a	In_b
1	src_name	src_name	src_name
2	address	src_address	residence
*			

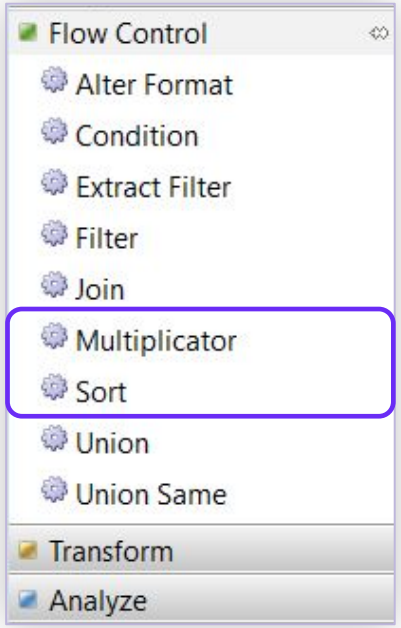
src_name	address
John Smith	8500 Leslei St
George Kent	27 Town Square
Jane Watson	20 St John Ave

ataccama

Union Same Step



Multiplicator & Sort Steps

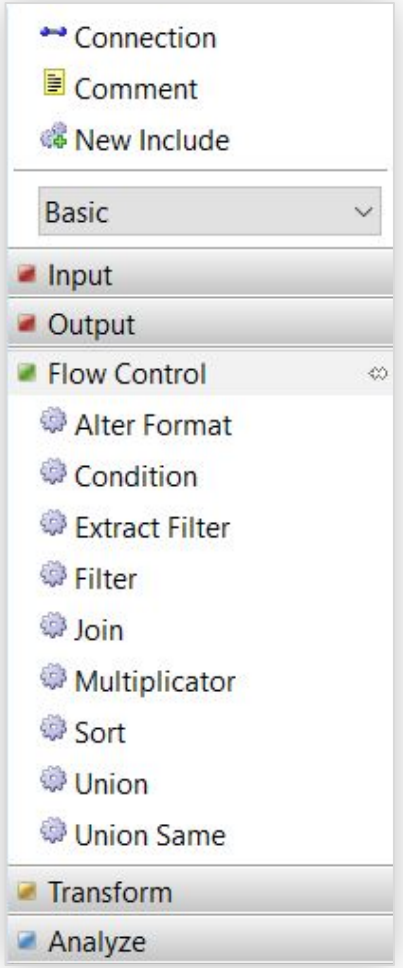


Multiplicator	
	Multiplies the data stream into more identical streams. Several connections can be connected from the '*out*' endpoint.
Sort	
	Sorts the data in the stream. Sometimes used before writing the output to a file.

ataccama

Topic Highlights

- Add or remove columns from the data flow (**Altar Format**).
- Split the data flow (**Multiplicator**).
- Extract or segregate certain data sets (**Condition**, **Filter**, **Extract Filter**).
- Join two or more data flows (**Join**, **Union**, **Union Same**).



ataccama

Memory Refresher #4 Flow Control

ataccama

Extract Filter retrieves records which pass a specific condition.

- TRUE
- FALSE

ataccama

Filter splits the data flow into two branches based on a logic expression.

- TRUE
- FALSE

ataccama

Union Same merges multiple data flows with the same format into a single data flow.

- TRUE
- FALSE

ataccama

Lab Exercise #3
Flow Control

ataccama

Best Practices & Conventions

ataccama

Data Quality Conventions

- **Understand your data!** The first step is always to analyze the data you have.
- **Never trust any data or any assertion.**
  - The data that was supposed to be perfect is often far from perfect.
  - Analyzing the data from all angles could shed light on new problems.
  - Assumptions are dangerous.
- Use the **Profiling** 

step to perform an initial analysis.
- Names of plans and components **shouldn't have spaces** – the best word separator to use is the underscore.
- Column name lengths should not exceed 30 characters (including prefix).

Naming Conventions

General Data Quality conventions dictate that:

- Source data should never be modified or transformed.
- A new set of columns (with prefixes – see below) should be created for each meaningful value in a data record .
- These columns will store transformed values, standardized values, data quality indicators, and audit information.

Basic column prefixes:

src_	The source value
pur_	A value used internally for transformations
std_	A value that has been standardized against a dictionary
cio_	The best possible value after all processing (STD or PUR)
mat_	A simplified version of data used for matching purposes only
dic_	A value that was verified in a dictionary

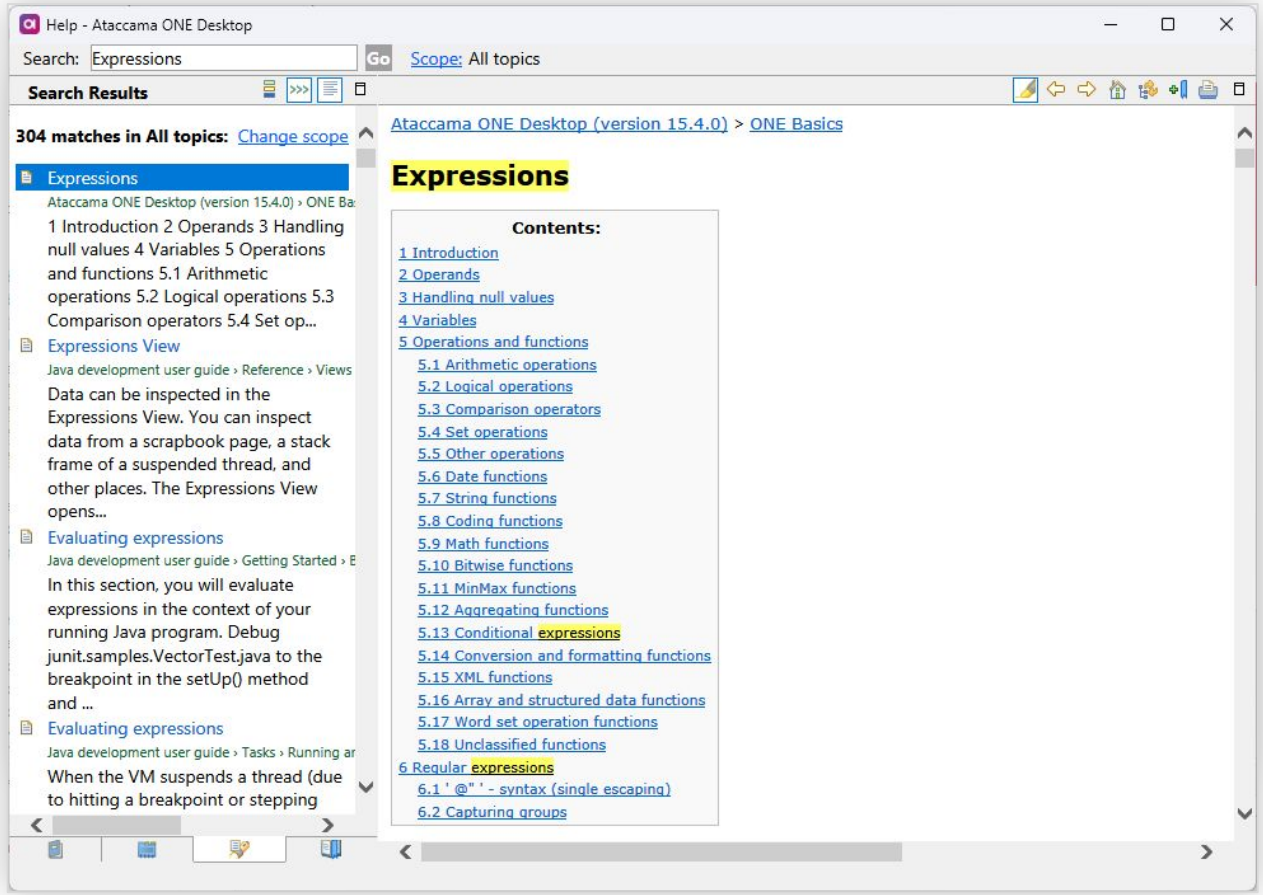
ataccama

Expressions

ataccama

Ataccama Expression Language

- Write your own transformations.
- Full reference in the built-in help.
- Tutorials (Using functions).
- Expression templates.



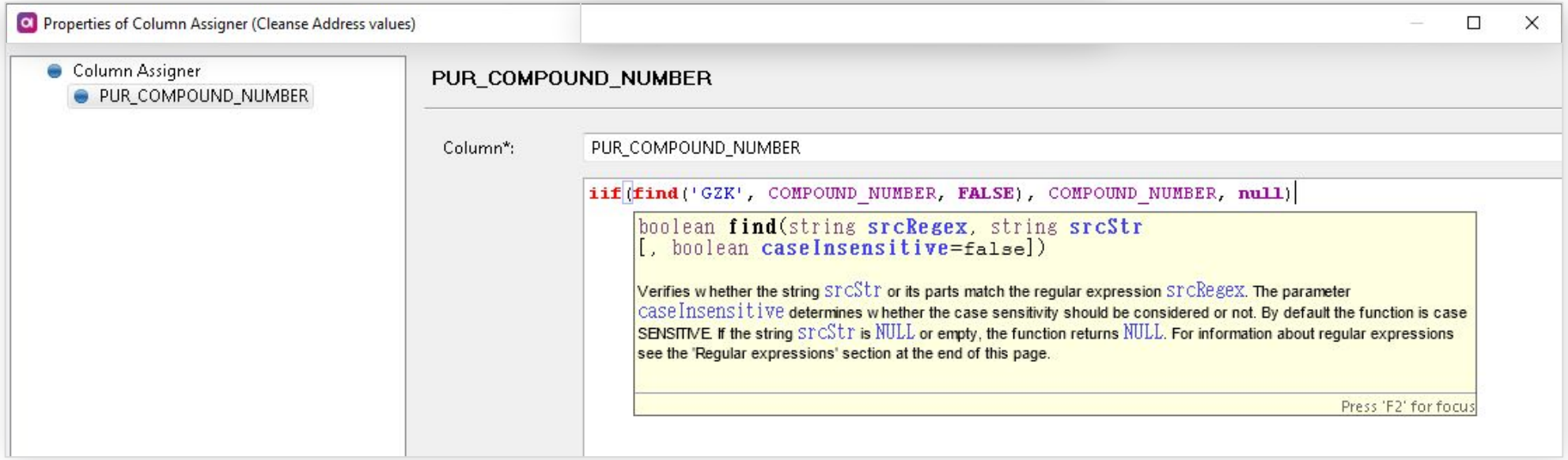
ataccama

Expressions Categories

- String functions.
- Date functions.
- Conditional expressions.
- Conversion and formatting functions.
- Math functions.
- Word set functions.
- And more; a complete list of expressions can be found in ONE Desktop's Help > Help Contents menu – enter the keyword "Expressions".

Expression Content Assist

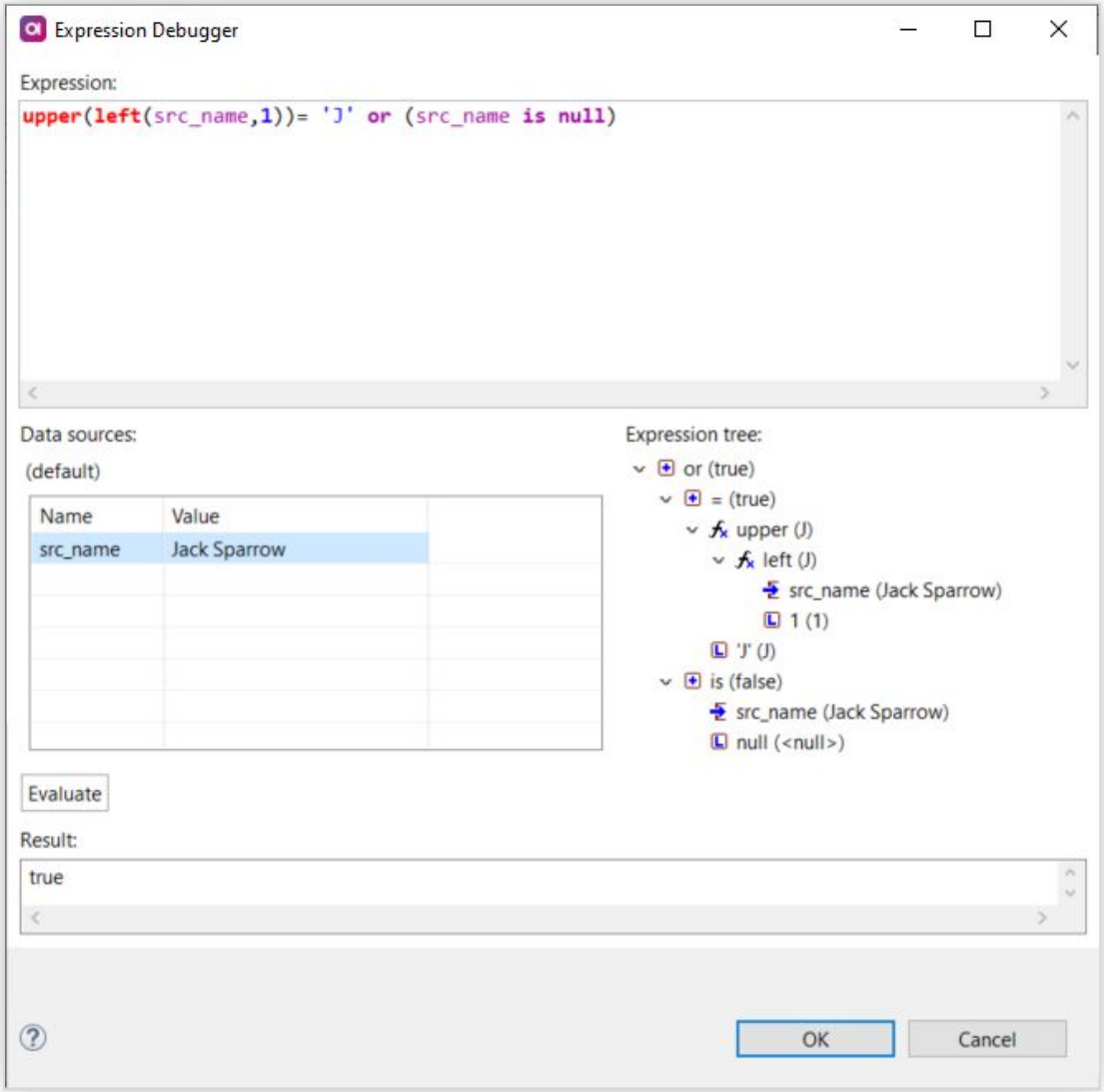
- Every time you write a function/expression in ONE Desktop, you can invoke content assist by pressing.
  - **CTRL + Spacebar**
- You can also place your cursor over the function name to see help for that function.



ataccama

Expression combinations

- Nesting expressions
  - **upper (left (src_string, 2)**
- Combining
  - **AND**, **OR**, **XOR**,...
- Evaluation priorities
  - visualized in Debug Expression Tree
- Conditionals - **If**, **Case**, **NVL**



ataccama

Expressions syntax

- In-built functions (length, right,..)
- Variables:
  - Simplify complex expressions.
  - Automatically typed (String, Integer).
- Evaluation:
  - The value of the last expression is returned.
  - Expression should produce a value matching the output column type.

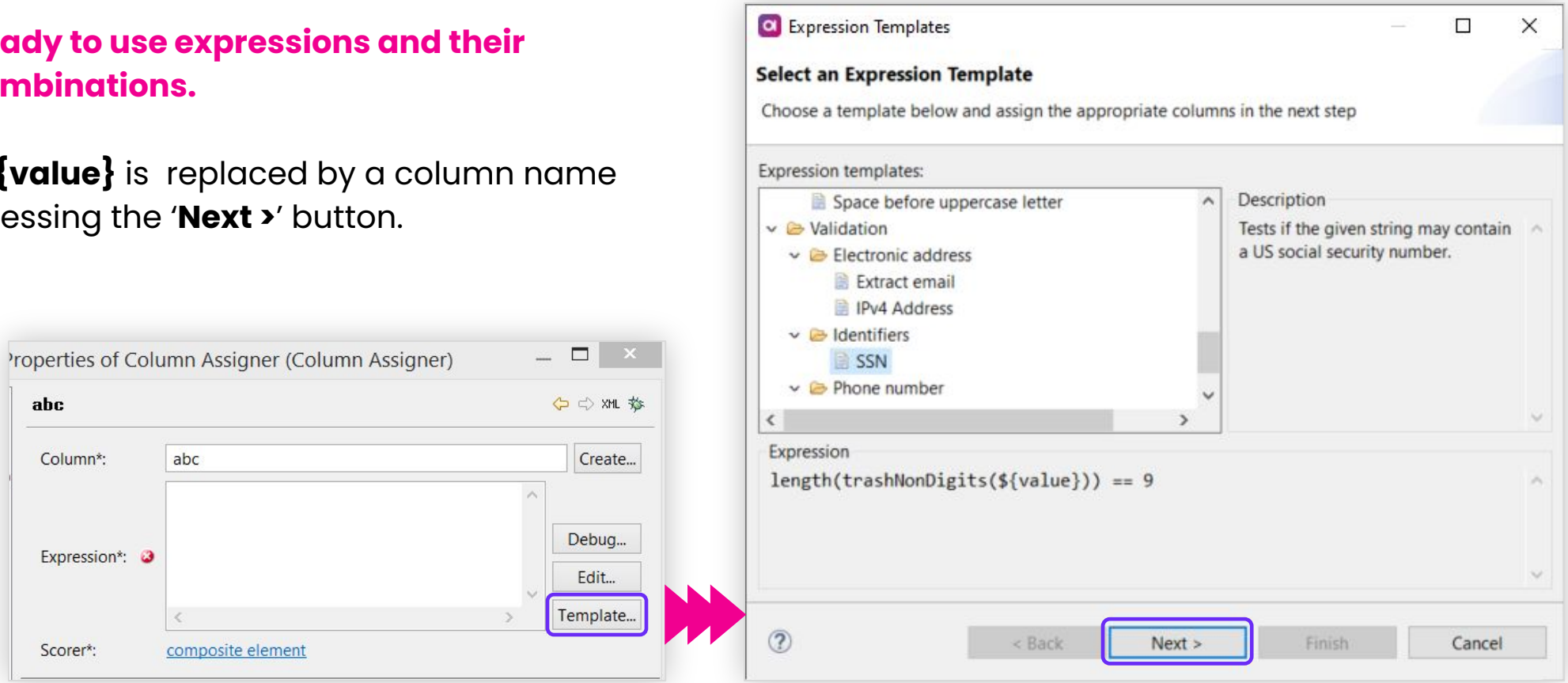
```
number_of_chars := 10;

src := src_num_as_string;

filler := '0';

iif(     //boolean predicate

    length(src)<number_of_chars and src is not
    null,

    //true

    right(replicate(filler, number_of_chars) +
    src, number_of_chars),

    //false

    Value

    )
```

ataccama

Expression Templates

- **Ready to use expressions and their combinations.**

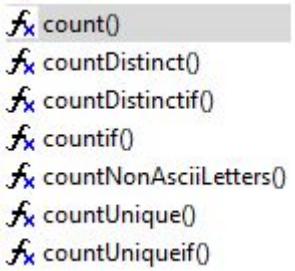
Note: **${value}** is replaced by a column name after pressing the '**Next >**' button.



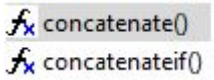
ataccama

Aggregations in Expressions

- **Context specific expressions (Aggregations)**
  - Count Functions

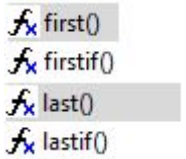


  - Concatenate

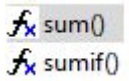


  - Averages

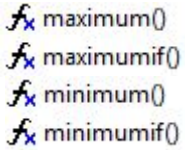
  - Selectors



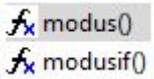
  - Sum



  - Min/Max



  - Modus



ataccama

Expressions on strings

Edit distance on string:

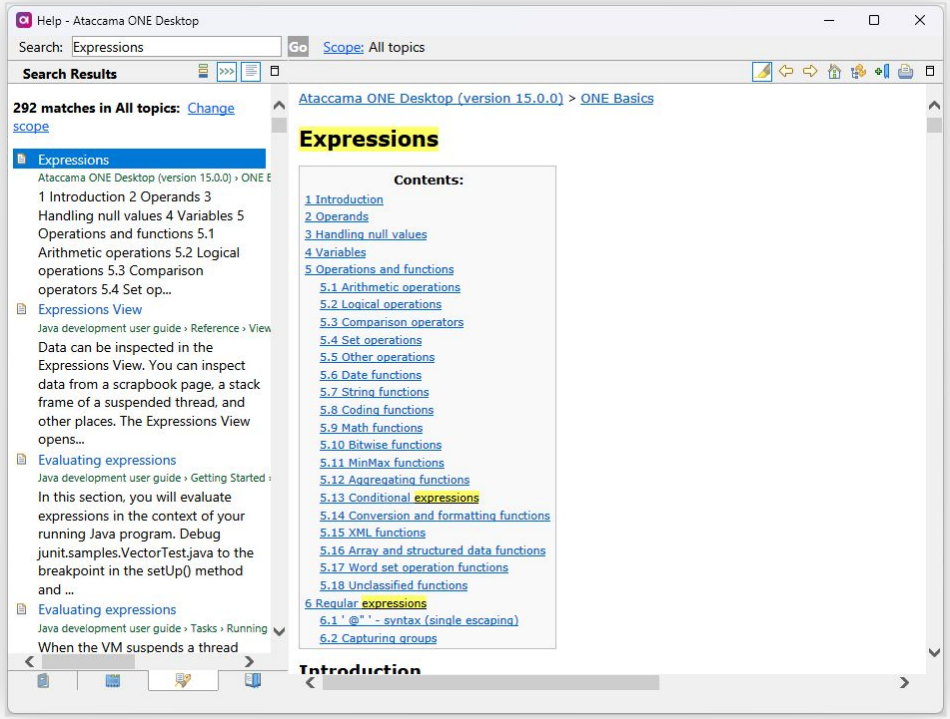
- Levenshtein (string difference - how many “typos”)

String set operations:

- Set functions accept string(s) and interpret them as a set of items:
  - **set.sort("Orange Banana Kiwi", " ") --> "Banana Kiwi Orange"**
- Union, difference, intersection of 2 strings
- Size, Sort
- MapExp, SumExp

Topic Highlights

- **Expression Content Assist** can be invoked assist by pressing **CTRL + Spacebar**.
- **Expression combinations**; expressions can be nested, combined with **AND**, **OR**, **XOR**, etc., and contain conditionals such as **IF**, **CASE**, **NVL**.
- **Expression Templates**, ready to use expressions and their combinations.



ataccama

Memory Refresher #5 Expressions

ataccama

Can Expression Templates be used as-is, without modification?

1. Yes.
2. No.

Expression Content
Assist can invoked by:

1. Hovering your mouse over the expression input field.
2. Pressing CTRL SPACEBAR on your keyboard.
3. Shouting loudly until a colleague comes to your assistance.

Can a logical expression contain inputs of different types, such as string or integer?

1. Yes.
2. No, all inputs must be of the same type
3. Yes, if the inputs are converted to the same type.

Lab Exercise #4
Data Quality Indicator

ataccama

Data Transformation

ataccama

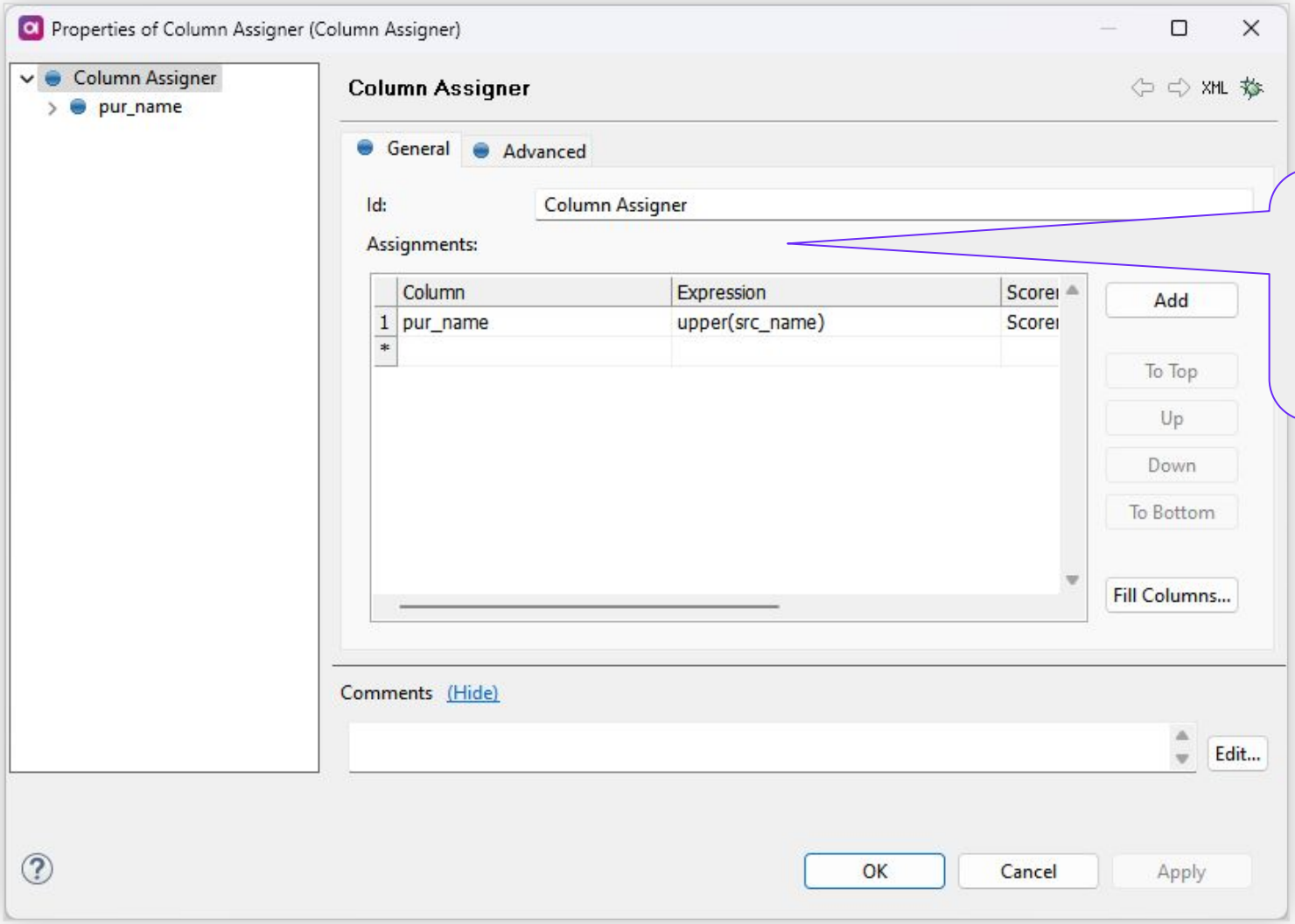
Basic Steps for Direct Data Transformation

- **Column assigner**
  - Allows data mapping (maps values from input columns to working columns, etc.).
  - Allows data value transformations with the use of IDE expressions.

- **Transliterate**
  - Translates any character or string into another defined character or string.

- **Regex matching**
  - Parses and extracts values from input strings via defined patterns and regular expressions.

Column Assigner Step

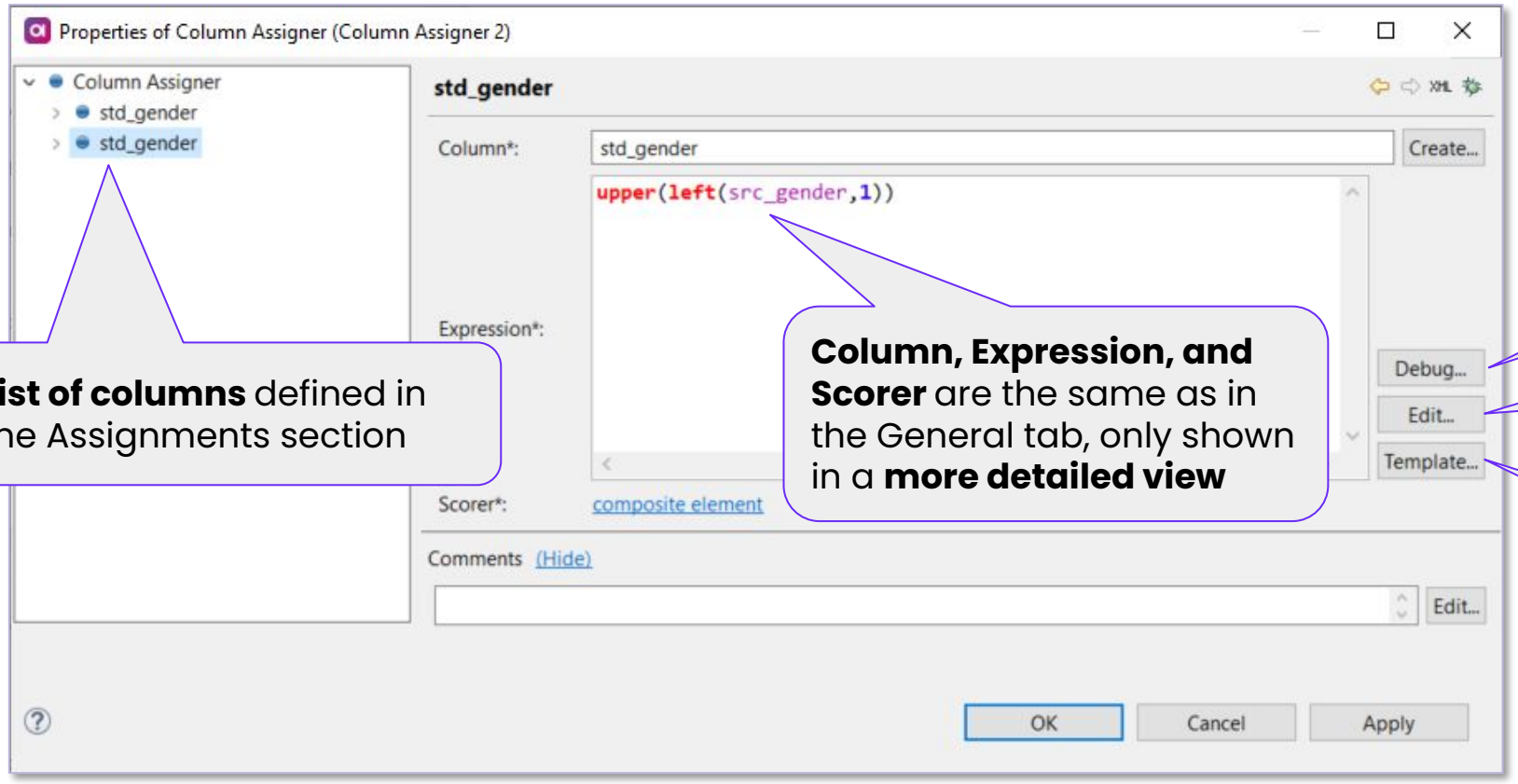


Assignments:

- **Mapping or transformation** operations
- **Scoring** of transformations
- **Comments** for each performed transformation

ataccama

Column Assigner Step



List of columns defined in the Assignments section

Column, Expression, and Scorer are the same as in the General tab, only shown in a **more detailed view**

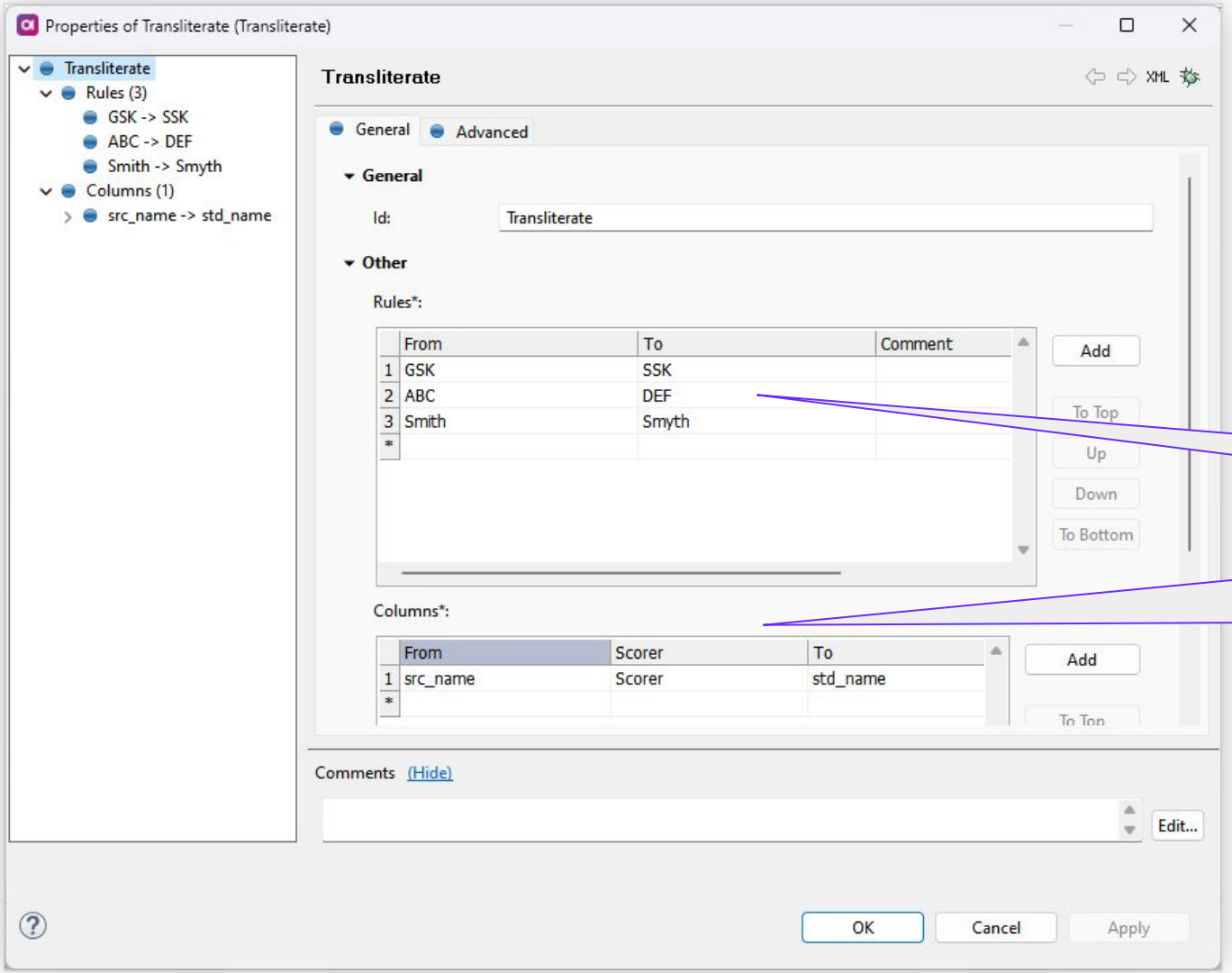
Debug allows you to debug your defined expression

Edit opens the Expression text in a separate window

Template opens a list of common expressions and operations and allows you to use them as a template

ataccama

Transliterate Step

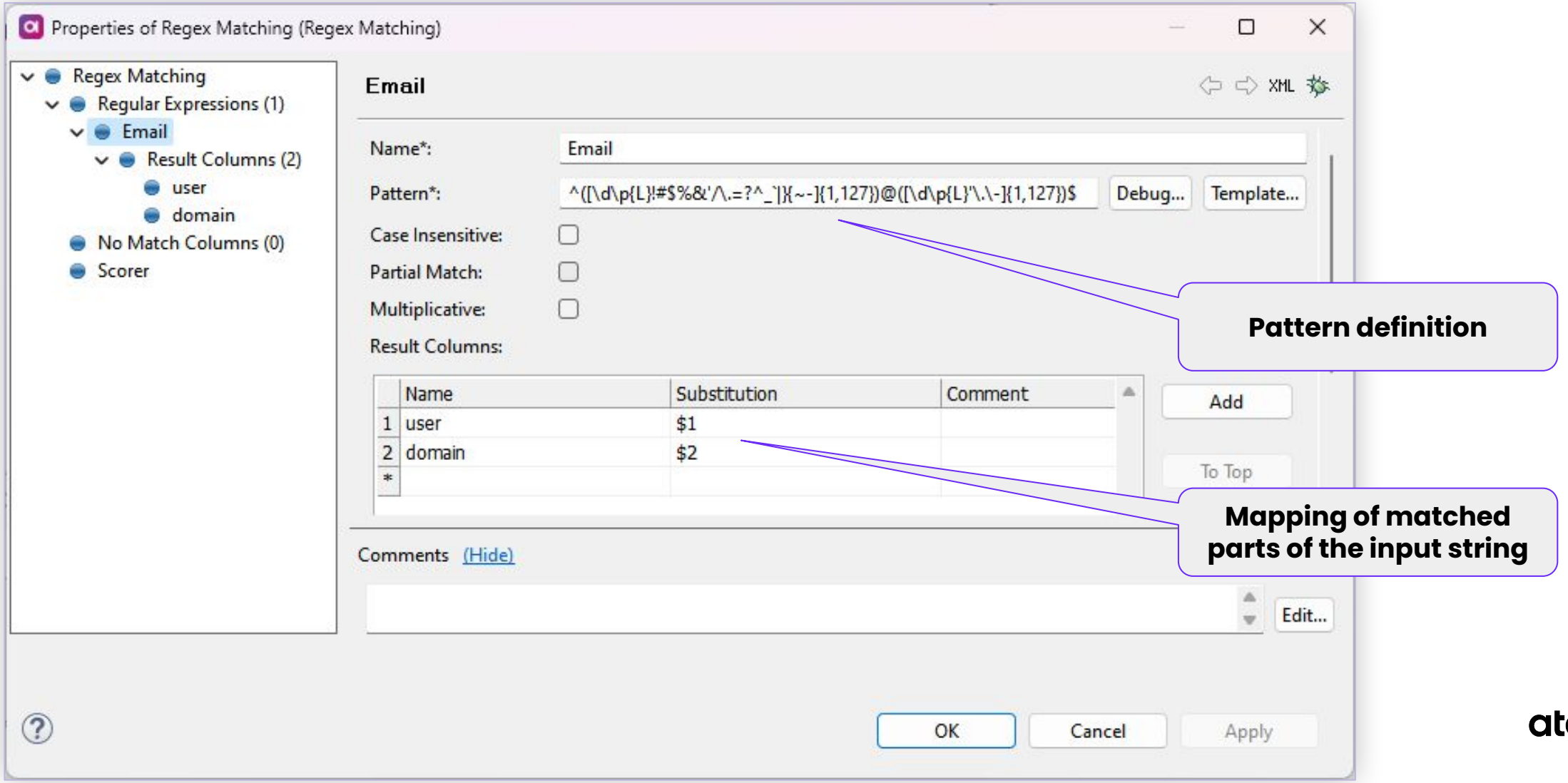


Defined **transliteration rules**

Defined **input column From** and **result column To**
(both may be the same column)

ataccama

Regex Matching Step



ataccama

Regex Matching Step

Regular expressions (Example usage):

- Parsing values in various formats (dates, phone numbers, identification numbers).
- Extracting values from fields containing multiple types of information (phone number mixed with address).
- Transforming input values.

The matched and unmatched parts of the input text can be accessed via special variables:

- **$&** or **$0** string matching the whole regular expression.
- **$1**, **$2**, … capturing groups (parts of matched string).
- **$`** part of input string before the match.
- **$'** part of input string after the match.

ataccama

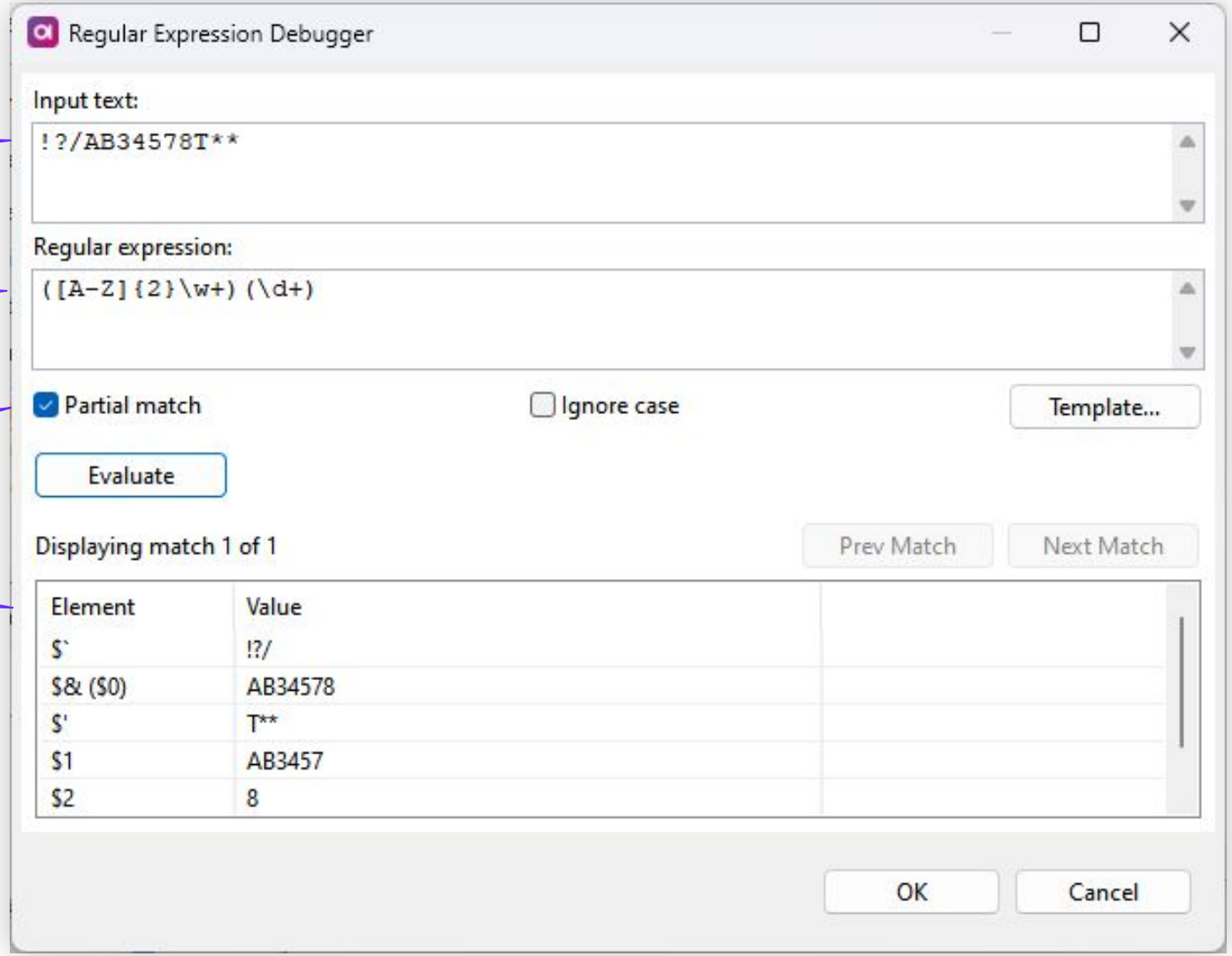
Regex Matching Debugger

Input string

Pattern for matching

Additional settings

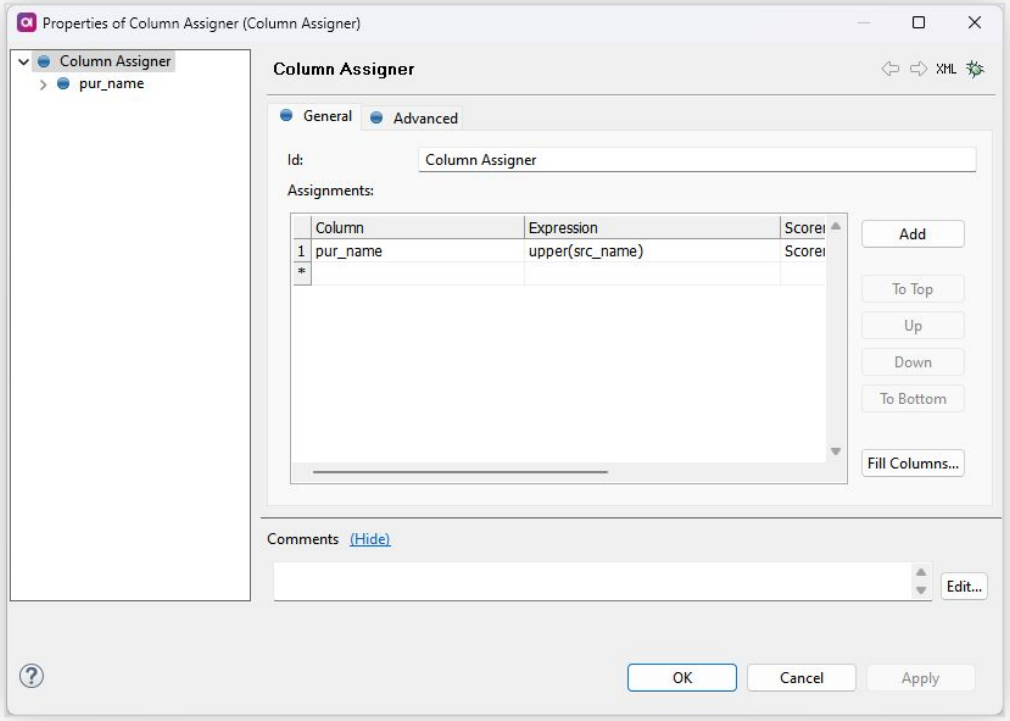
Matched parts of the input



ataccama

Topic Highlights

- **Column Assigner** inputs the result of a defined expression to a specified column.
- **Transliterate** transforms a specified string into another string.
- **Regex matching** parses data and extracts values in the input string using defined patterns of Regular expressions.



ataccama

Memory Refresher #6 Data Transformation

ataccama

Attribute names be dynamically changed in the Column Assigner.

- TRUE
- FALSE

ataccama

The Column Assigner can be used to add a new column.

- TRUE
- FALSE

ataccama

Lab Exercise #5
Data Transformation

ataccama

Step Debugging

ataccama

Debugging Overview

Debugging expressions:

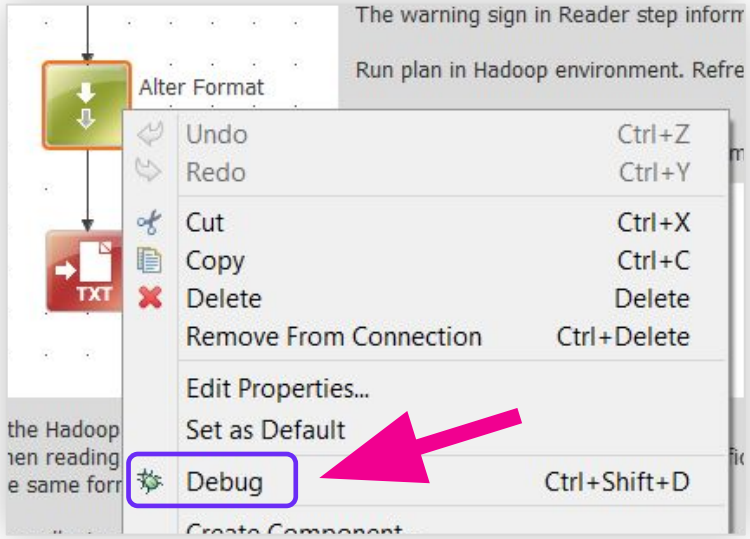
- Performed in a special form.

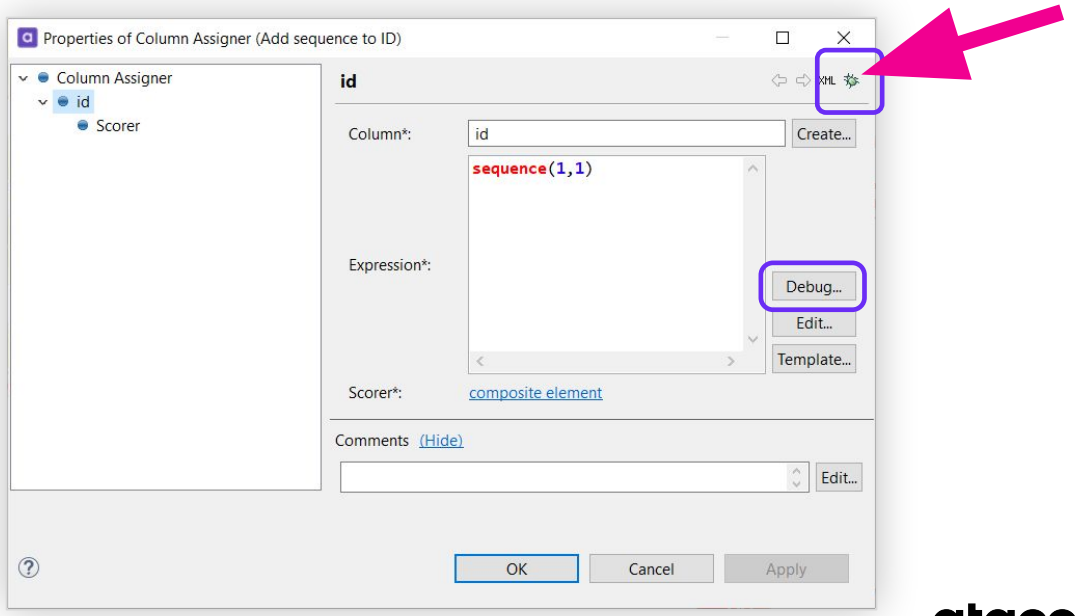
Debugging steps:

- Right-click on a step or click the bug icon in the upper right corner.
- Debugging a component step is also possible.

Testing some parts/whole part of the flow:

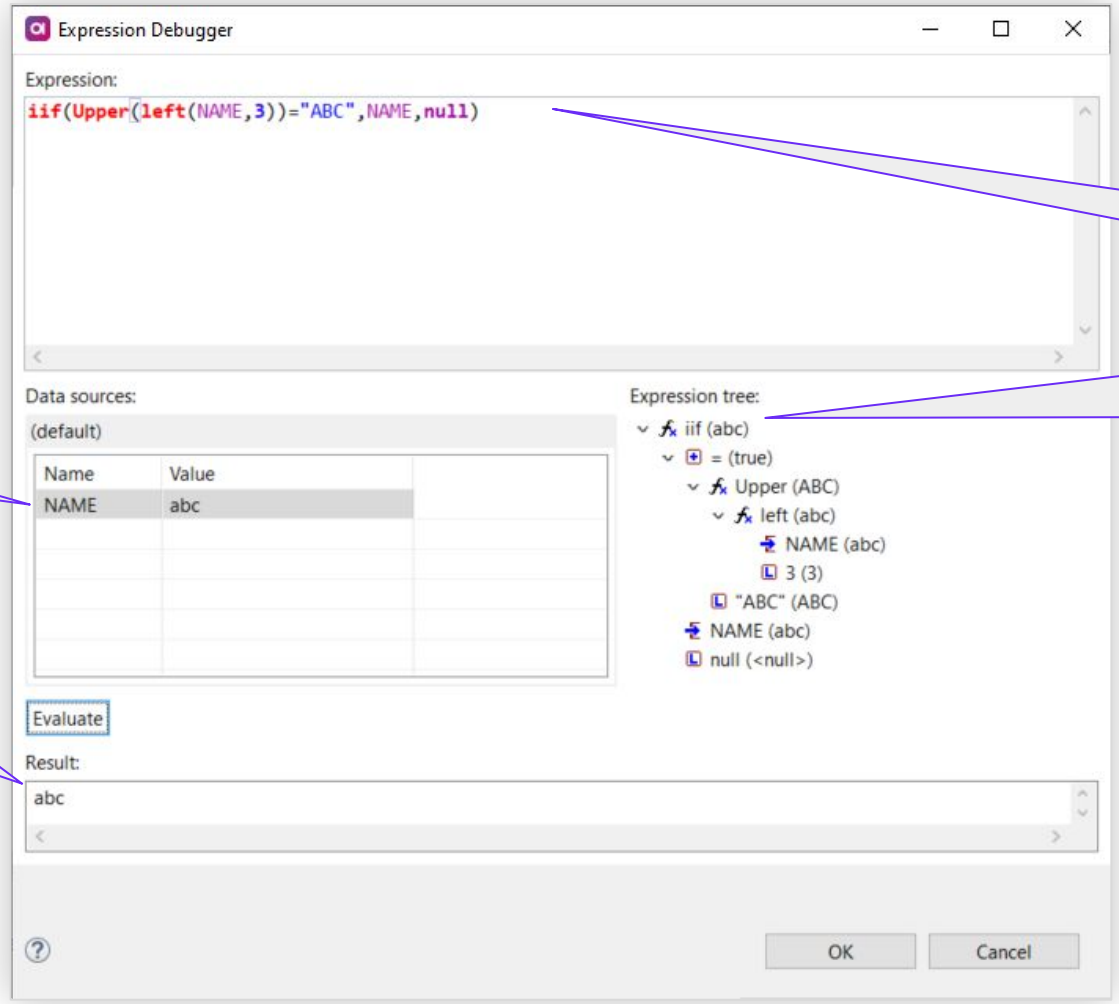
- Add Multiplicator + Text File Writer steps.





ataccama

Debugging Expressions



Expression definition comment format: /**/ or //

Input value used for debugging

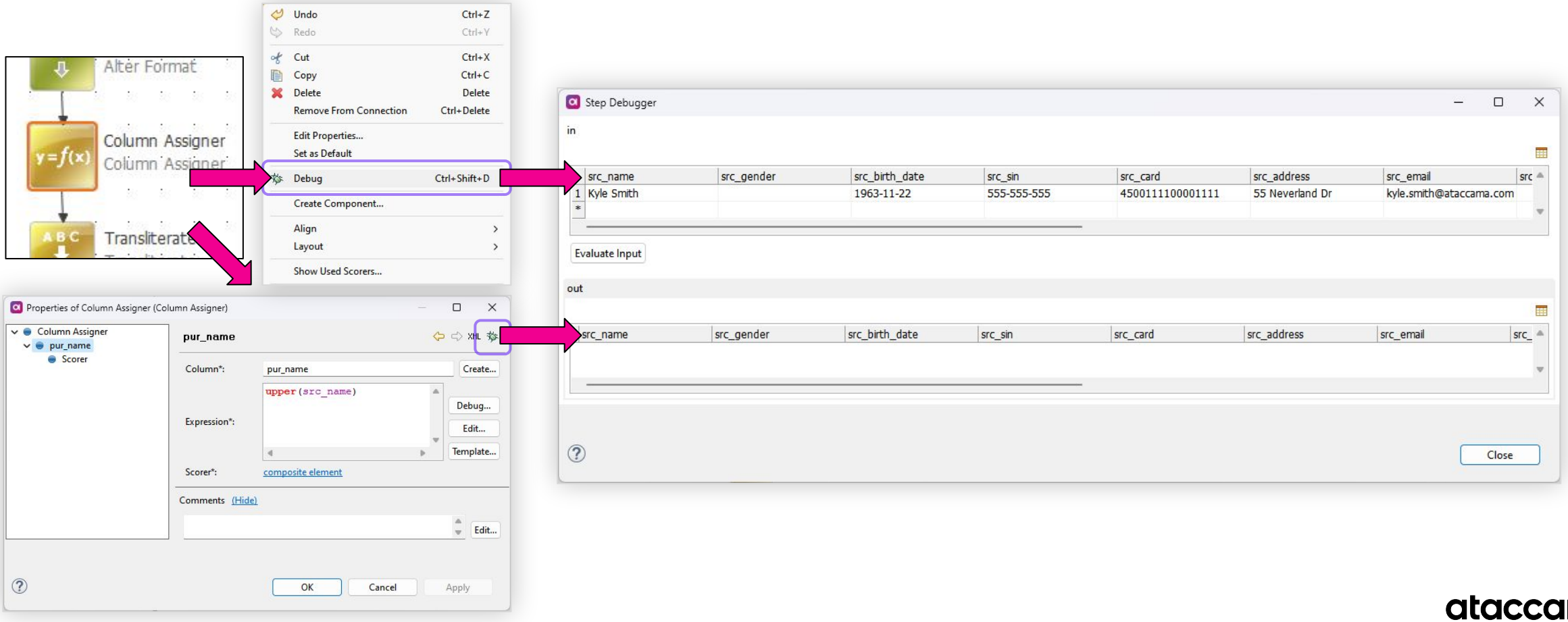
Expression tree shows individual operations in the expression step by step

Result shows the input value transformed by the defined expression

ataccama

Debugging Steps

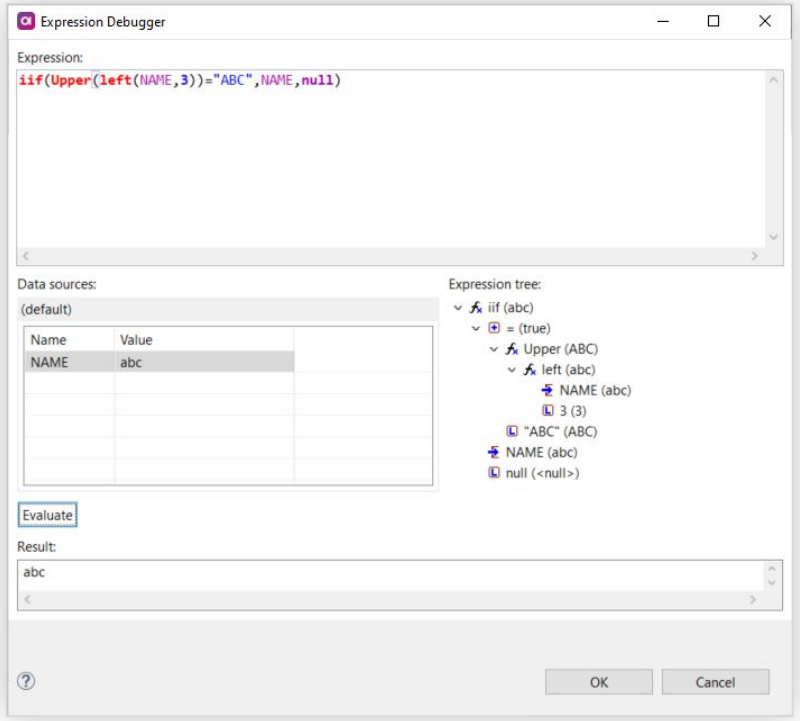
You can debug the entire step for a given sample input:



ataccama

Topic Highlights

- Debugging can be performed on expressions, steps, components, and parts of or entire plans.
- Functions or regular expressions can be debugged by clicking on **Debug** inside the step.
- Likewise, steps can be debugged by right clicking on the step then selecting **Debug** from the context menu.
- Plans can be debugged by inserting a **Multiplicator step** and **Text Writer** step at the desired point to view the plan's results to that point.



ataccama

Memory Refresher #7 Step Debugging

ataccama

Regular expressions can be debugged in the Regex Matching step.

- TRUE
- FALSE

ataccama

You can run a testing process for the whole plan or component.

- TRUE
- FALSE

ataccama

Is it possible to see intermediate results of an expression?

1. No.
2. Yes, during expression debug.
3. Yes, during debug for a step.

Lab Exercise #6
Debugging

ataccama

Scores & Explanations

ataccama

Scoring And Explanation Codes

- During scoring, every record is evaluated – scored – based on the quality of each value.
- The **higher the score, the bigger the difference between input and cleansed values**, and the lower the credibility of the input value.
- The **Cleansing Score** of any record is simply a sum of the cleansing scores of its individual attributes.
- Score is not self-explanatory by default and should be explained via **Explanation Codes**.

Best Practices For Scoring

Four scoring (quantification) levels:

Scoring level	Scoring description	Scoring result
0	The input value was **not modified.**	VALID
< 10,000	The input value **needed small modifications** (trim, squeeze spaces, uppercase, remove unsupported characters, safe replacements, etc.).	VALID
< 10,000,000	The input value **needed complex modifications** (not found in dictionary, does not meet official rules, major error, unsafe replacements, etc.).	UNSAFE / UNKNOWN
> 10,000,000	The input value (or pre-cleansed value after small modifications) **is null.**	NOT VALID

- The scoring value must be set in the specified range but does not have to exceed the maximum value in case the issue is repeated.
- It is recommended to use multiplications of 10 if you want to emphasize/differentiate the importance within one scoring level.

ataccama

Topic Highlights

- During scoring, every record is evaluated – scored – based on the quality of each value.
- The higher the score, the bigger the difference between input and cleansed values, and the lower the credibility of the input value.
- It is recommended to use multiplications of 10 if you want to. emphasize/differentiate the importance within one scoring level.
- Explanation should be without spaces.

Lab Exercise #7 Scores & Explanations

ataccama

Lookup & Dictionaries

ataccama

Building and using lookups

1 Build or obtain a Lookup File

Reference data (source)

- List of correct values (names, addresses)
- List of replacements (wrong □ correct)
- In a text file or database table

Lookup file

- A DQC binary (dictionary) file prepared for high-performance lookup operations
- Created by a plan with a **Lookup Builder** step

2 Use the Lookup file in a plan

Steps in your plan with the Lookup file:

- **Lookup**
- **Apply Replacements**
- **Strip Titles**
- **Pattern Parser**

ataccama

Lookups - Steps

- **Lookup Step**
  - Lookup structure: key + **additional columns** (optional)
  - Determines if the key exists in the lookup file
  - Enriches the input record with values from **additional columns** (if defined)

- **Apply Replacements**
  - Lookup structure: key + **replacement**
  - Replaces all occurrences of each key with the corresponding **replacement**

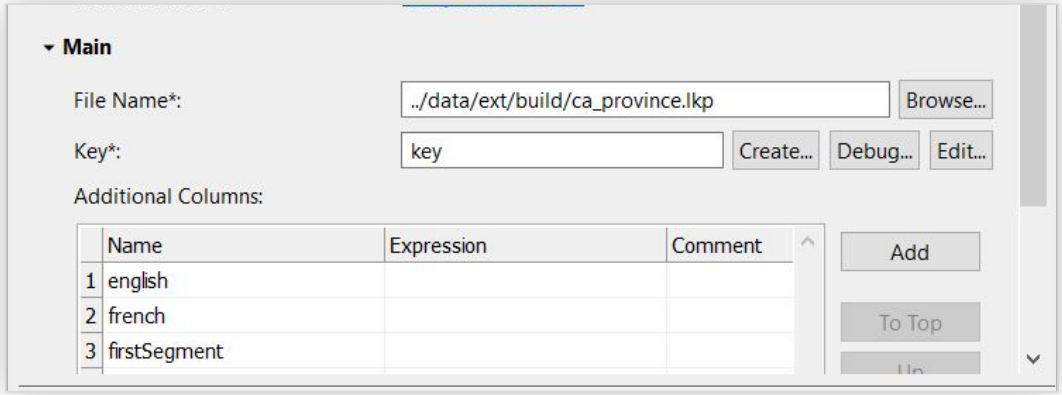
- **Strip Titles**
  - Lookup structure: key
  - **Removes** all occurrences of the key

Lookup Builder: File Structure

KEY

+

ADDITIONAL COLUMNS

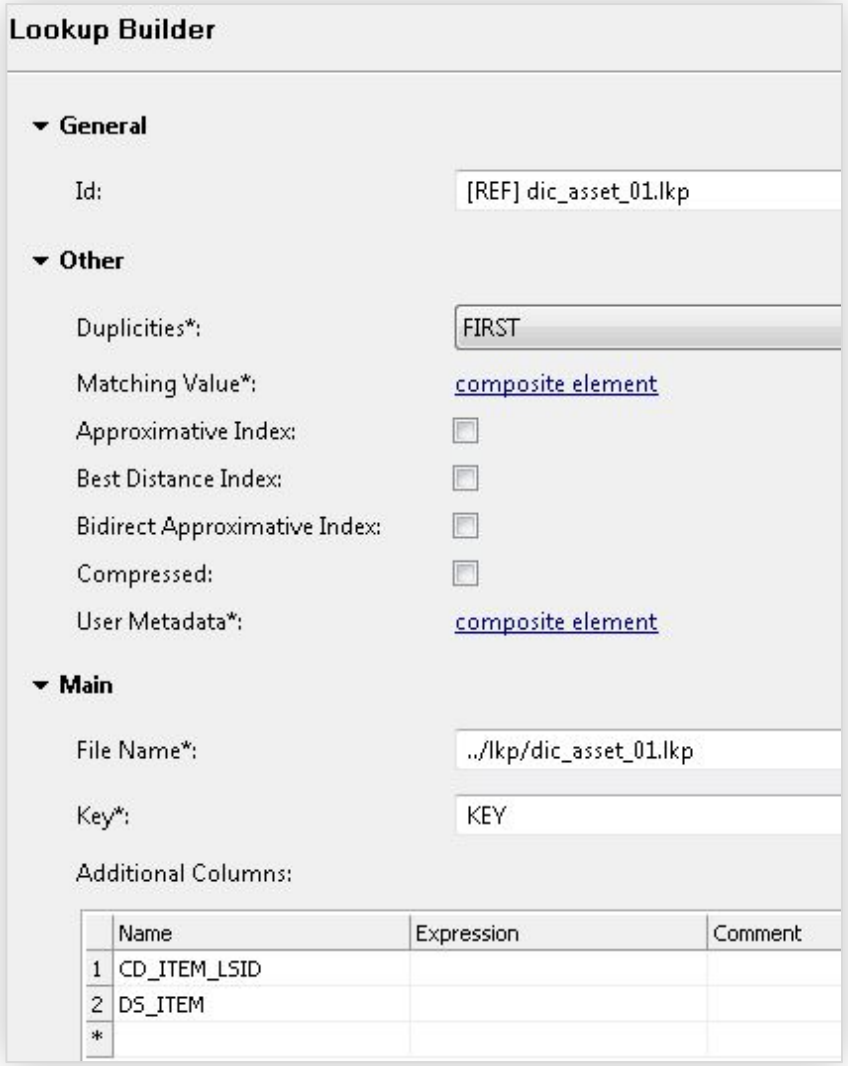


ca_ince.lkp

	key	english	french	firstSegment
1	AB	Alberta	Alberta	T
2	BC	British Columbia	Colombie-Britannique	V
3	MB	Manitoba	Manitoba	R
4	NB	New Brunswick	Nouveau-Brunswick	E
5	NL	Newfoundland and ...	Terre-Neuve-et-La...	A
6	NS	Nova Scotia	Nouvelle-Écosse	B

ataccama

Lookup Builder: Parameters



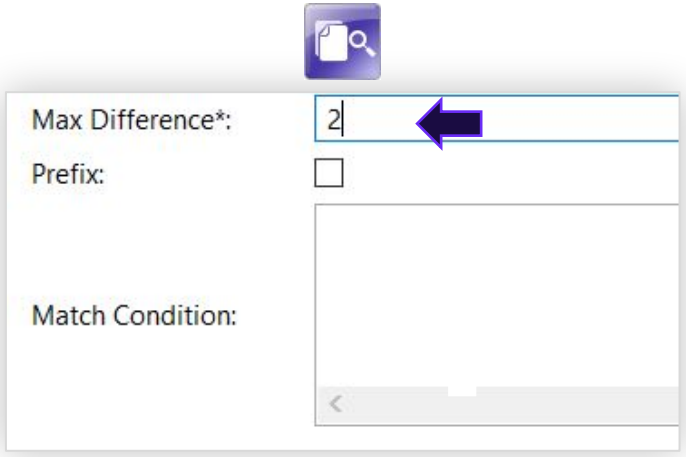
- **Duplicities** (first, accept, omit)
- **Matching Value** (Do upper case, squeeze whitespaces)
- **Approximative Index**
  - Necessary for approximative lookup (only for string keys)
- **Best Distance Index**
  - A more precise lookup stored in the index
- **Bidirect Approximative Index**
  - Improves the performance of approximative lookup
- **Compressed**
  - Compresses rows to improve performance; only useful with very wide datasets
- **User Metadata**
  - User-specific metadata will be included in the output
- Use the **lkp** folder for storing your lookup files

ataccama

Lookups:
Approximative Index

- Input strings are matched with lookup keys which have up to „**Max Difference**“ number of differences
- **What is considered a difference:**
  - One different character at any position within the string
  - One missing character at any position within the string
  - One additional character at any position within the string

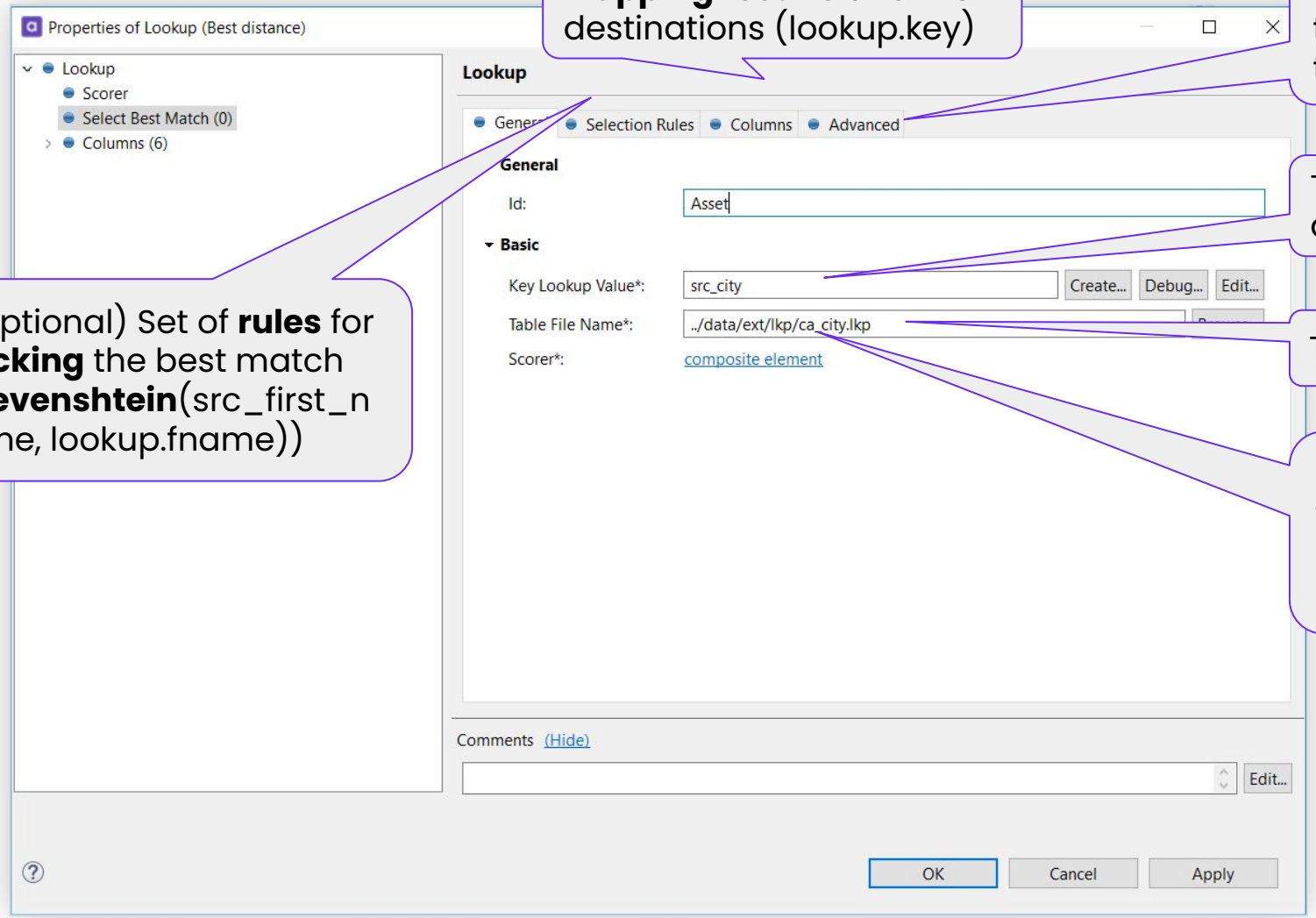
- Check the **Approximative** Index checkbox in the **Lookup Builder** step



- Set **Max Difference** in the **Lookup** step
- Maximum allowed number of differences between the strings
- No more than 2 is recommended

ataccama

Lookup Step: General



Mapping results and their destinations (lookup.key)

(Optional) **Expression** that must be satisfied for this step to be executed

(Optional) Set of **rules** for **picking** the best match (**Levenshtein**(src_first_name, lookup.fname))

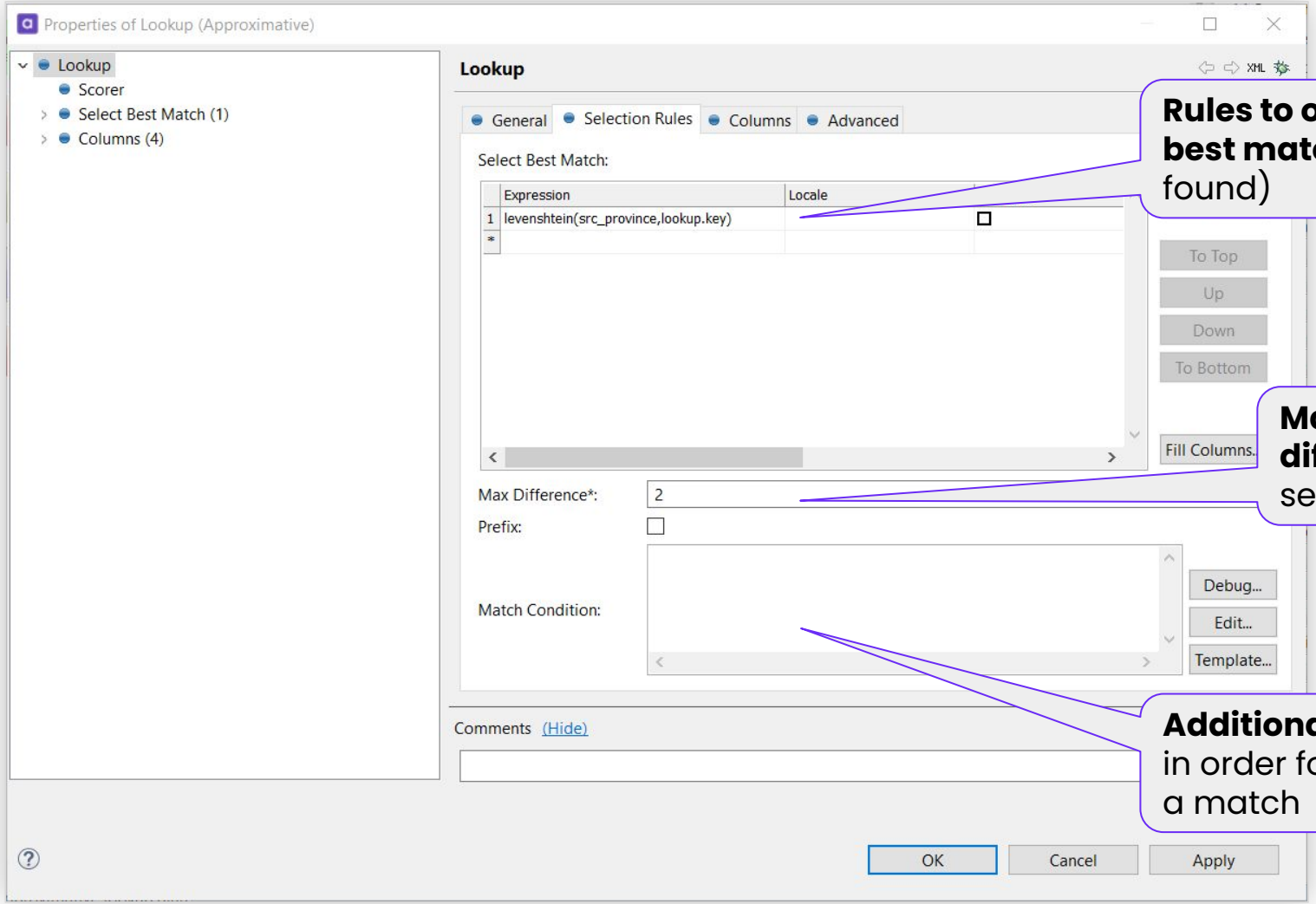
The column value used for querying the **lookup**

The **lookup file**

Scorer used for scoring the processing conditions (null key, no record found, etc.)

ataccama

Lookup Step: General



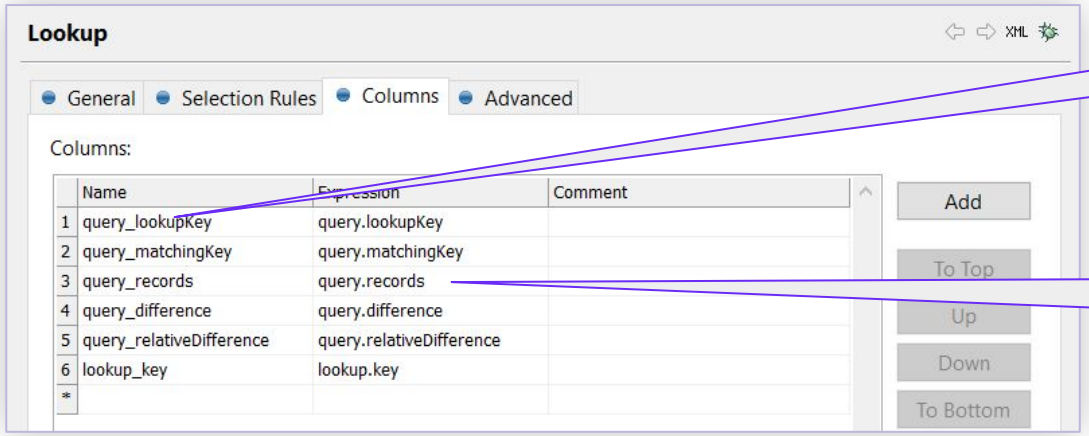
Rules to order the results and pick the best match (if multiple matches were found)

Maximum number of differences in approximative search string

Additional condition that has to be met in order for the record to be considered a match

ataccama

Lookup Step: Columns

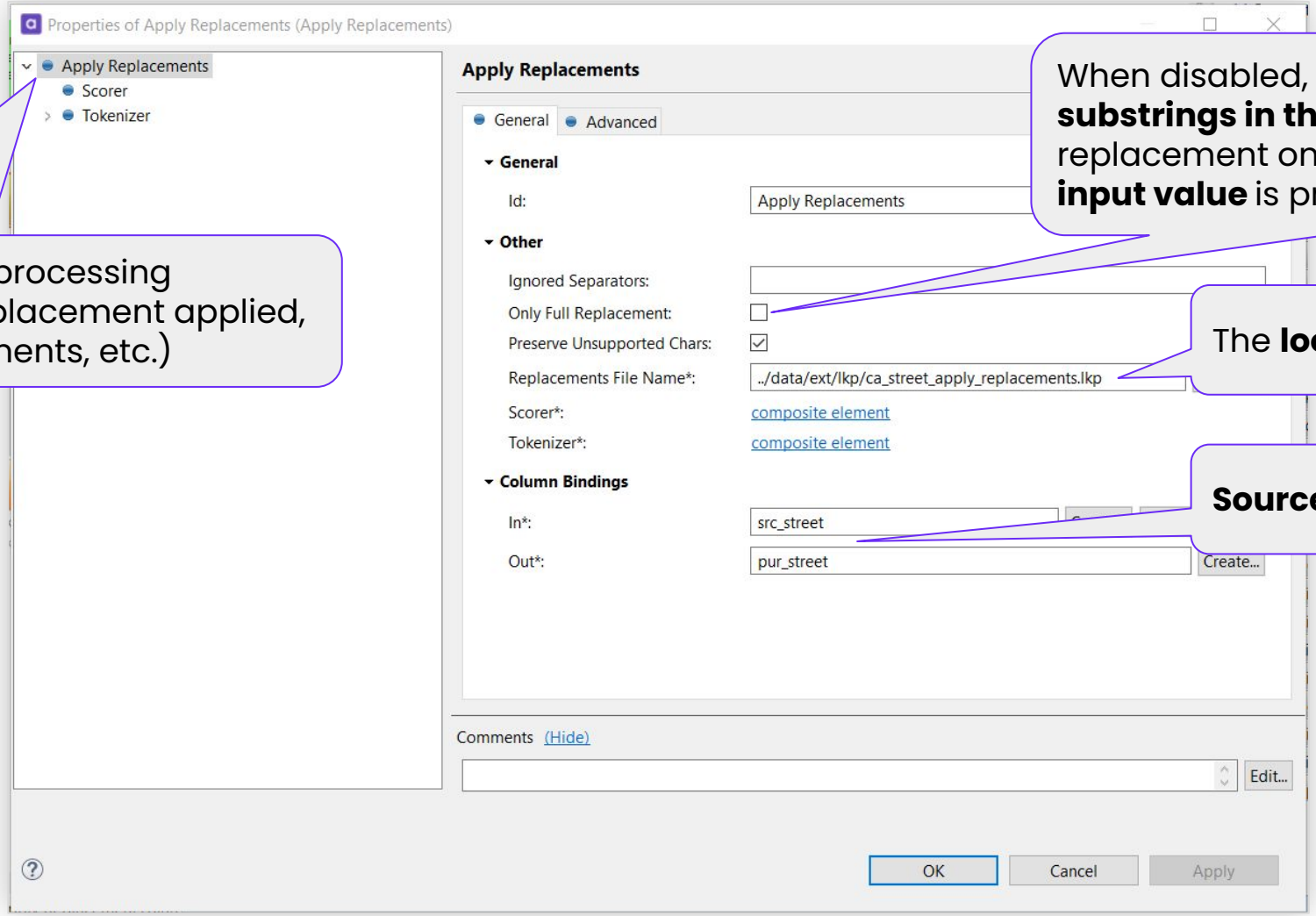


Besides the query key, there are many additional attributes which can be used or written to the output, such as:

• Lookup.**key**	The desired, initially specified key from the "Key Lookup Value" parameter.
• Query.**lookupKey**	The lookup key used in a Best Distance Index lookup.
• Query.**matchingKey**	The matching key used in a Best Distance Index lookup.
• Query.**records**	The number of records matched with a particular row.
• Query.**difference**	The number of differences between the match and the matching key (if they are allowed).
• Query.**relativeDifference**	If there were multiple matches, this shows how different each match was on average.

ataccama

Apply Replacements Step



Scorer for the processing conditions (replacement applied, more replacements, etc.)

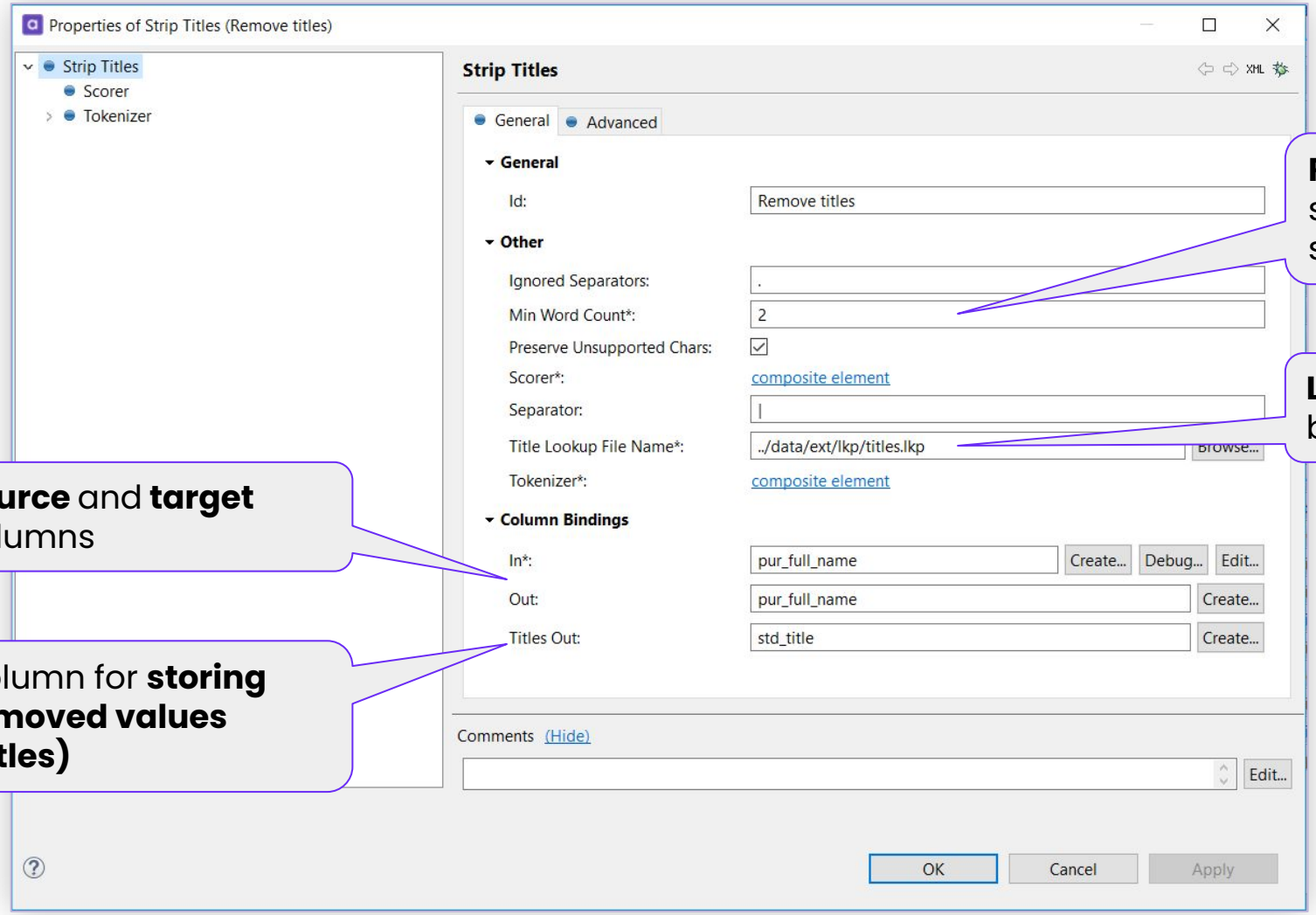
When disabled, the algorithm replaces **substrings in the input.** When enabled, replacement only occurs if the **entire input value** is present in the lookup file.

The **lookup file** with the data

Source and **target** columns

ataccama

Strip Titles Step



Removal will not occur if the resulting string would have fewer words than the specified number

Lookup file containing the list of titles to be removed

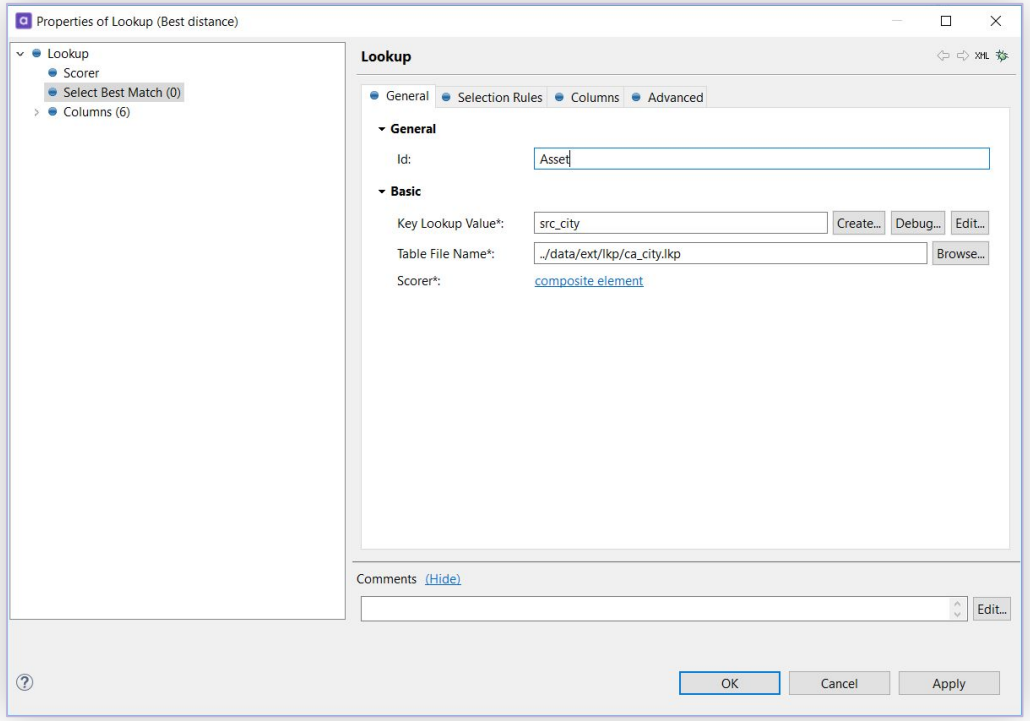
Source and **target** columns

Column for **storing removed values (titles)**

ataccama

Topic Highlights

- The **Lookup file** must be built in advance but can be reused as desired.
- **Lookup Builder** is used to create the Lookup file, a generic dictionary file.
- The **Lookup step** determines if the key exists in the lookup file.
- **Apply Replacements** will look for the input string in the dictionary, then perform replacements of all known instances of this string within the given text string.



ataccama

Memory Refresher #8 Lookup & Dictionaries

ataccama

What transformations occur with values when searching by lookup?

1. None.
2. You can specify the required transformations in the step's settings.
3. The same as described for the key during lookup building.

How many lines will lookup return if several values fit the key values?

1. One.
2. All.
3. None.

Lab Exercise #8
Lookup & Dictionaries

ataccama

ONE Desktop Core

v15.4.x