

# Talend Data Quality Framework – Dataset Onboarding Guide

## Introduction

This document describes the process for onboarding a dataset to the Talend Data Quality Framework (DQF). You can also use the procedures in this guide to modify an existing dataset that is already under management by the framework. Technical assets and other documents referenced by this guide are available for subscribers to download from the [Talend Data Quality Framework](#) page on Talend Academy. This guide assumes that the Talend Data Quality Framework has been installed and tested following the process documented in the *Talend Data Quality Framework Installation Guide*.

The target audience for this document is a Data Quality Developer or Data Engineer with data quality experience. Proficiency in using the data integration and data quality capabilities of Talend Studio is assumed. The processes outlined in this guide assume that this persona or user is onboarding a dataset to the framework in collaboration with a business user with the appropriate business knowledge and context to define the rules for the dataset.

## Prerequisites

### Recommended knowledge

- Snowflake or MySQL database administration
- Data integration development in Talend Studio
- Data quality development in Talend Studio
- Deploying Jobs from Talend Studio to Talend Management Console
- Talend Management Console Administration
- Talend Data Inventory (basic or advanced)
- Talend Pipeline Designer

## Recommended courses

- Talend Cloud Essentials
- Talend Data Quality Essentials
- Talend Cloud Data Preparation
- Talend Data Integration Basics
- Talend Data Integration Advanced
- Talend Cloud Administration
- Talend Cloud Pipeline Designer

## Software requirements

- Talend Studio:
  - Data Management or a superset thereof (API Services, Big Data, or Data Fabric)
  - Version **8** with monthly release **R2022-12** or higher applied if executing on a Remote Engine using Java 11
  - Version **8** with monthly release **R2023-04** or higher applied if executing on a Cloud Engine or Remote Engine using Java 8
- Talend Cloud subscription
- SQL client for Snowflake or MySQL
- [Power BI Desktop](#)

## Subscription requirements

A current subscription to one or both of the following Talend Accelerator Service Subscriptions:

- Enable Analytics
- Establish Data Excellence

# Contents

Key terminology.....	5
Onboarding request.....	5
Ingestion options .....	6
Data Inventory and Pipeline Designer .....	6
Talend Studio Data Integration Job .....	10
Table in the DQF database.....	10
Semantic types .....	11
Baseline analysis.....	12
Analysis types.....	12
Indicators to types .....	13
Indicator result types and DQF rule suitability .....	18
Analysis and indicator combinations suitable to become DQF rules.....	20
Business data quality rules.....	21
Reviewing analysis results with a business user.....	21
Semantic or basic type based data quality rules.....	21
Business data quality rules.....	22
Data quality dimension mapping.....	22
Scheduling and notifications.....	22
DQF metadata .....	23
Dataset metadata .....	23
Notification metadata .....	24
Implementing, testing, and deploying.....	25
Configuring the report and reporting Job.....	25

Avoiding conflicts with existing DQF Runs .....	29
Executing a test run and troubleshooting.....	29
Cleaning up the TDQ mart .....	30
Dataset and DQF run relationship .....	30
Deploying and scheduling .....	31

## Key terminology

- **Dataset** – In Talend Data Quality Framework, a dataset is a table, view, query, object, or file that, when loaded to the DQF Raw Schema for analysis, results in a single, flat table.
- **Reference data or lookup table** – This is data also held in the DQF Raw Schema for validation checks, lookups, and consistency checks for one or more datasets.
- **Indicator** – Each analysis in Talend Studio leverages one or more indicators to analyze a dataset. Talend Data Quality Framework leverages the Studio profiler for analysis and incorporates indicators as a component of a DQF “rule”. Talend provides a built-in set of indicators, and users can easily create new ones. Future versions of the framework will incorporate indicators that do not originate from the Studio profiler.
- **DQF rule** – The combination of an indicator, threshold, data quality dimension, and supporting metadata make up a DQF rule - more on this later in this guide.

## Onboarding request

Talend advises beginning the process with a business user completing a Dataset Onboarding Request form to capture the following:

- How the dataset is used and its importance to the business
- Dataset owner
- Data Steward
- Dataset attributes – The demo dashboard allows you to associate up to three fully customizable attributes with the dataset rules to allow slicing and dicing in the BI layer. For example, in the provided DQF demo, the attributes are:
  - **Department** – *Sales*, for example
  - **Data Domain** – *Customer*, for example
  - **Initiative** – *Analytics*, for example
- Dataset connectivity details
- The unique (primary) key field that identifies a specific row
- Critical data elements – Are there particular columns that should be the focus of the analysis and ongoing data quality monitoring? It is common for tables, views, and other data objects to have a large number of fields, but not all are used by or critical to a given business process. Capture any assumptions about these critical data elements (for example, **this field should contain a valid email address and is mandatory**).
- Known data quality rules to be investigated, validated, or implemented
- Scheduling information (for example, **this table is refreshed at 7 a.m. every day, and the data quality monitoring should be executed at 8 a.m. every day**)
- Notification information, if known at this stage (for example, **if data quality rule X fails, an email is sent to email distribution list Y**)

## Ingestion options

As described in the *DQF Prerequisite and Architecture Guide*, the Talend Data Quality Framework works in ELT (Extract, Load, Transform) mode, also known as SQL pushdown mode. In practical terms, this means that the data to be analyzed and the DQF data mart must be in the same database. For that reason, there are three options available to ensure the data is in the right place at the right time for analysis and processing by the framework.

### Data Inventory and Pipeline Designer

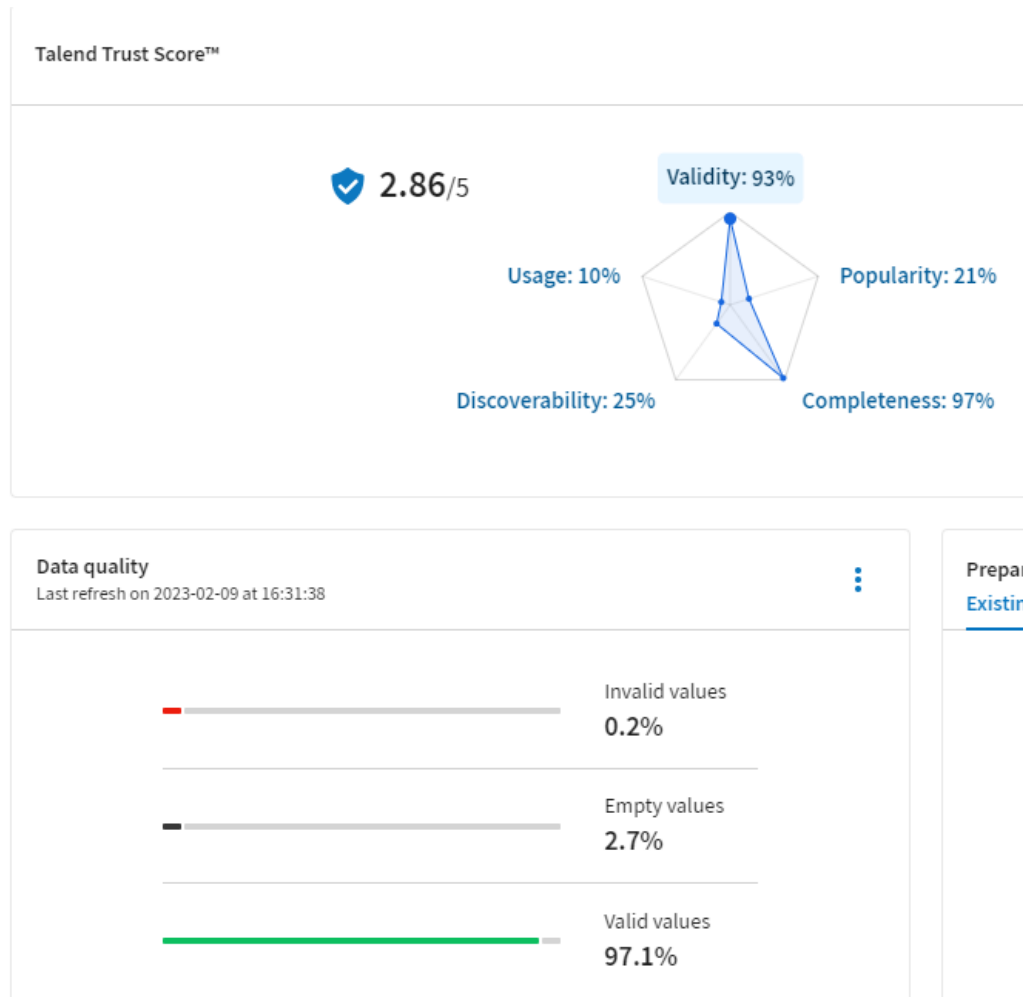
Talend Cloud Data Inventory provides a “single pane of glass” for your connections and available datasets accessed through those connections. Talend Data Inventory is available in basic and advanced versions, depending on your Talend subscription. For more information, see [About Talend Cloud Data Inventory on Data Inventory](#) in Talend Help Center.

For the Talend Data Quality Framework, Data Inventory as a standalone application (advanced) or as part of Data Preparation and Pipeline Designer (basic) can serve a number of purposes:

- It allows one user to create and share a connection and datasets with another user without the need to share sensitive credentials. For example, a system administrator could create a connection or dataset, and share it with a Talend Service Consultant who is delivering the Value Added Service: [Onboard a new Dataset to the Talend Data Quality Framework](#), so that they can set up an integration to the DQF Raw Schema.
- Semantic types – leveraged extensively by Data Inventory and covered in detail in the [Semantic types](#) section of this guide.
- Trust Score – includes two data quality dimensions (Validity and Completeness) as well as integration with the cloud data quality rules repository. This helps with an initial assessment of the dataset quality.
- Sources and targets for Pipeline Designer

For a full list of systems that can be connected to by Data Inventory (and therefore Pipeline Designer), see [Introducing Talend Connectors](#) in Talend Help Center.

This screen capture shows an example dataset in Data Inventory (Advanced edition) displaying useful information about the quality of the dataset.



This image shows the same dataset in sample view Data Inventory (Advanced edition), showing semantic types and quality issues.

Head sample ▾  
2023-02-09 at 16:31:38 by apemle@dqaas.com

Sample quality  
0.2% ▬ 2.7% ▬ 97.1% ▬

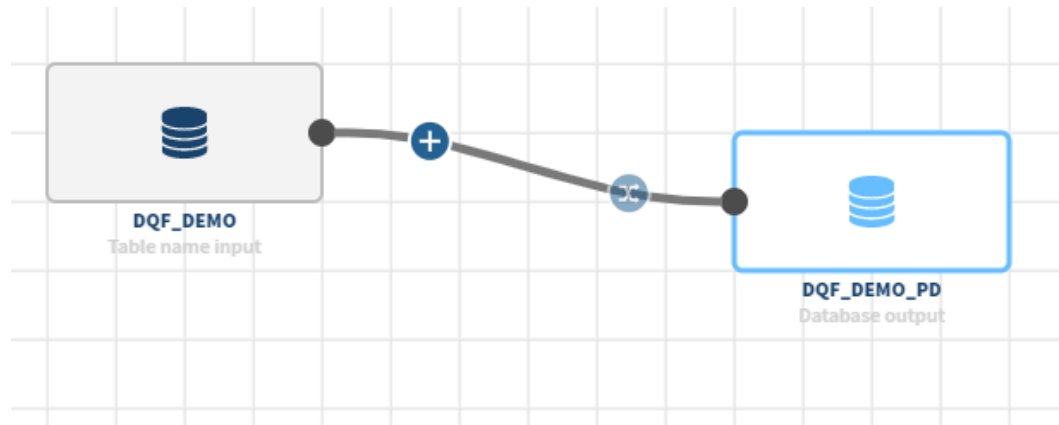
	COMPANYNUMBER* UK Company House (stri...	COMPANYNAME Text (string)	REGADDRESS_POST... UK Postal Code (string)	COMPANYCATEGORY CH_CompanyCategory (...)	COUNTRYOFORIGIN ISO Short Country Nam...	REGADDRESS_COU... ISO Short Country Nam...	ACCOUNTS_LASTMA... Date (long)	ACCOUNTS_NEXTD... Date (long)	ACCOUNTS_ACCOU... Text (string)	SICCODE_Si UK_SIC_Code
1	08073448	BAKER CONSULTING S...	RG27 9BS	Private Limited Co...	United Kingdom	United Kingdom	2022-01-01T00:00:0...	2022-03-30T00:00:0...	MICRO ENTITY	41100 -
2	09268250	TBI VAUXHALL LIMIT...	IG1 4TG	Private Limited Co...	United Kingdom	United Kingdom	2020-10-31T00:00:0...	2022-07-30T00:00:0...	TOTAL EXEMPTION FU...	96090 -
3	11550193	VIPA BUSINESS SERV...	NE65 9HP	Private Limited Co...	United Kingdom	United Kingdom	2020-09-29T00:00:0...	2022-06-29T00:00:0...	MICRO ENTITY	82110 -
4	06760452	GRANDVILLE LODGE L...	SS9 1BG	Private Limited Co...	United Kingdom	United Kingdom	2019-12-31T00:00:0...	2021-09-29T00:00:0...	TOTAL EXEMPTION FU...	87100 -
5	04446484	AARON PRINTERS LIM...	TQ12 2QA	Private Limited Co...	United Kingdom	United Kingdom	2020-05-30T00:00:0...	2022-02-28T00:00:0...	TOTAL EXEMPTION FU...	18129 -
6	04800195	AMBASSADOR DEVELOP...	IG6 3XJ	Private Limited Co...	United Kingdom	United Kingdom	2020-06-29T00:00:0...	2022-03-30T00:00:0...	TOTAL EXEMPTION FU...	68100 -
7	00047684	HOTCHKISS PATENTS ...	FY4 1EZ	Private Limited Co...	United Kingdom	United Kingdom	2020-03-30T00:00:0...	2021-12-31T00:00:0...	DORMANT	93290 -
8	04661314	ORCHID ENVIRONMENT...	PR7 4EZ	Private Limited Co...	United Kingdom	United Kingdom	2019-12-31T00:00:0...	2021-09-29T00:00:0...	SMALL	38210 -
9	08200504	CHINESE NEW GARDEN...	WN4 9AG	Private Limited Co...	United Kingdom	United Kingdom	2018-09-29T00:00:0...	2020-09-29T00:00:0...	TOTAL EXEMPTION FU...	56101 -
10	00660684	DTMGL F AFROSPACE I	SO51 7HX	Private Limited Co...	United Kingdom	United Kingdom	2020-06-29T00:00:0...	2022-03-30T00:00:0...	MICRO ENTITY	33160 -

COMPANYNUMBER  
Quality Description

Type  
UK Company House

Rules  
No rules applied to this field  
Apply rules to detect potential anomalies

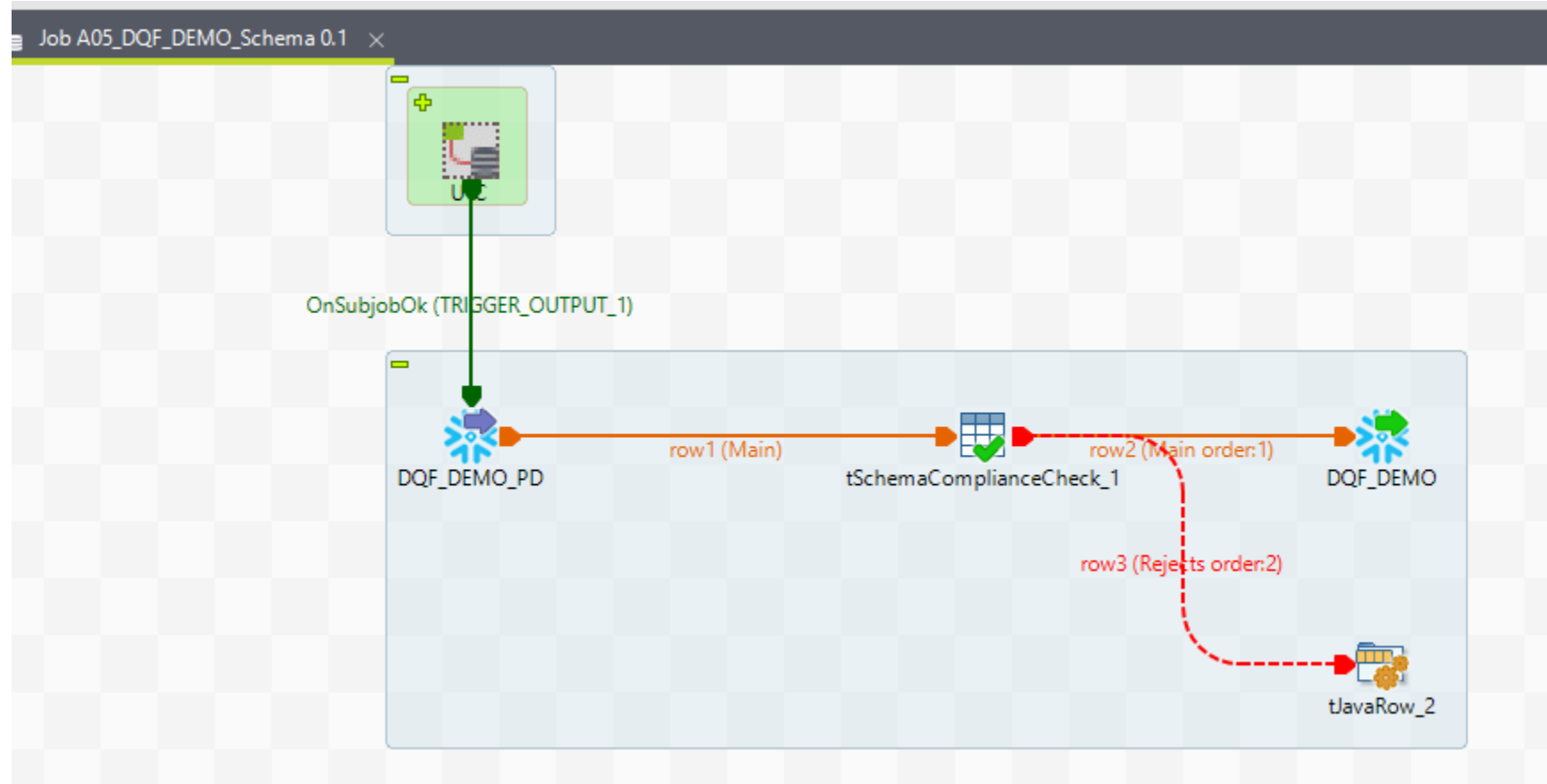
As described earlier, Pipeline Designer provides a simple way to implement and schedule a simple integration between the original source system for your dataset and the DQF Raw Schema (where data is analyzed). A typical pipeline for this purpose may look like this:



In this case, the pipeline is a simple “lift and shift” operation from the source system to the DQF RAW Schema in the DQF database.



**Note:** If the source is a flat file, you should land everything into an intermediate table with a loose schema structure, for example, with all fields set to VARCHAR(255) or even TEXT types. A similar principle applies if you want the source system schema to be different to the analysis schema for the purpose of analysis. For example, if the source system stores dates in a string field, but you want to cast this data to a DATE type in your analysis schema to allow date-specific analysis operations. This intermediate table approach allows for a second step in Studio, where you can test the actual schema against the data using a **tSchemaCompliance** check component. An example of this is included in the DQF demo Jobs:



## Talend Studio Data Integration Job

Talend Studio is a powerful data integration tool with a huge breadth of connectivity to many different systems. You can leverage it to build integrations between source systems and the DQF Raw Schema. For simple cases, create and schedule one Job per dataset as part of the overall DQF process flow. For integration at scale, consider using the [Metadata Driven Framework](#), which is available by subscription to Talend Academy or by Talend Professional Services engagement.

## Table in the DQF database

In this scenario, the table you want to analyze is already in the same database as the DQF Mart, so moving the data to the DQF Raw Schema may not be required. For this approach, consider the following:

- The user that is used to access the DQF mart and DQF Raw Schema must also be granted access to any other schemas and objects that contain the data you want to analyze.
- For lookup type operations, the profiler requires that the objects being compared are in the same schema. This may mean that you need to create reference data tables in the schema where your source data resides.
- The source data should not be altered while a DQF run is being executed to avoid receiving inconsistent results.

## Semantic types

In Talend Cloud, semantic types are used in many of the cloud applications, including Data Inventory, Data Preparation, Pipeline Designer, and Data Stewardship. They are a key part of the data quality capabilities provided by the cloud platform. Semantic types are stored in and administered by the [Talend Dictionary Service](#).

Talend Data Quality Framework leverages the same concepts as Talend Cloud, namely that the two types of semantic type, Dictionary and Regex (plus the Compound type that allows semantic types to be combined), are useful in both the analysis phase and also as “rules” for ongoing monitoring.

In Talend Dictionary Service, the semantic types are divided into three main categories:

- The Dictionary type, based on an open or closed list of values
- The Regular expression type that compares your data against a preselected regular expression
- The Compound type, under which you can group several existing types

As part of the onboarding process, it is useful to know the semantic type of each field. Leveraging Data Inventory provides an automated attempt to align the semantic types (out of the box or custom) stored in the Dictionary Service to the data and metadata of the dataset. The most value comes when these semantic types (if found) are reviewed by the Business Data Owner, who may then perform the following actions:

- Create semantic types in the Dictionary Service that are relevant to the dataset being onboarded to Talend DQF, but have not yet been created
- Update the definitions of semantic types where they are not correct or incomplete
- Change the base type or semantic type assigned to a given field from the default assignment to a different semantic type or base type

When the business user is finished reviewing the semantic types assigned to each field in a given dataset, the Data Quality Developer can use these assignments as part of the DQF baseline analysis. The following example illustrates how this is helpful. A given field in a dataset contains 60% valid email addresses, 20% NULLS, and 20% other strings that are not email addresses. When performing a baseline data quality analysis, if you only know that the type is a String, you can apply a variety of data quality indicators to that field that are relevant to all strings. Looking at the data and analysis results, a Data Quality Developer may suggest a rule that the field should contain valid email addresses, but this will require a review iteration and feedback from the Business Data Owner to confirm. It also relies on the individual Data Quality Developer to make the appropriate recommendation, which is not always a given. If, however, during the initial baseline analysis, the Data Quality Developer knows more than what can be derived from the metadata in the database schema (that is, it is a string field), such as the field should be of type email and mandatory, then appropriate indicators and rules can be created immediately.

**Note:** Future releases of the Talend Data Quality Framework will allow Data Quality Developers to automatically synchronize the semantic types from Talend Cloud to the framework using a number of different mechanisms. In this release, synchronization requires implementation or manual processes during onboarding.

# Baseline analysis

## Analysis types

Talend Data Quality Framework can leverage all types of Studio Profiler analyses to help the Data Quality Developer and business user understand a dataset and test their assumptions about the data. This is at the discretion of the Data Quality Developer. See [Data Profiling and Data Quality](#) in the Talend Help Center for more details on analysis types. However, as previously mentioned, Talend recommends that you first perform an initial baseline analysis when onboarding a dataset to the framework. This process uses a subset of the available analysis types:

- **Basic Column Analysis:** Use this analysis on every field with a scope of the whole dataset (as defined by the business user). Recommended indicators are described in the [Indicators to types](#) section.
- **Basic Column Analysis with filter:** Use this analysis if the business user has specified on the request form a basic type or semantic type-based rule that only applies in certain conditions. For example, **The email address must be present and valid if the order source field = WEB.**

**Note:** The filter in an analysis defines the analysis **scope**. That is, it defines what data is analyzed; it is not a condition used to check the validity of data.

- **Redundancy Analysis:** Use this analysis to do cross-table (or view) analysis within the same schema. This is useful for checking that a field (or set of fields) is valid when compared to reference data (that is, lookup tables). This reference data may be sourced from the business user or from a Talend Cloud semantic (dictionary) type. You can also use this analysis for consistency checks, such as primary–foreign key consistency.

**Note:** For the purposes of Talend Data Quality Framework, Talend recommends, where possible, that all analyses are configured to run in SQL pushdown mode (which is the default) as opposed to Java mode.

## Indicators to types

For the baseline analysis, Talend recommends using the indicators by type (base or semantic) described in the following sections.

### Text types

These include cases where String is the base type of a semantic type.

Indicator category	Indicator	Notes
Simple Statistics	Row Count	This is the count of rows after an analysis filter is applied. It is not required on every field; one field per basic column analysis is sufficient.
Simple Statistics	Null Count	
Simple Statistics	Blank Count	Oracle treats null and empty strings as the same thing, but strings containing only whitespace characters are counted as “blanks”.
Simple Statistics	Distinct Count	Case sensitivity can be set in the indicator, but it is also determined by the database collation.
Simple Statistics	Unique Count	Case sensitivity can be set in the indicator, but it is also determined by the database collation.
Simple Statistics	Duplicate Count	Case sensitivity can be set in the indicator, but it is also determined by the database collation.
Advanced Statistics	Value Frequency	Use this when the field is meant to contain a limited set of values, such as reference data and dictionary semantic types. You can configure the number of values displayed, but it becomes unusable if set too high. Use it in combination with Value Low Frequency to look for outliers and anomalies that are potential data quality issues for which a DQF rule should be created.
Advanced Statistics	Value Low Frequency	
Pattern Frequency Statistics	Pattern Frequency	Use this to see if values in the field follow a limited set of patterns and that there is potential for a regular expression DQF rule to be created. You can configure the number of values displayed, but it becomes unusable if set too high. Use it in combination with Pattern Low Frequency to look for outliers and anomalies that are potential data quality issues for which a DQF rule should be created.

Indicator category	Indicator	Notes
Pattern Frequency Statistics	Pattern Low Frequency	
Text Statistics	Minimal Length	This is helpful in looking for anomalies.
Text Statistics	Maximal Length	This is helpful in looking for anomalies.

## Number types

These include cases where a number is the base type of a semantic type.

Indicator category	Indicator	Notes
Simple Statistics	Row Count	This is the count of rows after an analysis filter is applied. It is not required on every field; one field per basic column analysis is sufficient.
Simple Statistics	Null Count	
Simple Statistics	Distinct Count	Case sensitivity can be set in the indicator, but it is also determined by the database collation.
Simple Statistics	Unique Count	Case sensitivity can be set in the indicator, but it is also determined by the database collation.
Simple Statistics	Duplicate Count	Case sensitivity can be set in the indicator, but it is also determined by the database collation.
Advanced Statistics	Value Frequency	This is useful if the field is meant to contain a limited set of values, such as reference data and dictionary semantic types. You can configure the number of values displayed, but it becomes unusable if set too high. Use it in combination with Value Low Frequency to look for outliers and anomalies that are potential data quality issues for which a DQF rule should be created.
Advanced Statistics	Value Low Frequency	
Pattern Frequency Statistics	Pattern Frequency	This is useful to see if values in field follow a limited set of patterns and that there is potential for a regular expression DQF rule to be created. You can configure the number of values displayed, but it becomes unusable if it is set too high. Use it in combination with Pattern Low

Indicator category	Indicator	Notes
		Frequency to look for outliers and anomalies that are potential data quality issues for which a DQF rule should be created.
Pattern Frequency Statistics	Pattern Low Frequency	
Summary Statistics	Minimum	This is helpful in looking for anomalies.
Summary Statistics	Maximum	This is helpful in looking for anomalies.

## Date types

Indicator category	Indicator	Notes
Simple Statistics	Row Count	This is the count of rows after an analysis filter is applied. It is not required on every field; one field per basic column analysis is sufficient.
Simple Statistics	Null Count	
Simple Statistics	Distinct Count	Case sensitivity can be set in the indicator, but it is also determined by the database collation.
Simple Statistics	Unique Count	Case sensitivity can be set in the indicator, but it is also determined by the database collation.
Simple Statistics	Duplicate Count	Case sensitivity can be set in the indicator, but it is also determined by the database collation.
Advanced Statistics	Value Frequency	This is useful if the field is meant to contain a limited set of values, such as reference data and dictionary semantic types. You can configure the number of values displayed, but it becomes unusable if set too high. Use it in combination with the Value Low Frequency to look for outliers and anomalies that are potential data quality issues for which a DQF rule should be created.
Advanced Statistics	Value Low Frequency	

Indicator category	Indicator	Notes
Summary Statistics	Minimum	This is helpful in looking for anomalies.
Summary Statistics	Maximum	This is helpful in looking for anomalies.

## Other base types

	Indicator	Notes
Simple Statistics	Row Count	This is the count of rows after an analysis filter is applied. It is not required on every field; one field per basic column analysis is sufficient.
Simple Statistics	Null Count	
Simple Statistics	Distinct Count	Case sensitivity can be set in the indicator, but it is also determined by the database collation.
Simple Statistics	Unique Count	Case sensitivity can be set in the indicator, but it is also determined by the database collation.
Simple Statistics	Duplicate Count	Case sensitivity can be set in the indicator, but it is also determined by the database collation.

## Dictionary semantic types

Base type indicators **plus** the following:

Indicator category	Indicator	Notes
Redundancy Analysis	Redundancy Analysis	This compares data in two tables.



## Regex semantic types

Base type indicators **plus** the following:

Indicator category	Indicator	Notes
Regular Expression	Regular Expression (appears in Studio as the name of the pattern)	This is a built-in or custom regex.

**Note:** The indicators listed in this section are not an exhaustive list of indicators that should be applied to a specific dataset. Rather, they are a universal set of indicators that add value to the analysis of most datasets. Where additional context is known or can be inferred from the data, Talend recommends the appropriate use of additional indicators.

## Indicator result types and DQF rule suitability

For the indicators that are applicable for usage in a Basic Column Analysis, observe that the indicators provide a variety of result “types”:

- **Count of occurrence:** On how many rows does this thing occur? For example, the Null Count indicator may return a count of 50, which means this field is null on 50 rows in the dataset. If you consider a null to be a failure, you have 50 failures in your dataset.
- **Count of criteria match:** On how many rows does this field meet the specified criteria? The Regular Expression indicator is a good example of this. For example, if you specify a regular expression that defines a valid email address on a dataset of 1000 records, the indicator might return a value of 997. This means there are 997 valid email addresses, and, by inference, there are three invalid email addresses. Ultimately, this is doing the same thing as the previous definition (**Count of occurrence**), but your perspective on how you interpret the results may be different.
- **Single value derived from aggregate analysis of the dataset:** A simple example of this is the Minimum indicator on a numerical field, which provides the lowest occurring value for a field.
- **List of values with counts:** For example, the pattern frequency indicator:

### ▼ Pattern Frequency

Value	Count	%
99999999	895	89.77%
AA999999	98	9.83%
AA99999A	2	0.20%
AAA999	1	0.10%
AAA9999	1	0.10%

The percentage shown is derived from the number of occurrences versus the total row count, but it is not persisted in the TDQ Mart.

Consider the idea of enforcing these indicators as “rules”. For reporting on row-level rules, you need a count of rows affected by a given issue. Therefore, it should be evident that indicators that provide a count are required. For a given row, you need to know simply if the result is a pass or a fail, which the **List of values with counts** does not provide. Therefore, only indicators that provide a **Count of occurrence** or a **Count of criteria match** are suitable to become row level rules. There is a nuance to this, however. Certain **Count of occurrence** indicators do not provide a count that can be directly tied to a count of rows with the failure. For example, when checking for data duplication issues you have three indicators to choose from: **Distinct Count**, **Duplicate Count**, and **Unique Count**.

Consider a dataset containing 100 rows. Three rows contain the value “Talend Ltd.” for your analyzed field. Two rows contain the value “Qlik Ltd.” for your analyzed field. The other rows all contain unique values. For the three indicators, the results would be as follows:

- **Distinct Count:** 97
- **Unique Count:** 95
- **Duplicate Count:** 2

**Distinct count** is a useful indicator if you assume that one “Talend Ltd.” row and one “Qlik Ltd.” row is correct. Therefore two “Talend Ltd.” rows and one “Qlik Ltd.” row is incorrect. However, you do not know *which* record is the correct one, so you would place all five duplicate rows into your failed rows drilldown table, versus a count of three failures for the report.

**Unique Count** does allow you to derive that there are five rows in the dataset with the duplication issue. This approach allows you to match your report count to your failed rows count, but it does not tell you how many “sets” of duplicates you have.

**Duplicate Count** tells you how many sets of duplicates you have but not the number of records in each set. The report count will not have a one-to-one correspondence with the number of failed rows.

Typically, Talend recommends using **Unique Count** or **Duplicate Count** for creating DQF duplicate check rules.

Finally, you need to consider when an indicator used in a baseline analysis can become a rule. Take the example of an email address field that in a source system is not mandatory and has no email validation at point of entry; that is, it is a free text field. Therefore, when you analyze this column you may find both blank or null values and invalid email addresses. A Data Quality Developer without the appropriate business context can guess that an email address data quality rule *might* be required, but this needs to be confirmed by someone who has the business context. Assuming it is a rule potentially skews the value of your calculated data quality scores, ironically becoming a data quality issue itself. Going a step further, the actual rule may be that the email address needs to be present and valid based on the value of another field. Again, tooling or the Data Quality Developer may attempt to infer this rule, but business context is the only way to confirm or deny the rule because the correct answer is not contained in the technical metadata of the table or view. If such a rule has not been explicitly defined by a business user on the request form, you should confirm it during the review session before being implemented and enabled on the dataset.

## Analysis and indicator combinations suitable to become DQF rules

### Basic Column Analysis

- Row Count\*
- Null Count
- Distinct Count
- Unique Count
- Duplicate Count
- Blank Count
- Default Value Count\*\*
- Regular Expression (pattern)
- SQL Pattern
- User-defined

\*Special case: When used as a DATASET scope rule with a filter. Should the row count return zero rows, it denotes a failure for the entire dataset. Thus, the failure will be included in the calculation of every row score. For more details, see the *DQF Data Dictionary*.

\* Edge case, limited use

### Redundancy Analysis

- Redundancy Analysis

### Table Analysis

- Business Rule\*
- Column set (not fully tested in DQF v1)

\* This rule type is incredibly powerful and allows you to implement complex data quality rules. See [Creating a table analysis with SQL business rules](#) in Talend Help Center.

## Business data quality rules

Unlike the Studio Business Rule object type used in table analysis, in the DQF onboarding process, business rules refer to rules that cannot be simply inferred from the semantic type of a column. Typically, these rules are expressed by business users in a syntax similar to: **If this, then that must be true** (ROW scope rules) or **X must be true for this dataset** (DATASET scope rules). These rules could have a scope of a given row, or they may apply to a set of rows within a given dataset or the entire dataset. These types of rules are usually defined by a business user using the form or in the analysis review session, or they are suggested by an experienced Data Quality Developer and confirmed by a business user. Some examples are:

- **Row scope:** The postcode must be present and valid if the company is UK based and the account status is (1,2,5).
- **Dataset scope:** The most recent ACCOUNTS\_LASTMADEUPDATE in the dataset must be within the last two months.\*

\* This is a timeliness rule, ensuring that the dataset is being refreshed.

## Reviewing analysis results with a business user

### Semantic or basic type based data quality rules

- If not already known, confirm if basic checks of validity, completeness, and uniqueness should be converted into data quality rules for ongoing monitoring. For example, are nulls/blanks allowed for this field?
- For dictionary type rules, you may require the business user to provide the allowed list of values or connectivity to a system where this list can be acquired. Consider how the reference data will be refreshed on an ongoing basis.
- For regular expression type rules, does a regex exist already in the public domain or within the organization for this check?

## Business data quality rules

### Suggest rules

Data Quality Developers are typically experts on potential pattern recognition. For example, they may identify that three columns in a dataset appear to be related in a three-level hierarchy and may have used functional dependency analyses to examine the strength of this potential relationship. A formal definition of this hierarchy may exist within the organization and the allowed values, which the business user can facilitate access to. Assuming that it is important for the data in these fields to be accurate and valid, you can propose a rule that checks each row against the official hierarchy.

### Elicit rules

After reviewing the results of baseline analysis, it is highly likely that rules will occur to a business user that they had not considered in their original request. These should be discussed, captured, confirmed, and implemented.

## Data quality dimension mapping

Talend Data Quality Framework uses the six dimensions of data quality as defined by DAMA UK (see the [DAMA DMBOK](#)):

- **Completeness:** the proportion of data stored against the potential for 100%
- **Timeliness:** the degree to which data represent reality from the required point in time
- **Validity:** the degree to which data conforms to the syntax (format, type, or range) of its definition
- **Accuracy:** the degree to which data correctly describes the real-world object or event being described
- **Consistency:** the absence of difference when comparing two or more representations of a thing against a definition
- **Uniqueness:** nothing is recorded more than once (based upon how that thing is identified)

These dimensions are a useful tool in articulating to end users the nature of different types of data quality problems. Within the framework, any rule can be mapped to any dimension using metadata. Use the review session to align each proposed rule to the dimension that makes the most sense to the business users who will consume the framework outputs.

## Scheduling and notifications

Some DQF rules warrant a proactive notification being sent when issues occur. The review session is the ideal opportunity to gather this information if it has not already been provided.

## DQF metadata

This section explains how to enter metadata into the DQF schema to onboard a dataset into the framework. You should use it alongside the *DQF Data Dictionary*.

**Before entering new metadata into an existing DQF Mart, Talend strongly recommends making a backup of the schema in case you need to roll back the changes. Failure to do so may interrupt production DQF executions.**

### Dataset metadata

#### 1. DQF\_DATASET

Add a row to this table to define the dataset.

**Note:** This requires DQF\_PERSON to hold definitions for your data owner and Data Steward for the dataset.

#### 2. DQF\_DATASET\_ATTRIBUTE (optional)

If you are using dataset-level attributes in your dashboard, enter the appropriate attributes for the new dataset.

#### 3. DQF\_DQ\_RULE

Enter a row for each rule you want to implement on the dataset according to the guidelines in the *DQF Data Dictionary*. Each rule requires a corresponding analysis/indicator written to the TDQ mart on each execution of the end-to-end DQF process. It also requires appropriate entries in the DQF\_FAILED\_QUERY\_VARIABLE table. Deactivate the rules using the IS\_ACTIVE flag, if required.

#### 4. DQF\_DQ\_RULE\_ATTRIBUTE (optional)

If you are using rule level attributes in your dashboard, enter the appropriate attributes for the new rule.

#### 5. DQF\_FAILED\_QUERY\_TEMPLATE

For the rules entered in the DQF\_DQ\_RULE table, ensure that an appropriate SQL template exists in the DQF\_FAILED\_QUERY\_TEMPLATE table. If not, insert rows into this table for each new template required. Follow the variable naming guidelines, in particular for the reserved variables [PK\_FIELD], [FILTER], and [FILTER\_A] (redundancy analysis only). The DQF SQL Templates spreadsheet serves as a useful guide for creating new templates. Ensure that new templates are thoroughly tested on the database where data is to be analyzed.

## 6. DQF\_FAILED\_QUERY\_VARIABLE

For each new data quality rule defined for your dataset, select the appropriate SQL template from the DQF\_FAILED\_QUERY\_TEMPLATE table, and add a row for each variable in the template. For the special filters, [FILTER] and [FILTER\_A], Talend recommends that you create a variable regardless of whether the original analysis has a SQL filter. In this case, set the variable to **1=1**, which evaluates to true in SQL. Ensure that there is a [PK\_FIELD] variable defined for every data quality rule, regardless of whether that filter appears in the SQL template because this is used (along with the filter) to populate the in-scope rows table during execution.

## Notification metadata

1. Enter metadata in the DQF\_PERSON, DQF\_NOT\_CHANNEL, DQF\_NOT\_CHANNEL\_PERSON, DQF\_NOT\_GROUP, and DQF\_NOT\_GROUP\_MEMBER tables as appropriate. These tables define who receives a notification and how.
2. Enter notification definitions in the DQF\_NOTIFICATION table.
3. Define subscriptions to the notifications in the DQF\_NOTIFICATION table by entering metadata in the DQF\_NOT\_SUBSCRIPTION table.



# Implementing, testing, and deploying

## Configuring the report and reporting Job

Ensure each analysis has the appropriate context added to it, so that you can configure connectivity using Talend Cloud connections. You may need to select **Window > Show View > Profiling > Context** to see the Context tab.

Column Analysis

Analysis Metadata

Data Preview

Connection: DQF\_RAW Version:0.1

New Connection

Select Columns

Select Indicators

Limit 50

n first rows

Refresh Data

Run

☐ Run with sample data

	COMPANYNUMBER	COMPANYNAME	REGADDRESS_P...	COMPANYCATEG...	COUNTRYOFORI...	REGADDRESS_C...	ACCOUNTS_LAS...	ACCOUNTS_NEX...	ACCOUNTS_ACC...	SICCODE_SICTEX...
1	08073448	BAKER CONSULTI...	RG27 9BS	Private Limited C...	United Kingdom	United Kingdom	2022-01-01	2022-03-30	MICRO ENTITY	41100 - Develop...
2	09268250	TBI VAUXHALL LI...	IG1 4TG	Private Limited C...	United Kingdom	United Kingdom	2020-10-31	2022-07-30	TOTAL EXEMPTIO...	96090 - Other ser...
3	11550193	VIPA BUSINESS SE...	NE65 9HP	Private Limited C...	United Kingdom	United Kingdom	2020-09-29	2022-06-29	MICRO ENTITY	82110 - Combine...
4	06760452	GRANDVILLE LOD...	SS9 1BG	Private Limited C...	United Kingdom	United Kingdom	2019-12-31	2021-09-29	TOTAL EXEMPTIO...	87100 - Residenti...
5	04446484	AARON PRINTER...	TQ12 2QA	Private Limited C...	United Kingdom	United Kingdom	2020-05-30	2022-02-28	TOTAL EXEMPTIO...	18129 - Printing ...
6	04800195	AMBASSADOR D...	IG6 3XJ	Private Limited C...	United Kingdom	United Kingdom	2020-06-29	2022-03-30	TOTAL EXEMPTIO...	68100 - Buying a...
7	00047684	HOTCHKISS PATE...	FY4 1EZ	Private Limited C...	United Kingdom	United Kingdom	2020-03-30	2021-12-31	DORMANT	93290 - Other am...
8	04661314	ORCHID ENVIRO...	PR7 4EZ	Private Limited C...	United Kingdom	United Kingdom	2019-12-31	2021-09-29	SMALL	38210 - Treatmen...
9	08200504	CHINESE NEW G...	WN4 9AG	Private Limited C...	United Kingdom	United Kingdom	2018-09-29	2020-09-29	TOTAL EXEMPTIO...	56101 - Licensed ...
10	09660684	DINGLE AEROSP...	SO51 7HX	Private Limited C...	United Kingdom	United Kingdom	2020-06-29	2022-03-30	MICRO ENTITY	33160 - Repair an...

Analyzed Columns

Select Indicators

Run

Go to page 1/2

Analysis Settings

Analysis Results

Contexts(A01\_DQF\_DEMO\_DATASET1\_BCA 0.1)

Talend Git Staging

Run job

Component

	Name	Type	Comment	Default
				Value
1	DQF_CONNECTIONS (from repository c...			
2	connection_DQF_driver_class	String		net.snowflake.client.jdbc.SnowflakeDriver
3	connection_DQF_drivers	String		"mvn:net.snowflake/snowflake-jdbc/3.13.14/jar"
4	connection_DQF_jdbc_url_mart	String	jdbc:snowflake://	
5	connection_DQF_jdbc_url_raw	String	jdbc:snowflake://	
6	connection_DQF_password	Password		*****
7	connection_DQF_user	String		

Default context environment

Default

Add all analyses that you want to execute on each run of the DQF process to a report. Refer to [Reports](#) in Talend Help Center. This can include analyses/indicators that are not used as part of DQF rules. Running these on a regular basis can assist with root cause analysis when a new issue arises. Ensure each analysis is set to **Evolution** mode to store history over time in the TDQ Mart.


### ▼ Analysis List

[Select analyses](#)

Analysis	Execution Date	Refresh	Template type			Remove
 A01_DQF_DEMO_DAT...	10 Feb 2023, 14:53:36	<input checked="" type="checkbox"/>	Evolution	▼	<input type="text" value="Browse..."/>	<input checked="" type="checkbox"/>
 A02_DQF_DEMO_DAT...	26 Jan 2023, 16:33:10	<input checked="" type="checkbox"/>	Evolution	▼	<input type="text" value="Browse..."/>	<input checked="" type="checkbox"/>
 A03_DQF_DEMO_DAT...	26 Jan 2023, 16:37:18	<input checked="" type="checkbox"/>	Evolution	▼	<input type="text" value="Browse..."/>	<input checked="" type="checkbox"/>
 A04_DQF_DEMO_DAT...	26 Jan 2023, 16:38:00	<input checked="" type="checkbox"/>	Evolution	▼	<input type="text" value="Browse..."/>	<input checked="" type="checkbox"/>

Disable the **Generate output file** option.

### ▼ Generated Report Settings

☐ Generate output file 

Output Folder

Output FileName  ☒ with timestamp

File Type

Analysis executed between  -  (Date pattern: MM/dd/yyyy)

Logo:

Top Left

Top Middle

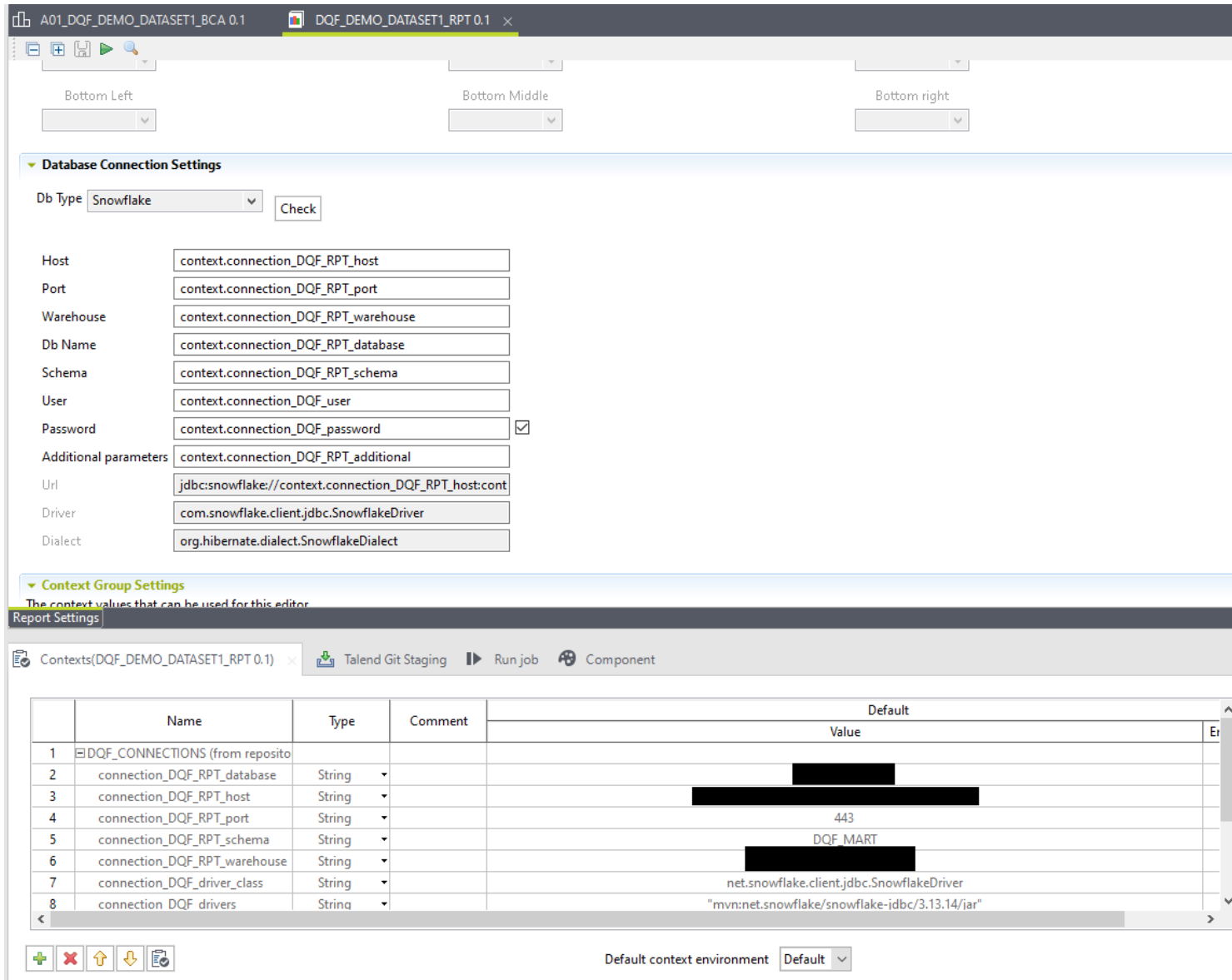
Top Right

Bottom Left

Bottom Middle

Bottom right

Add the appropriate context to the report.



The screenshot shows the Talend Studio interface with two tabs: 'A01\_DQF\_DEMO\_DATASET1\_BCA 0.1' and 'DQF\_DEMO\_DATASET1\_RPT 0.1'. The 'DQF\_DEMO\_DATASET1\_RPT 0.1' tab is active, displaying the 'Database Connection Settings' and 'Context Group Settings' sections.

**Database Connection Settings:**

- Db Type: Snowflake (dropdown)
- Check (button)
- Host: context.connection\_DQF\_RPT\_host
- Port: context.connection\_DQF\_RPT\_port
- Warehouse: context.connection\_DQF\_RPT\_warehouse
- Db Name: context.connection\_DQF\_RPT\_database
- Schema: context.connection\_DQF\_RPT\_schema
- User: context.connection\_DQF\_user
- Password: context.connection\_DQF\_password (checked)
- Additional parameters: context.connection\_DQF\_RPT\_additional
- Url: jdbc:snowflake://context.connection\_DQF\_RPT\_host:cont
- Driver: com.snowflake.client.jdbc.SnowflakeDriver
- Dialect: org.hibernate.dialect.SnowflakeDialect

**Context Group Settings:**

The context values that can be used for this editor

Report Settings


Contexts(DQF\_DEMO\_DATASET1\_RPT 0.1)

Talend Git Staging | Run job | Component

	Name	Type	Comment	Default	
				Value	Environment
1	DQF_CONNECTIONS (from repository)				
2	connection_DQF_RPT_database	String			
3	connection_DQF_RPT_host	String			
4	connection_DQF_RPT_port	String		443	
5	connection_DQF_RPT_schema	String		DQF_MART	
6	connection_DQF_RPT_warehouse	String			
7	connection_DQF_driver_class	String		net.snowflake.client.jdbc.SnowflakeDriver	
8	connection_DQF_drivers	String		"mvn:net.snowflake/snowflake-jdbc/3.13.14/jar"	

Default context environment: Default (dropdown)

In the Data Integration perspective, make a copy of the **A20\_Run\_Dataset1\_RPT** template Job and name it appropriately. Point the new Job to your new report using the **tDQReportRun** component **Browse Reports** button, and set the output folder to **/tmp**.



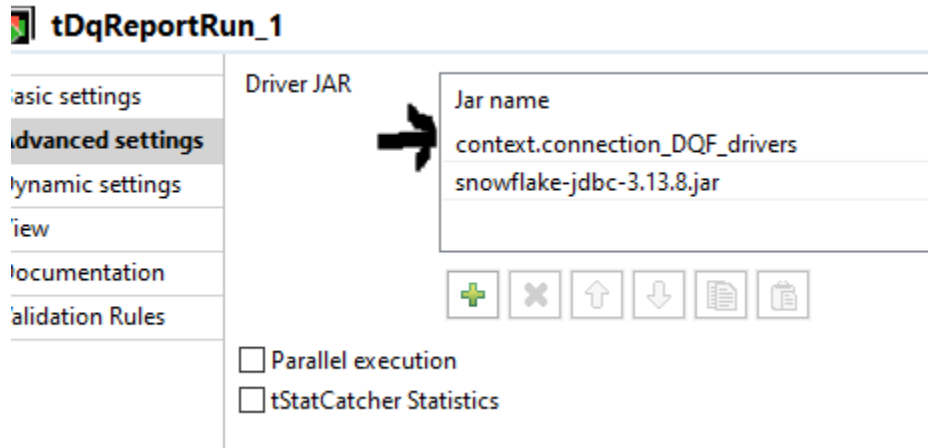
**tDQReportRun\_1**

Schema: Built-In Edit schema

Refresh Dependency Browse Reports Report filenames: "/DQF/TDQ\_Data Profiling/Reports/DQF\_DEMO/DQF\_DEMO\_DATASET1\_RPT\_0.1.rep"

Output folder: "/tmp"

Click **Advanced settings** and remove the entry for the context driver. Add the actual driver if required.



**tDQReportRun\_1**

Driver JAR

Jar name
context.connection_DQF_drivers
snowflake-jdbc-3.13.8.jar

Parallel execution

tStatCatcher Statistics

## Avoiding conflicts with existing DQF Runs

Do not attempt to test run any DQF Job while a production DQF run is in progress.

## Executing a test run and troubleshooting

Make sure a DQF Run is not in progress or scheduled to commence imminently, then follow this procedure to perform a test run on your new dataset:

1. On Talend Management Console (TMC), execute the **DQF\_10\_Initialise\_DQF\_Run** task.
2. In Studio, execute your newly created Job that contains the **tdQReportRun** component. Ensure that it completes successfully.
3. This step is optional, but recommended. On Talend Management Console (TMC), create a new task that replicates the task **DQF\_30\_Dataset\_Run\_Population**, using the same artifact/version. Set the **DATASET\_PK\_LIST** context parameter to the **DATASET\_PK** that identifies your new dataset from the DQF\_DATASET table.
4. Execute the new or existing **DQF\_30\_Dataset\_Run\_Population** task as appropriate.
5. Execute the **DQF\_40\_Failed\_Rows\_ELT** task. Check the logs thoroughly. This Job constructs and executes the SQL templates, and it may fail if mistakes have been made in metadata population (DQF\_FAILED\_QUERY\_VARIABLE) or SQL template definition (DQF\_FAILED\_QUERY\_TEMPLATE). Thoroughly check the results populated into the DQF\_FAILED\_ROW table, and ensure the counts of each error match the counts generated by the analysis (from Studio execution of the analysis or by looking at the counts recorded in the TDQ mart). If you need to correct an issue and repeat the execution of this task, you first need to clean up the DQF\_FAILED\_ROW table by executing the following SQL statement:

```
DELETE FROM DQF_FAILED_ROW where DQF_RUN_ID_START_FK = <CURRENT DQF RUN ID from DQF_RUN>
```

**Note:** This assumes you are executing the first test run for this dataset.

6. Execute the **DQF\_50\_In\_Scope\_Rows\_ELT** task. Check the logs thoroughly. This Job constructs and executes a SQL SELECT statement, and it may fail mistakes have been made in metadata population (DQF\_FAILED\_QUERY\_VARIABLE). Thoroughly check the results populated into the DQF\_ROW\_RULE\_HISTORY table and ensure the counts of each row match the counts in the source. If you need to correct an issue and repeat the execution of this task, you need to first clean-up the DQF\_ROW\_RULE\_HISTORY table by executing the following SQL statement:

```
DELETE FROM DQF_ROW_RULE_HISTORY where DQF_RUN_ID_START_FK = <CURRENT DQF RUN ID from DQF_RUN>
```

**Note:** This assumes you are executing the first test run for this dataset.

7. Create a table in the DQF mart schema by copying the DDL for the Z\_CUSTOMER\_DS1 table. Name the table **Z\_<DQF\_DATASET.SOURCE\_OBJECT\_NAME>\_DS< DQF\_DATASET.DATASET\_PK>** .

**Important:** strictly follow this naming convention or the next task will fail.

8. Execute the **DQF\_60\_Generate\_Failed\_Rows\_Report** task, and check that the expected results are populated into your new table.
9. Execute the **DQF\_70\_PopulateNotifications** task, and check that the expected populations are in the DQF\_NOTIFICATION table.
10. Execute the **DQF\_80\_Notification\_Send** task, and check that the expected notifications are sent.
11. Execute the **DQF\_85\_Refresh\_Materialized\_Views** task and ensure there is data in the MV\_ tables (MySQL only)
12. Execute the **DQF\_90\_Finalize\_Run\_Complete** task to finish the run.

**Important:** You must execute this task before the next scheduled production DQF run, or the test run will be marked as failed.

## Cleaning up the TDQ mart

You can remove the execution of a report from the TDQ mart using the approach outlined in [Deleting a report from the data mart](#) in Talend Help Center. Failing to remove the corresponding DQF run data will cause reporting issues. Exercise extreme caution when deleting reports (all runs) or specific runs of reports (modifications to SQL provided in the guide are required), because you may inadvertently delete data associated with production DQF runs.

## Dataset and DQF run relationship

A DQF run can contain one to *n* datasets as defined by providing a comma separated list of DATSET\_PKs to the **DQF\_30\_Dataset\_Run\_Population** task and running the associated profiling reports just prior to this task being executed. You can schedule more than one end-to-end DQF process on TMC as long as there is no possibility of the timings overlapping.

## Deploying and scheduling

**Option A:** Add the new dataset to an existing DQF run execution plan by deploying and adding the new report execution Job to TMC and inserting it into the appropriate step in the existing execution plan.

**Option B:** Create a new execution plan according to the process outlined in the *Talend Data Quality Framework Installation Guide* (download from the [Talend Data Quality Framework](#) page on Talend Academy) for the DQF demo. This approach requires a separate task for the **DQF\_30\_Dataset\_Run\_Population** step with appropriate values for the **DATASET\_PK\_LIST** context parameter in each instance. When you schedule the execution plans, make sure there is no possibility of two DQF runs overlapping.