Talend Data Quality Framework – Installation Guide

Introduction

This document describes the process to install, configure, and test the Talend Data Quality Framework (DQF). 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.

Prerequisites

Technical prerequisites for the Talend Data Quality Framework as well as the training and knowledge requirements for users of the framework are detailed in **DQF Architecture and Prerequisites Guide**.

Recommended knowledge

- Snowflake or MySQL database administration
- Executing Jobs from Talend Studio
- Deploying Jobs from Talend Studio to Talend Management Console
- Talend Management Console administration

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 using a Remote Engine with Java 11
  - Version **8** with monthly release **R20223-04** or higher applied if using a cloud engine or a Remote Engine with 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 Services:

- Enable Analytics
- Establish Data Excellence

Environment setup

This guide assumes that your Talend Cloud environment is installed and configured according to Talend guidance before starting the installation.

Framework specific database requirements

A supported database– this release supports either:

- MySQL 8 or a compatible cloud offering (for example, AWS Aurora)
- Snowflake

Different RDBMS technologies use the terms **database** and **schema** in different ways. The following sections detail the DQF requirements for each supported database and standardizes terminology for the remainder of this guide:

Snowflake

A Snowflake instance is known as the **tenant** or the **account**. A Snowflake instance contains one to many databases and each Snowflake database contains one to many schemas. These schemas contain tables and views.

The DQF requirements for Snowflake are as follows:

- A Snowflake database in a Snowflake tenant– Talend recommends naming this database **DQF**. In this guide, this database is referred to as the **DQF Database**.
- A schema in the **DQF Database** for the DQF data mart– Talend recommends naming this schema **DQF_MART**. In this guide, this schema is called the **DQF Mart Schema**.
- A schema in the **DQF Database** for DQF data analysis– Talend recommends naming this schema **DQF_RAW**. In this guide, this schema is referred to as the **DQF Raw Schema**.

MySQL 8

An instance of MySQL is known as a database. Each MySQL database contains one to many schemas, and these schemas contain tables and views.

The DQF requirements for MySQL are as follows:

- A MySQL 8 database– In this guide, this database is referred to as the **DQF Database**. However, an exception to this exists when specifying connection parameters, which will be detailed later in this guide.
- A schema in the **DQF Database** for the DQF data mart– Talend recommends naming this schema **DQF_MART**. In this guide, this schema is called the **DQF Mart Schema**.
- A schema in the **DQF Database** for DQF data analysis– Talend recommends naming this schema **DQF_RAW**. In this guide, this schema is referred to as the **DQF Raw Schema**.

Other framework specific requirements

- A GitHub repository and associated Talend Cloud project that are used for all DQF templates, examples, and development. In this guide, this project is called the **DQF Project**.
- One or more Talend Remote Engines (Generation 1) used to execute the DQF processes. The Engine(s) should have suitable access to source systems, the **DQF Database**, and any desired notification systems (email and Slack are supported in this release).
- A Talend Studio and associated Talend Cloud account with permissions to develop in the **DQF Project** and deploy Jobs to Talend Cloud.
- A Talend Cloud account with permissions to:
    - Create environments (optional)
    - Create workspaces
    - Deploy Jobs (requires a personal access token or service account)
    - Create connections
    - Create, configure, and execute tasks
    - Use Talend Data Preparation (optional)

Refer to the **DQF Architecture and Prerequisites Guide** for complete environment details.

Required downloads

- DQF core deployment bundle v1.1.0.zip
- DQF template and demo DI metadata v1.1.0.zip
- DQF template and demo DQ metadata v1.1.0.zip
- DQF demo dashboard PowerBI Snowflake v1.1.0.pbix or DQF demo dashboard PowerBI MySQL v1.1.0.pbix
- DQF mart DDL v1.1.0.zip
- DQF demo DDL v1.1.0.zip

Contents

Preparing the DQF Mart Schema

1. Using Talend Studio, follow the instructions in Managing the report database from the Talend Studio User Guide to create the Profiler Data Mart in the **DQF Mart Schema**.

   Note: Only Snowflake and MySQL 8 are supported for this release of the Talend Data Quality Framework.

2. Unzip **DQF mart DDL v1.x.zip**.

   Snowflake:

   Edit the **DQF tables snowflake v1.x.sql** script and set the correct values for the **USE SCHEMA DQF.DQF_MART** statement. The syntax is:

```
USE SCHEMA <DQF Database> .<DQF Mart Schema>;
```

   MySQL:

   Edit the **DQF tables mysql v1.x.sql** script and set the correct values for the **USE DQF_MART** statement. The syntax is:

```
USE < DQF Mart Schema>;
```

3. Execute the script using your SQL client tool and confirm that all create table statements execute successfully.

   Snowflake:

   Edit the **DQF views snowflake v1.x.sql** script and set the correct values for the **USE SCHEMA DQF.DQF_MART** statement. The syntax is:

```
USE SCHEMA <DQF Database> .<DQF Mart Schema>
```

MySQL:

Edit the **DQF views mysql v1.x.sql** script and set the correct values for the **USE DQF_MART** statement. The syntax is:

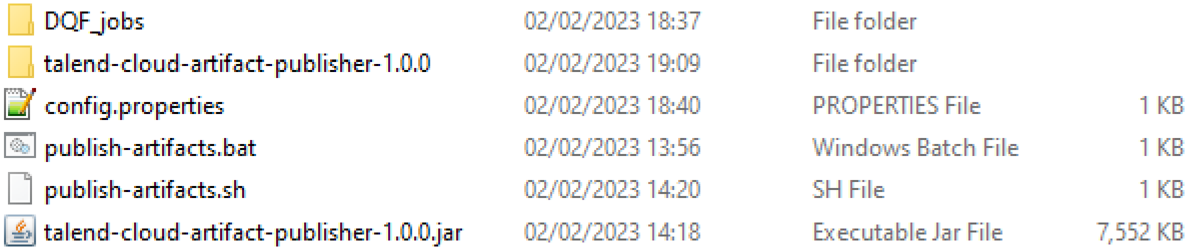
```
USE < DQF Mart Schema>;
```

4. Execute the script using your SQL client tool and confirm that all create view statements execute successfully.

Deploying the DQF core Jobs to Talend Management Console

1. If your desired target environment and workspace do not already exist in Talend Management Console (TMC), create them and assign the appropriate access permissions. Refer to Creating environments and Managing workspaces in Talend Help Center for more information.

2. Unzip **DQF core deployment bundle v1.x.zip**.

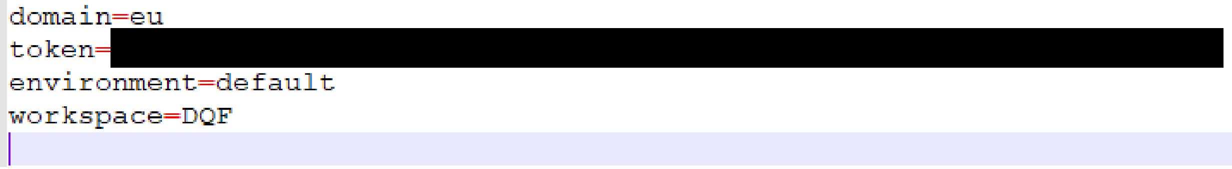


DQF_jobs	02/02/2023 18:37	File folder	
talend-cloud-artifact-publisher-1.0.0	02/02/2023 19:09	File folder	
config.properties	02/02/2023 18:40	PROPERTIES File	1 KB
publish-artifacts.bat	02/02/2023 13:56	Windows Batch File	1 KB
publish-artifacts.sh	02/02/2023 14:20	SH File	1 KB
talend-cloud-artifact-publisher-1.0.0.jar	02/02/2023 14:18	Executable Jar File	7,552 KB

3. Edit **config.properties**:

```
domain=<TMC domain: eu | ap | us | au | us-west >
environment=<TMC target environment>
workspace=<TMC target workspace>
token=<personal or service account access token>
```

For example:

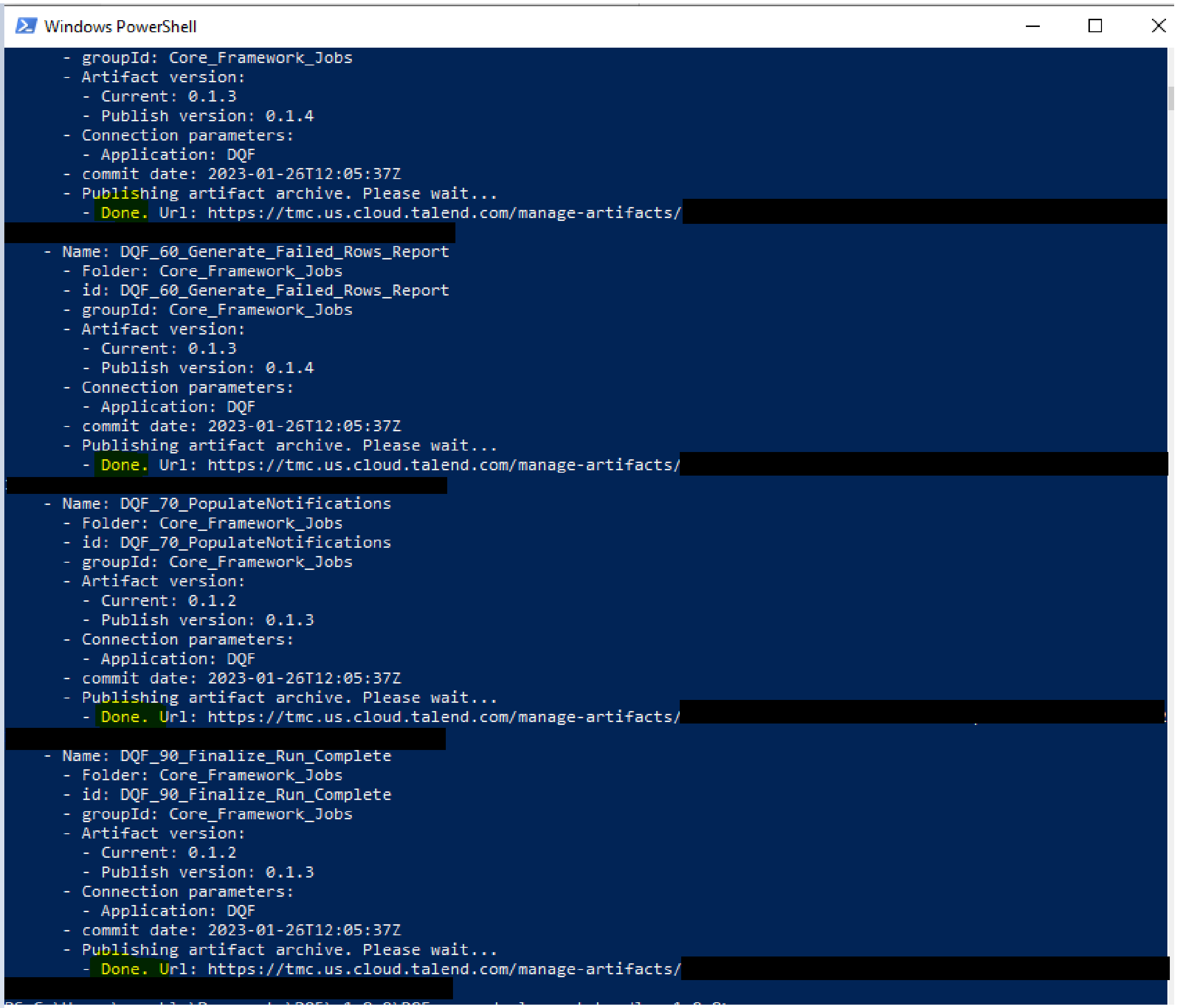


```
domain=eu
token=[redacted]
environment=default
workspace=DQF
```

Refer to URLs to Talend Cloud applications in Talend Help Center for a description of the available data centers and regions.

4. In the command line/shell, browse to the folder where the zip was extracted and execute **publish-artifacts.bat** (Windows) or **publish-artifacts.sh** (Linux), and ensure the script executes successfully.

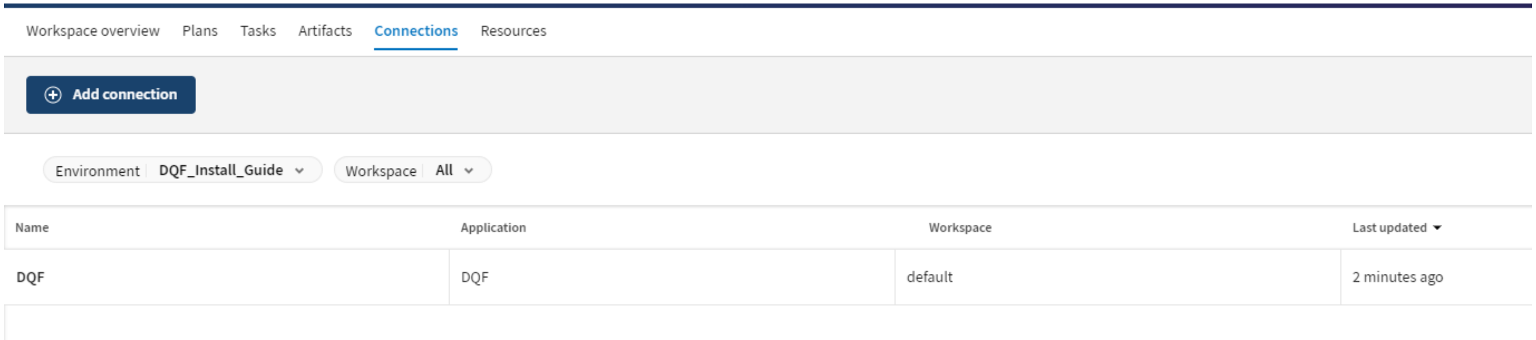
   Note: The script requires Java 11 or higher to be installed and available as part of the PATH. Refer to [How do I set or change the PATH system variable?]() for help. The script creates a connection called **DQF** in the target environment.



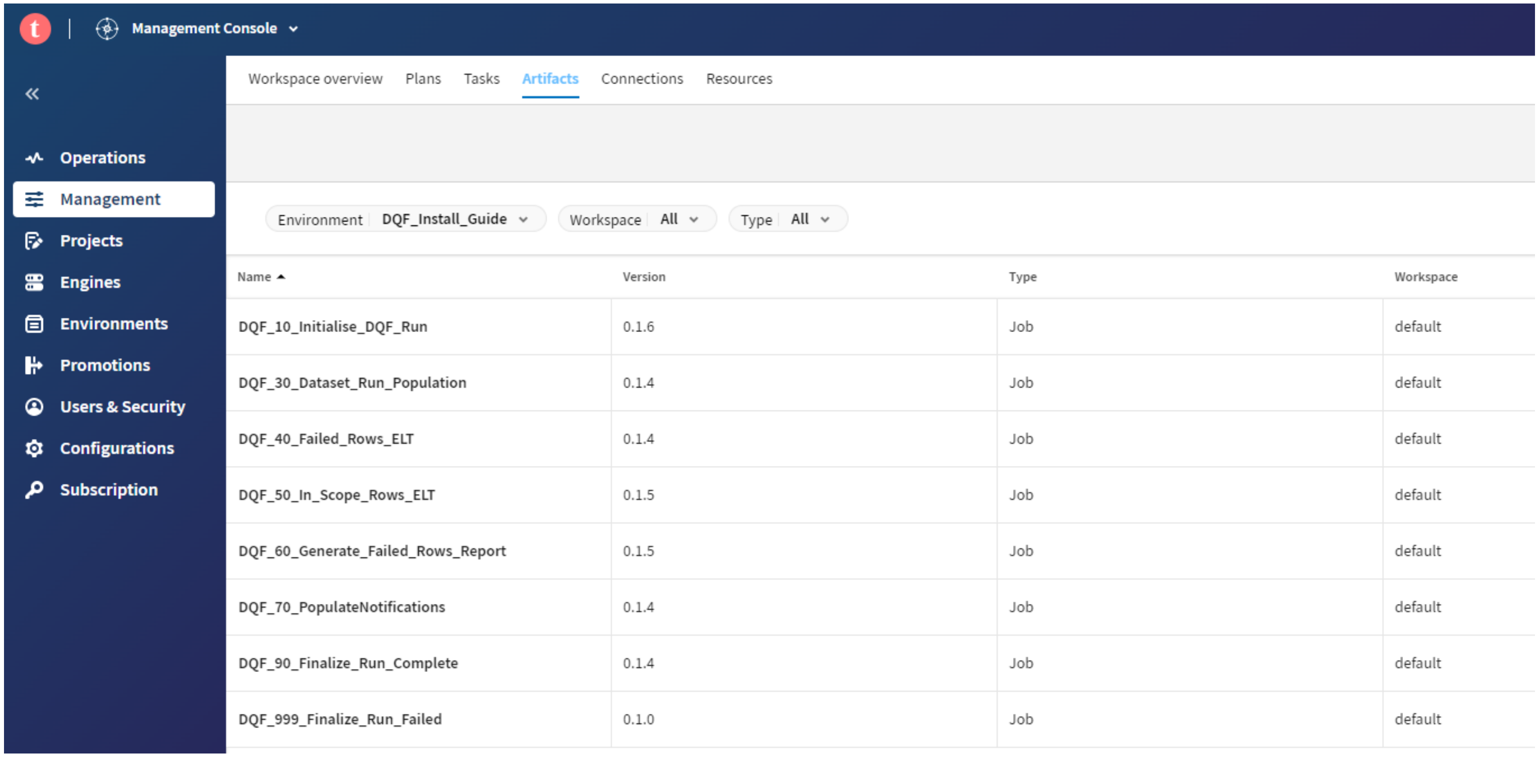
Windows PowerShell

```
      - groupId: Core_Framework_Jobs
      - Artifact version:
        - Current: 0.1.3
        - Publish version: 0.1.4
      - Connection parameters:
        - Application: DQF
      - commit date: 2023-01-26T12:05:37Z
      - Publishing artifact archive. Please wait...
        - Done. Url: https://tmc.us.cloud.talend.com/manage-artifacts/[redacted]
    - Name: DQF_60_Generate_Failed_Rows_Report
      - Folder: Core_Framework_Jobs
      - id: DQF_60_Generate_Failed_Rows_Report
      - groupId: Core_Framework_Jobs
      - Artifact version:
        - Current: 0.1.3
        - Publish version: 0.1.4
      - Connection parameters:
        - Application: DQF
      - commit date: 2023-01-26T12:05:37Z
      - Publishing artifact archive. Please wait...
        - Done. Url: https://tmc.us.cloud.talend.com/manage-artifacts/[redacted]
    - Name: DQF_70_PopulateNotifications
      - Folder: Core_Framework_Jobs
      - id: DQF_70_PopulateNotifications
      - groupId: Core_Framework_Jobs
      - Artifact version:
        - Current: 0.1.2
        - Publish version: 0.1.3
      - Connection parameters:
        - Application: DQF
      - commit date: 2023-01-26T12:05:37Z
      - Publishing artifact archive. Please wait...
        - Done. Url: https://tmc.us.cloud.talend.com/manage-artifacts/[redacted]
    - Name: DQF_90_Finalize_Run_Complete
      - Folder: Core_Framework_Jobs
      - id: DQF_90_Finalize_Run_Complete
      - groupId: Core_Framework_Jobs
      - Artifact version:
        - Current: 0.1.2
        - Publish version: 0.1.3
      - Connection parameters:
        - Application: DQF
      - commit date: 2023-01-26T12:05:37Z
      - Publishing artifact archive. Please wait...
        - Done. Url: https://tmc.us.cloud.talend.com/manage-artifacts/[redacted]
```

5. Verify that the connection and artifacts are visible in Talend Management Console.



Name	Application	Workspace	Last updated
DQF	DQF	default	2 minutes ago



Name	Version	Type	Workspace
DQF_10_Initialise_DQF_Run	0.1.6	Job	default
DQF_30_Dataset_Run_Population	0.1.4	Job	default
DQF_40_Failed_Rows_ELT	0.1.4	Job	default
DQF_50_In_Scope_Rows_ELT	0.1.5	Job	default
DQF_60_Generate_Failed_Rows_Report	0.1.5	Job	default
DQF_70_PopulateNotifications	0.1.4	Job	default
DQF_90_Finalize_Run_Complete	0.1.4	Job	default
DQF_999_Finalize_Run_Failed	0.1.0	Job	default

6. Configure the new **DQF** connection with the appropriate values for your environment:

Parameter Name	Description	Example value
password	Password to access the DQF database	
jdbc_url_mart	Full JDBC URL to the DQF Mart Schema **Note**: For Snowflake, the TIMEZONE=UTC parameter is required.	**Snowflake**: `jdbc:snowflake://mytenant.snowflakecomputing.com/?warehouse=MY_WH&db=DQF&schema=DQF_MART&role=MY_ROLE&TIMEZONE=UTC` **MySQL**: `jdbc:mysql://mysqldb:3306/DQF_MART?characterEncoding=UTF8&noDatetimeStringSync=true&enabledTLSProtocols=TLSv1.2,TLSv1.1,TLSv1`

drivers	Maven path to the appropriate JDBC driver
	The double quotes are required.
		Snowflake:
		`"mvn:net.snowflake/snowflake-jdbc/3.13.14/jar"`
		MySQL:
		`"mvn:mysql/mysql-connector-java/8.0.18/jar"`

driver_class	JDBC driver class	**Snowflake**:
		`net.snowflake.client.jdbc.SnowflakeDriver`
		MySQL:
		`com.mysql.cj.jdbc.Driver`

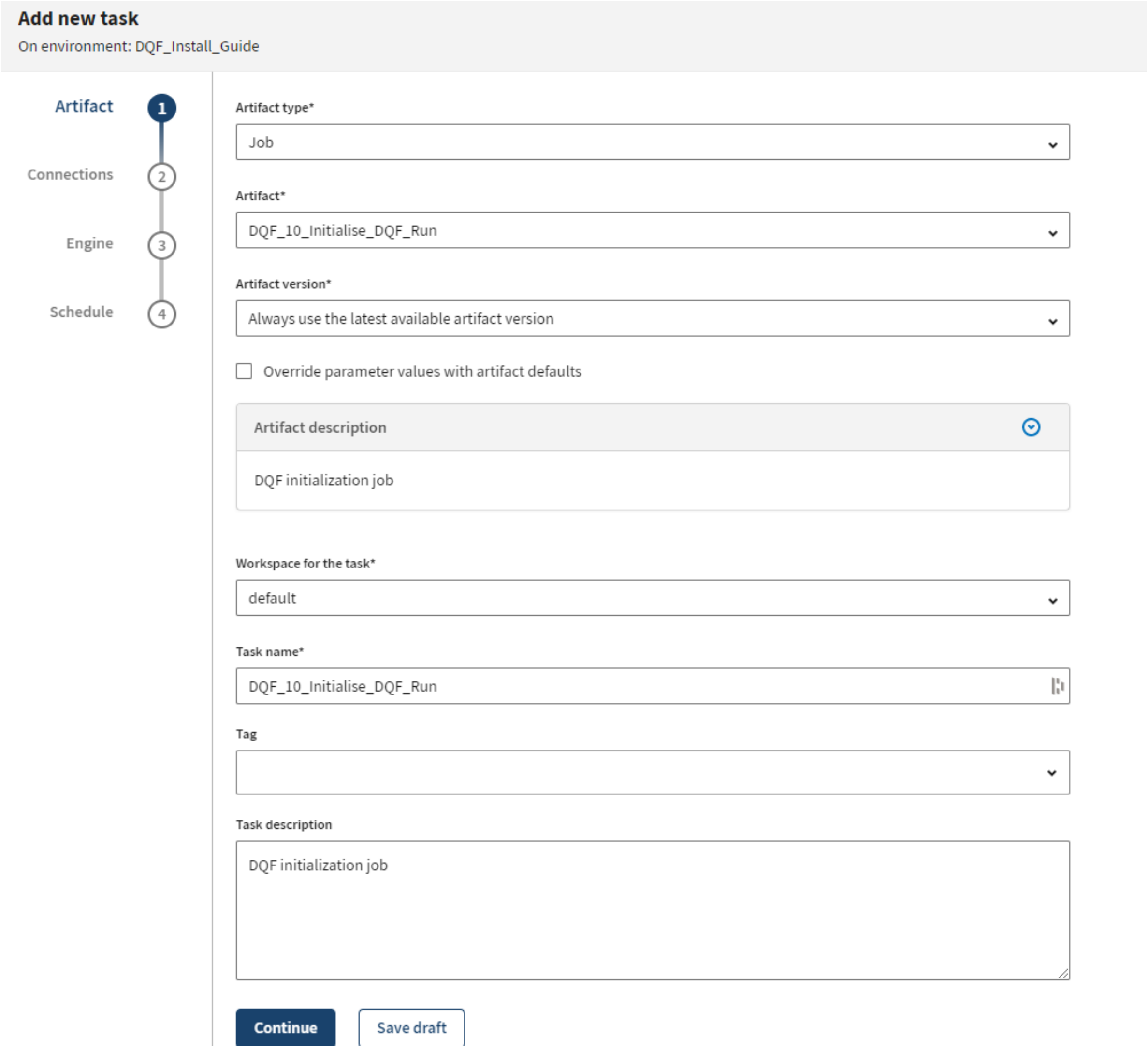
RPT_database	DQF database **NOTE:** set to DQF MART Schema for MySQL, in addition to setting the RPT_schema parameter to the same value	**Snowflake**: DQF **MySQL**: DQF_MART
RPT_port	DQF database port For Snowflake, use 443	**Snowflake:** `443` **MySQL:** `3306`

Parameter Name	Description	Example value
RPT_schema	DQF Mart Schema	**Snowflake**: `DQF_MART` **MySQL**: `DQF_MART`
user	DQF database user The same user must be used to access both the DQF Mart Schema and the DQF Raw Schema.	`myuser`
RPT_warehouse	Snowflake warehouse to be used to execute DQF actions. Leave blank for other databases.	`MY_WH`

RPT_additional	Additional JDBC parameters
	Note: This parameter is used to execute the Data Quality Reports and differs from the recommended additional JDBC parameters in the jdbc_url_mart and jdbc_url_raw parameters.
		Snowflake:
		`role=MY_ROLE&TIMEZONE=UTC&CLIENT_RESULT_COLUMN_CASE_INSENSITIVE=true`
		MySQL:
		`characterEncoding=UTF8&noDatetimeStringSync=true&enabledTLSProtocols=TLSv1.2,TLSv1.1,TLSv1`

RPT_host	DQF database host	`mytenant.snowflakecomputing.com`
jdbc_url_raw	Full JDBC URL to DQF Raw Schema **Note**: For Snowflake, the TIMEZONE=UTC parameter is required.	**Snowflake**: `jdbc:snowflake://mytenant.snowflakecomputing.com/?warehouse=MY_WH&db=DQF &schema=DQF_RAW&role=MY_ROLE&TIMEZONE=UTC` **MySQL**: jdbc:mysql://`mysqldb`:3306/DQF_RAW?characterEncoding=UTF8&noDatetimeStringSync=true&enabledTLSProtocols=TLSv1.2,TLSv1.1, TLSv1

7. Check to see if there is a task for each DQF artifact and if not, create a task. Refer to Creating Job Tasks in Talend Help Center for more information.



Add new task

On environment: DQF_Install_Guide

Artifact type*: Job

Artifact*: DQF_10_Initialise_DQF_Run

Artifact version*: Always use the latest available artifact version

□ Override parameter values with artifact defaults

Artifact description

DQF initialization job

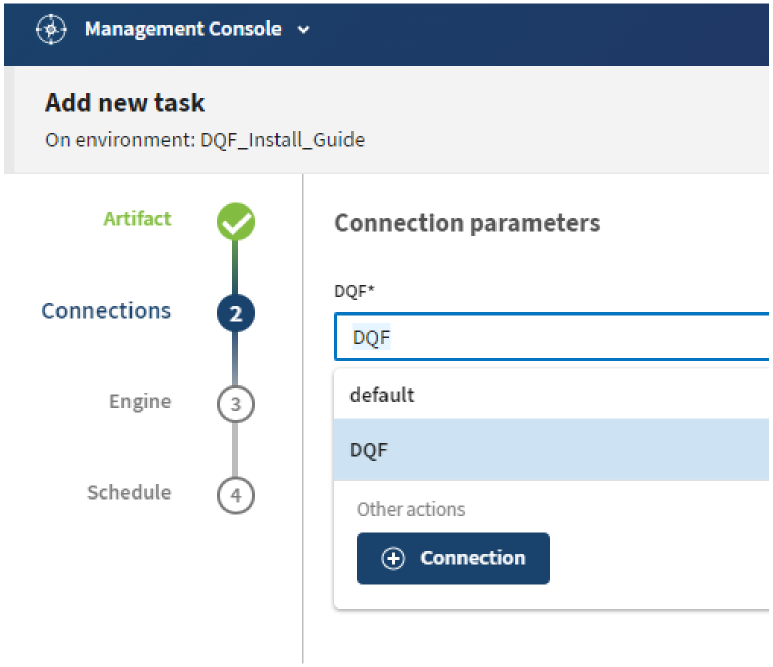
Workspace for the task*: default

Task name*: DQF_10_Initialise_DQF_Run

Tag:

Task description: DQF initialization job

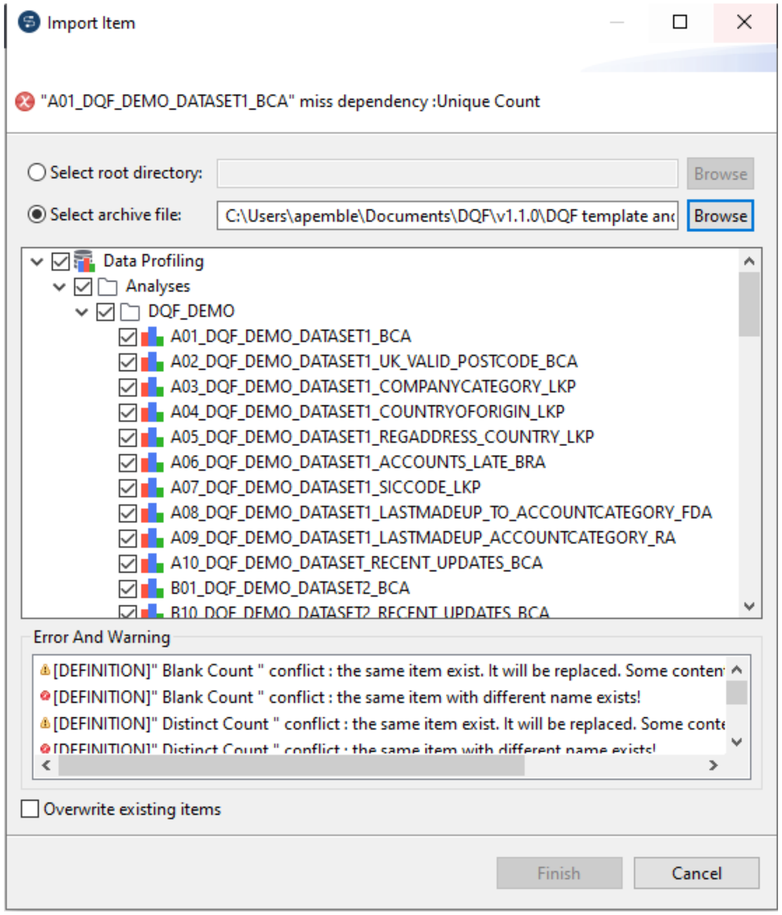
8. Configure each task to use the DQF connection.



9. Set the Job to execute on your Remote Engine (or a Cloud Engine if your studio version is R2023-04 or higher) of choice and save it without setting a schedule.

Importing the template and demo artifacts into your DQF Talend project

1. Import **DQF template and demo DQ metadata v1.0.0.zip** into the profiling perspective of your DQF Talend Studio project. As part of this process, you will see a number of errors and warnings listed in the import wizard:



"A01_DQF_DEMO_DATASET1_BCA" miss dependency :Unique Count

○ Select root directory: Browse

● Select archive file: C:\Users\apemble\Documents\DQF\v1.1.0\DQF template and Browse

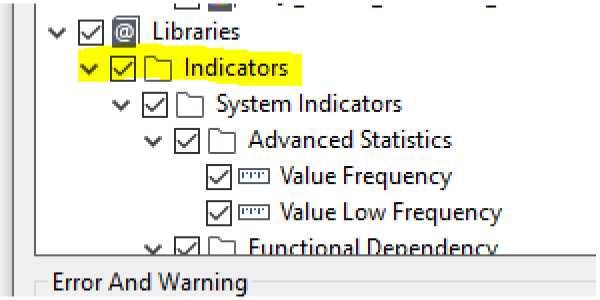
☑ Data Profiling
- ☑ Analyses
  - ☑ DQF_DEMO
    - ☑ A01_DQF_DEMO_DATASET1_BCA
    - ☑ A02_DQF_DEMO_DATASET1_UK_VALID_POSTCODE_BCA
    - ☑ A03_DQF_DEMO_DATASET1_COMPANYCATEGORY_LKP
    - ☑ A04_DQF_DEMO_DATASET1_COUNTRYOFORIGIN_LKP
    - ☑ A05_DQF_DEMO_DATASET1_REGADDRESS_COUNTRY_LKP
    - ☑ A06_DQF_DEMO_DATASET1_ACCOUNTS_LATE_BRA
    - ☑ A07_DQF_DEMO_DATASET1_SICCODE_LKP
    - ☑ A08_DQF_DEMO_DATASET1_LASTMADEUP_TO_ACCOUNTCATEGORY_FDA
    - ☑ A09_DQF_DEMO_DATASET1_LASTMADEUP_ACCOUNTCATEGORY_RA
    - ☑ A10_DQF_DEMO_DATASET_RECENT_UPDATES_BCA
    - ☑ B01_DQF_DEMO_DATASET2_BCA
    - ☑ B10_DQF_DEMO_DATASET2_RECENT_UPDATES_BCA

Error And Warning

```
[DEFINITION]" Blank Count " conflict : the same item exist. It will be replaced. Some conten
[DEFINITION]" Blank Count " conflict : the same item with different name exists!
[DEFINITION]" Distinct Count " conflict : the same item exist. It will be replaced. Some conte
[DEFINITION]" Distinct Count " conflict : the same item with different name exists!
```

☐ Overwrite existing items

2. To allow successful import, perform the following steps:
   - Click **Overwrite existing items**.
   - Select the **Indicators** check box to import all the indicators.



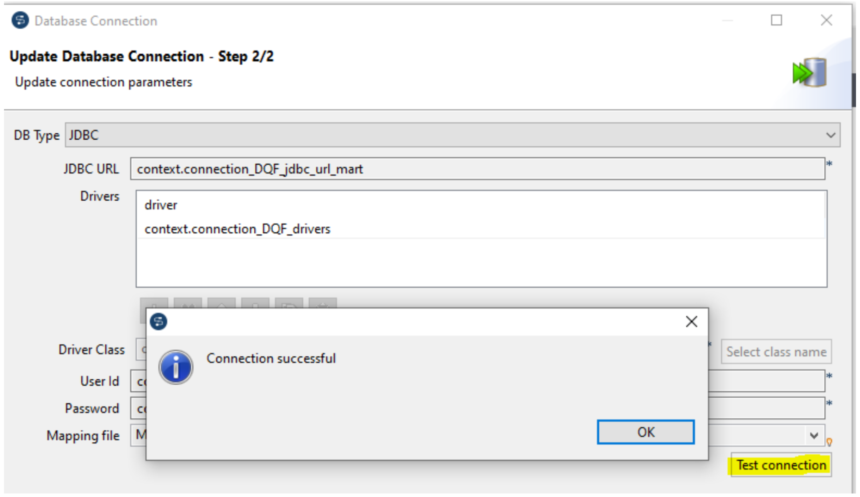
☑ Libraries
☑ Indicators
☑ System Indicators
☑ Advanced Statistics
☑ Value Frequency
☑ Value Low Frequency
☑ Functional Dependency
Error And Warning

   - Click **Finish**.

3. Import **DQF template and demo DI metadata v1.0.0.zip** into the Data Integration perspective of your DQF Talend Studio project. You can ignore the warning about the context already existing, you do not need to tick the **Overwrite existing items** check box.

4. Modify the **DQF_CONNECTIONS** context group to set your connection parameters. This is usually the same set of parameters you entered in Talend Management Console in the previous section. Adding the parameters to Talend Studio enables you to test the Jobs you develop prior to deployment to Talend Cloud. Allow the changes to propagate to all impacted artifacts.

5. Open the **DQF_MART** metadata connection, set the appropriate DB mapping type, and test it. Troubleshoot if it is unsuccessful.



Update Database Connection - Step 2/2

Update connection parameters

DB Type: JDBC

JDBC URL: context.connection_DQF_jdbc_url_mart *

Drivers:
driver
context.connection_DQF_drivers

Driver Class

User Id

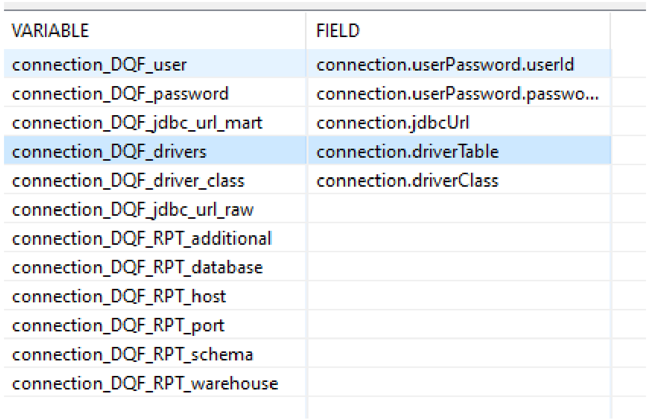
Password

Mapping file

Connection successful

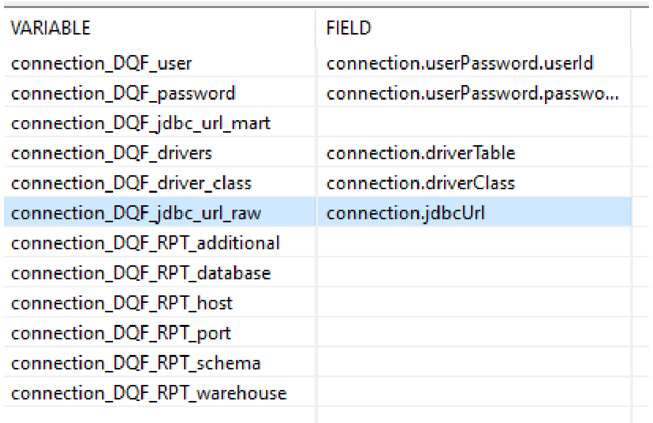
Known Issue: Sometimes the password context variable may appear to be missing from the connection, which causes the connection test to fail. To fix this, revert the context and then export the context again. Select the option to use an existing context (DQF_CONNECTIONS in this case) and map the connection variables to the appropriate values in the context:

DQF_MART:



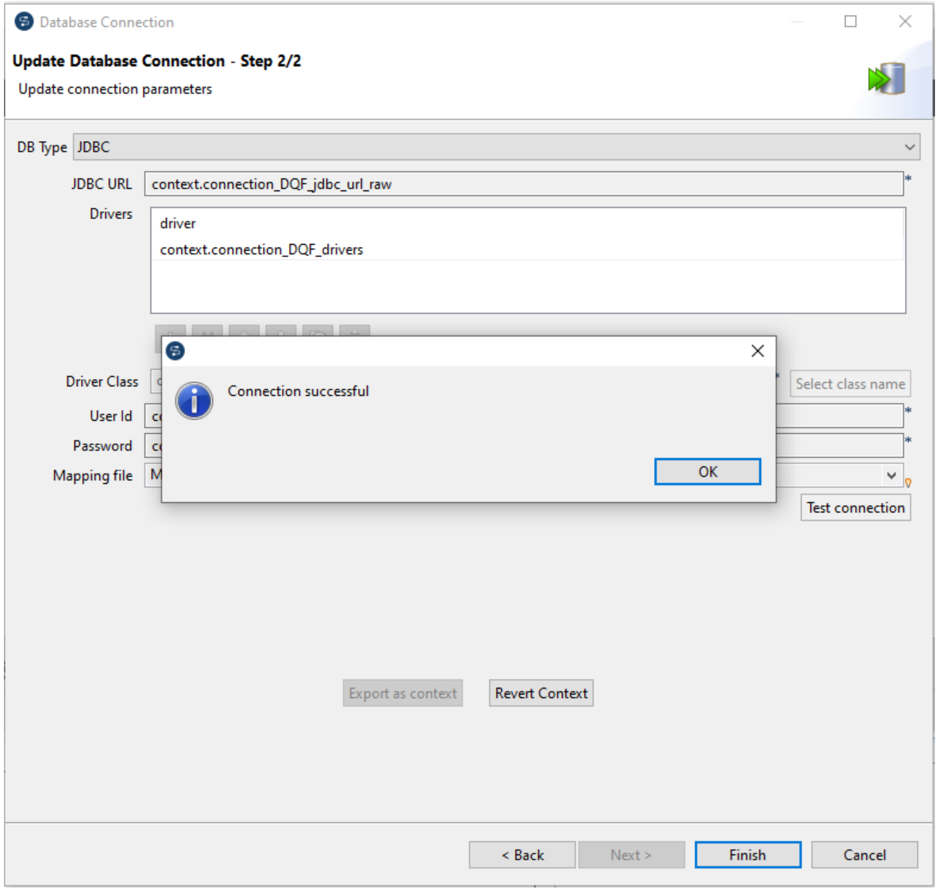
VARIABLE	FIELD
connection_DQF_user	connection.userPassword.userId
connection_DQF_password	connection.userPassword.passwo...
connection_DQF_jdbc_url_mart	connection.jdbcUrl
connection_DQF_drivers	connection.driverTable
connection_DQF_driver_class	connection.driverClass
connection_DQF_jdbc_url_raw	
connection_DQF_RPT_additional	
connection_DQF_RPT_database	
connection_DQF_RPT_host	
connection_DQF_RPT_port	
connection_DQF_RPT_schema	
connection_DQF_RPT_warehouse	

DQF_RAW:



VARIABLE	FIELD
connection_DQF_user	connection.userPassword.userId
connection_DQF_password	connection.userPassword.passwo...
connection_DQF_jdbc_url_mart	
connection_DQF_drivers	connection.driverTable
connection_DQF_driver_class	connection.driverClass
connection_DQF_jdbc_url_raw	connection.jdbcUrl
connection_DQF_RPT_additional	
connection_DQF_RPT_database	
connection_DQF_RPT_host	
connection_DQF_RPT_port	
connection_DQF_RPT_schema	
connection_DQF_RPT_warehouse	

6. Open the **DQF_RAW** metadata connection, set the appropriate DB mapping type, and test it. Troubleshoot if it is unsuccessful.



Database Connection

Update Database Connection - Step 2/2

Update connection parameters

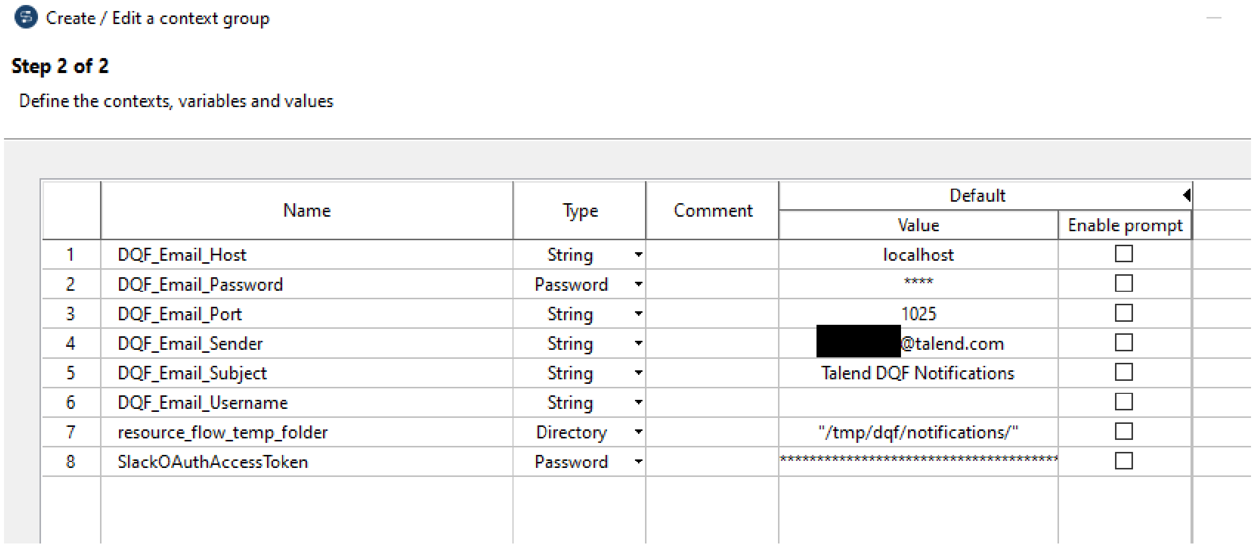
DB Type: JDBC

JDBC URL: context.connection_DQF_jdbc_url_raw

Drivers:
driver
context.connection_DQF_drivers

Connection successful

7. Modify the **DQF_CHANNELS** context group to set the parameters used to send notifications and allow the changes to propagate to all impacted artifacts. The Talend Data Quality Framework currently supports email and Slack as notification channels, but it is fully customizable to allow integration with additional channels. For the purposes of testing the framework, this guide assumes notifications will be sent by email. For that reason, you can leave the **SlackOAuthAccessToken** parameter blank if desired.



Create / Edit a context group

Step 2 of 2

Define the contexts, variables and values

	Name	Type	Comment	Default	
				Value	Enable prompt
1	DQF_Email_Host	String		localhost	□
2	DQF_Email_Password	Password		****	□
3	DQF_Email_Port	String		1025	□
4	DQF_Email_Sender	String		[redacted]@talend.com	□
5	DQF_Email_Subject	String		Talend DQF Notifications	□
6	DQF_Email_Username	String			□
7	resource_flow_temp_folder	Directory		"/tmp/dqf/notifications/"	□
8	SlackOAuthAccessToken	Password		**	□

Importing the demo data and metadata

1. Unzip **DQF demo DDL v1.x.zip**.

   Snowflake:

   Edit the **DQF demo RAW tables and data v1.x - Snowflake.sql** script and set the correct values for the **USE SCHEMA DQF.DQF_RAW** statement. The syntax is:

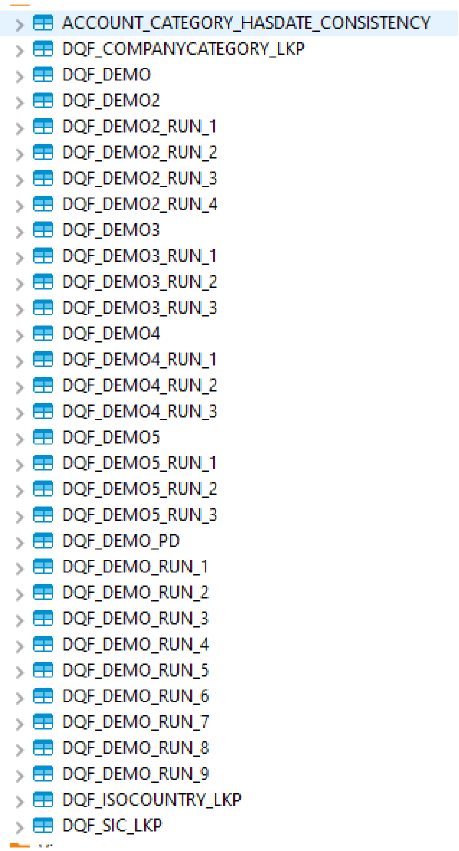
```
USE SCHEMA <DQF Database> .<DQF Raw Schema>
```

   MySQL:

   Edit the **DQF demo RAW tables and data v1.x - MySQL.sql** script and set the correct values for the **USE DQF_RAW** statement. The syntax is:

```
USE <DQF Raw Schema>;
```

2. Execute the script using your SQL client tool and confirm that all statements execute successfully and all tables contain data.



ACCOUNT_CATEGORY_HASDATE_CONSISTENCY
DQF_COMPANYCATEGORY_LKP
DQF_DEMO
DQF_DEMO2
DQF_DEMO2_RUN_1
DQF_DEMO2_RUN_2
DQF_DEMO2_RUN_3
DQF_DEMO2_RUN_4
DQF_DEMO3
DQF_DEMO3_RUN_1
DQF_DEMO3_RUN_2
DQF_DEMO3_RUN_3
DQF_DEMO4
DQF_DEMO4_RUN_1
DQF_DEMO4_RUN_2
DQF_DEMO4_RUN_3
DQF_DEMO5
DQF_DEMO5_RUN_1
DQF_DEMO5_RUN_2
DQF_DEMO5_RUN_3
DQF_DEMO_PD
DQF_DEMO_RUN_1
DQF_DEMO_RUN_2
DQF_DEMO_RUN_3
DQF_DEMO_RUN_4
DQF_DEMO_RUN_5
DQF_DEMO_RUN_6
DQF_DEMO_RUN_7
DQF_DEMO_RUN_8
DQF_DEMO_RUN_9
DQF_ISOCOUNTRY_LKP
DQF_SIC_LKP

Snowflake:

Edit the **DQF demo MART metadata v1.x - Snowflake.sql** script and set the correct values for the **USE SCHEMA DQF.DQF_MART** statement. The syntax is:

```
USE SCHEMA <DQF Database> .<DQF Mart Schema>;
```

MySQL:

Edit the **DQF demo MART metadata v1 - MySQL.x.sql** script and set the correct values for the **USE DQF_MART** statement. The syntax is:

```
USE <DQF Mart Schema>;
```

3. Execute the script using your SQL client tool and confirm that all statements execute successfully and the following tables contain data:

   - **DQF_DATASET**
   - **DQF_DATASET_ATTRIBUTE**
   - **DQF_DQ_RULE**
   - **DQF_DQ_RULE_ATTRIBUTE**
   - **DQF_FAILED_QUERY_TEMPLATE**
   - **DQF_FAILED_QUERY_VARIABLE**
   - **DQF_NOT_CHANNEL**
   - **DQF_NOT_CHANNEL_PERSON**
   - **DQF_NOT_DEFINITION**
   - **DQF_NOT_GROUP**
   - **DQF_NOT_SUBSCRIPTION**
   - **DQF_NOTIFICATION**
   - **DQF_PERSON**
   - **Z_CUSTOMER_DS1** (created by the script, but will contain no data)
   - **Z_CUST_DS2** (created by the script, but will contain no data)
   - **Z_CUSTOMER_DS3** (created by the script, but will contain no data)
   - **Z_KNA1_DS4** (created by the script, but will contain no data)
   - **Z_CUSTREF_DS5** (created by the script, but will contain no data)

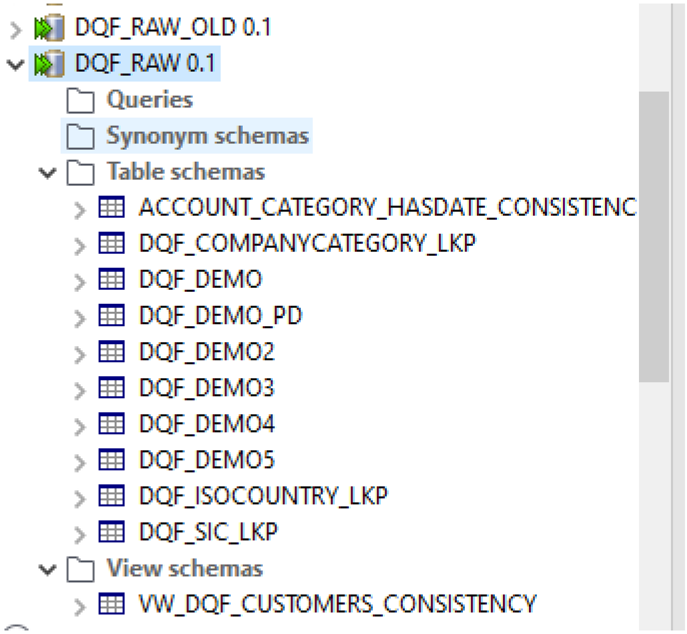
Building the demo analyses

The demo studio analyses included with the framework were created using a generic JDBC connection that was targeted at a Snowflake database called **DQF_DEV** and a schema called **DQF_RAW.** In this release of the framework, the connection cannot be changed to different names on Snowflake or to the schema-only approach used by MySQL without losing the indicator to column mappings of the analysis. This section instructs you on how to simply recreate the demo analyses with your database and database object names of choice, so they can be used to test the framework.

1. In the Data Integration perspective, right click the **DQF_RAW** connection and edit it. Rename the connection *DQF_RAW_OLD* and click **OK**.

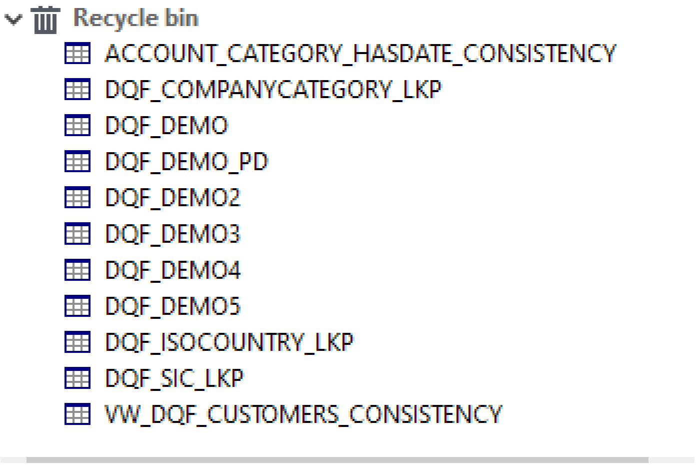
2. Duplicate the **DQF_RAW_OLD** connection and name it *DQF_RAW.*

3. Delete the table and view the schemas under the DQF_RAW connection:



```
> DQF_RAW_OLD 0.1
v DQF_RAW 0.1
    Queries
    Synonym schemas
  v Table schemas
    > ACCOUNT_CATEGORY_HASDATE_CONSISTENC
    > DQF_COMPANYCATEGORY_LKP
    > DQF_DEMO
    > DQF_DEMO_PD
    > DQF_DEMO2
    > DQF_DEMO3
    > DQF_DEMO4
    > DQF_DEMO5
    > DQF_ISOCOUNTRY_LKP
    > DQF_SIC_LKP
  v View schemas
    > VW_DQF_CUSTOMERS_CONSISTENCY
```

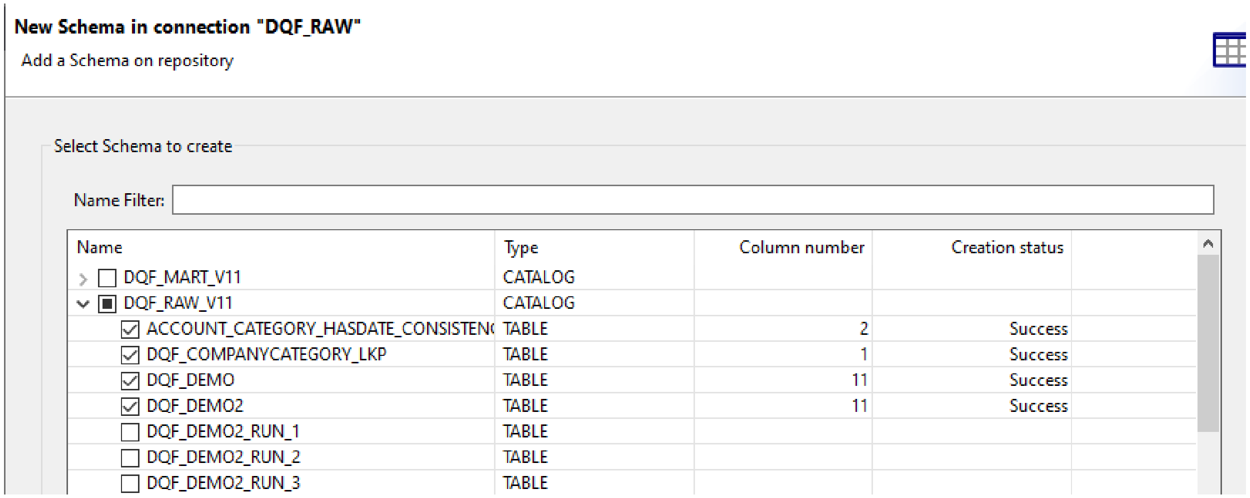
4. Empty the recycle bin.



Recycle bin
- ACCOUNT_CATEGORY_HASDATE_CONSISTENCY
- DQF_COMPANYCATEGORY_LKP
- DQF_DEMO
- DQF_DEMO_PD
- DQF_DEMO2
- DQF_DEMO3
- DQF_DEMO4
- DQF_DEMO5
- DQF_ISOCOUNTRY_LKP
- DQF_SIC_LKP
- VW_DQF_CUSTOMERS_CONSISTENCY

5. Right click the **DQF_RAW** connection and select **Retrieve Schema**.

6. Select the following tables and views and finish the wizard:
   - ACCOUNT_CATEGORY_HASDATE_CONSISTENCY
   - DQF_COMPANYCATEGORY_LKP
   - DQF_DEMO
   - DQF_DEMO_PD
   - DQF_DEMO2
   - DQF_DEMO3
   - DQF_DEMO4
   - DQF_DEMO5
   - DQF_ISOCOUNTRY_LKP
   - DQF_SIC_LKP
   - VW_DQF_CUSTOMERS_CONSISTENCY



New Schema in connection "DQF_RAW"

Add a Schema on repository

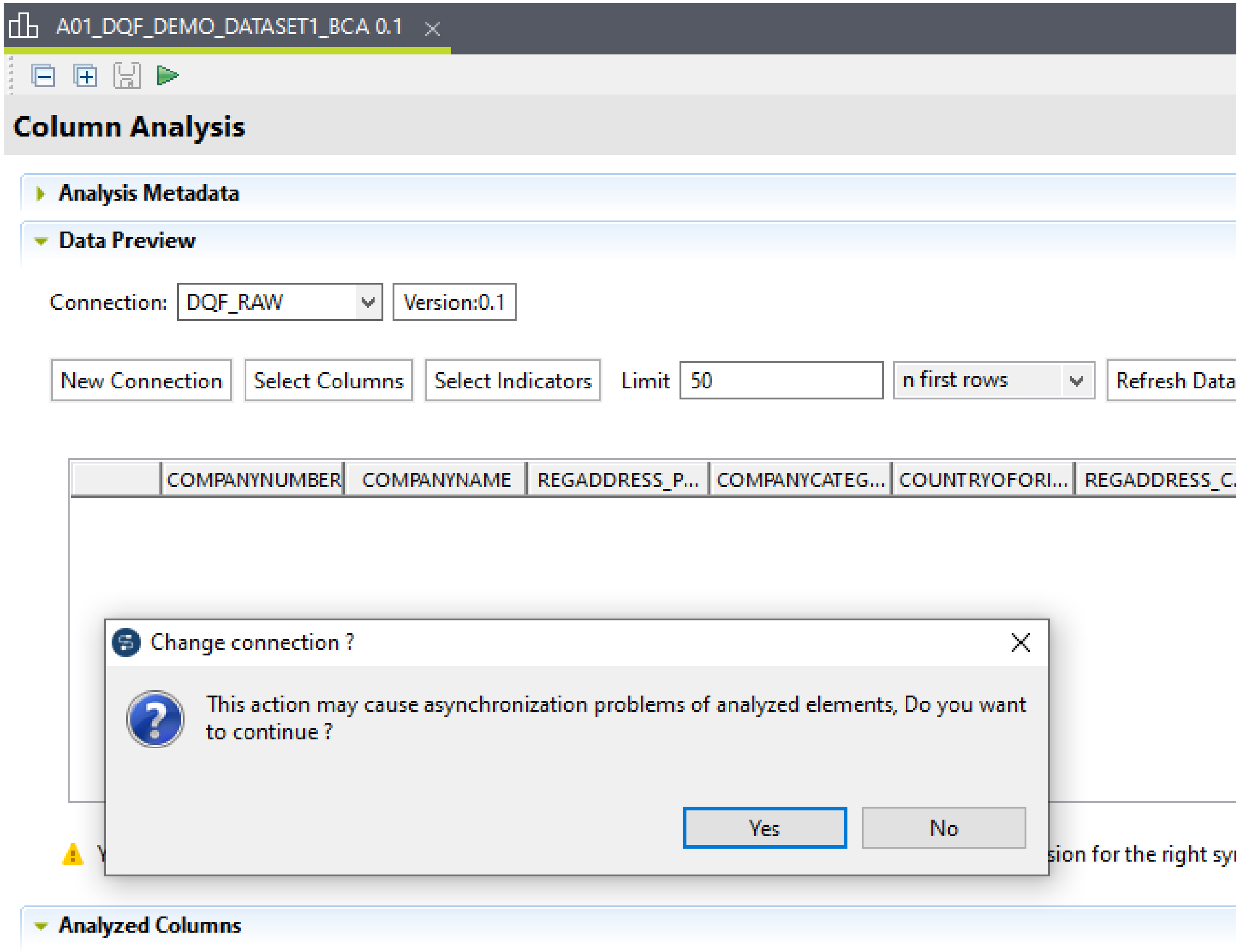
Select Schema to create

Name Filter:

Name	Type	Column number	Creation status
› ☐ DQF_MART_V11	CATALOG		
⌄ ■ DQF_RAW_V11	CATALOG		
☑ ACCOUNT_CATEGORY_HASDATE_CONSISTEN(	TABLE	2	Success
☑ DQF_COMPANYCATEGORY_LKP	TABLE	1	Success
☑ DQF_DEMO	TABLE	11	Success
☑ DQF_DEMO2	TABLE	11	Success
☐ DQF_DEMO2_RUN_1	TABLE		
☐ DQF_DEMO2_RUN_2	TABLE		
☐ DQF_DEMO2_RUN_3	TABLE		

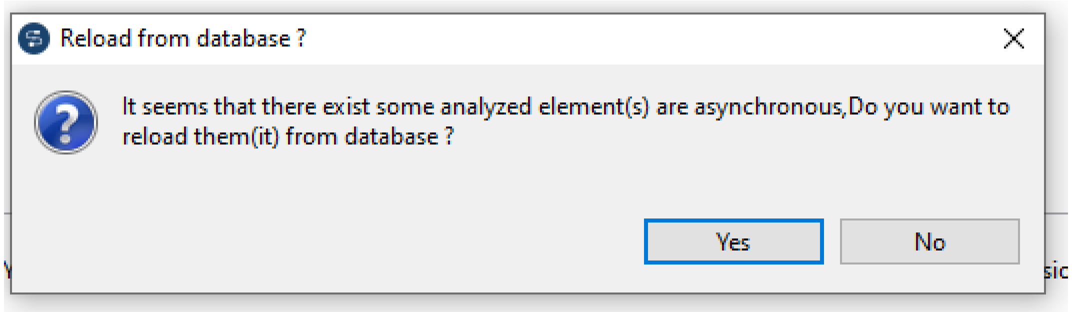
7. Switch to the **Profiling** perspective and open the **A01_DQF_DEMO_DATASET1_BCA** analysis.

8. Change the connection from **DQF_RAW_OLD** to **DQF_RAW**. Select **Yes** or **OK** when prompted:



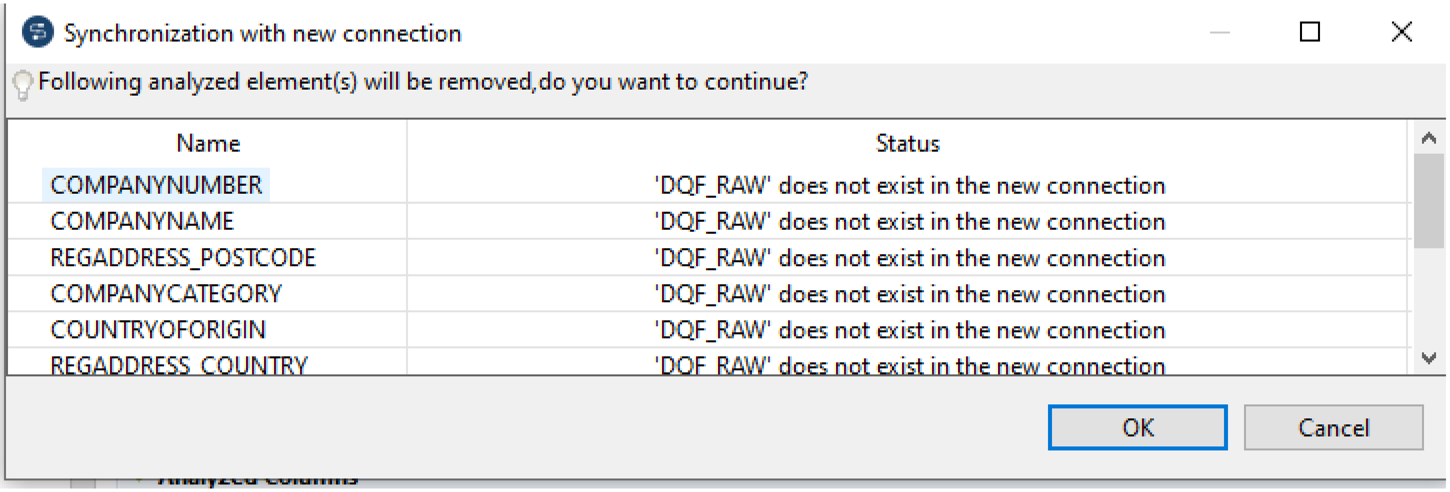
Change connection ?

This action may cause asynchronization problems of analyzed elements, Do you want to continue ?



Reload from database ?

It seems that there exist some analyzed element(s) are asynchronous,Do you want to reload them(it) from database ?

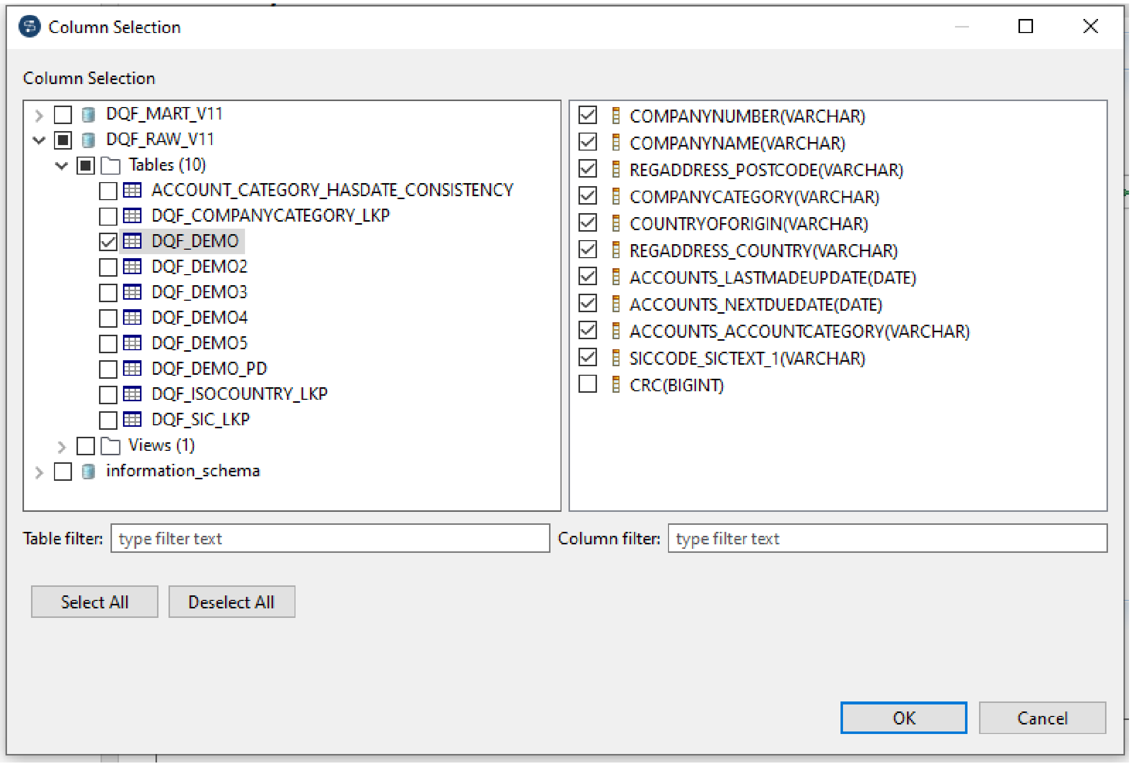


Synchronization with new connection

Following analyzed element(s) will be removed,do you want to continue?

Name	Status
COMPANYNUMBER	'DQF_RAW' does not exist in the new connection
COMPANYNAME	'DQF_RAW' does not exist in the new connection
REGADDRESS_POSTCODE	'DQF_RAW' does not exist in the new connection
COMPANYCATEGORY	'DQF_RAW' does not exist in the new connection
COUNTRYOFORIGIN	'DQF_RAW' does not exist in the new connection
REGADDRESS_COUNTRY	'DQF_RAW' does not exist in the new connection

9. Click **Select Columns** and select all the fields except CRC:



Column Selection

Column Selection

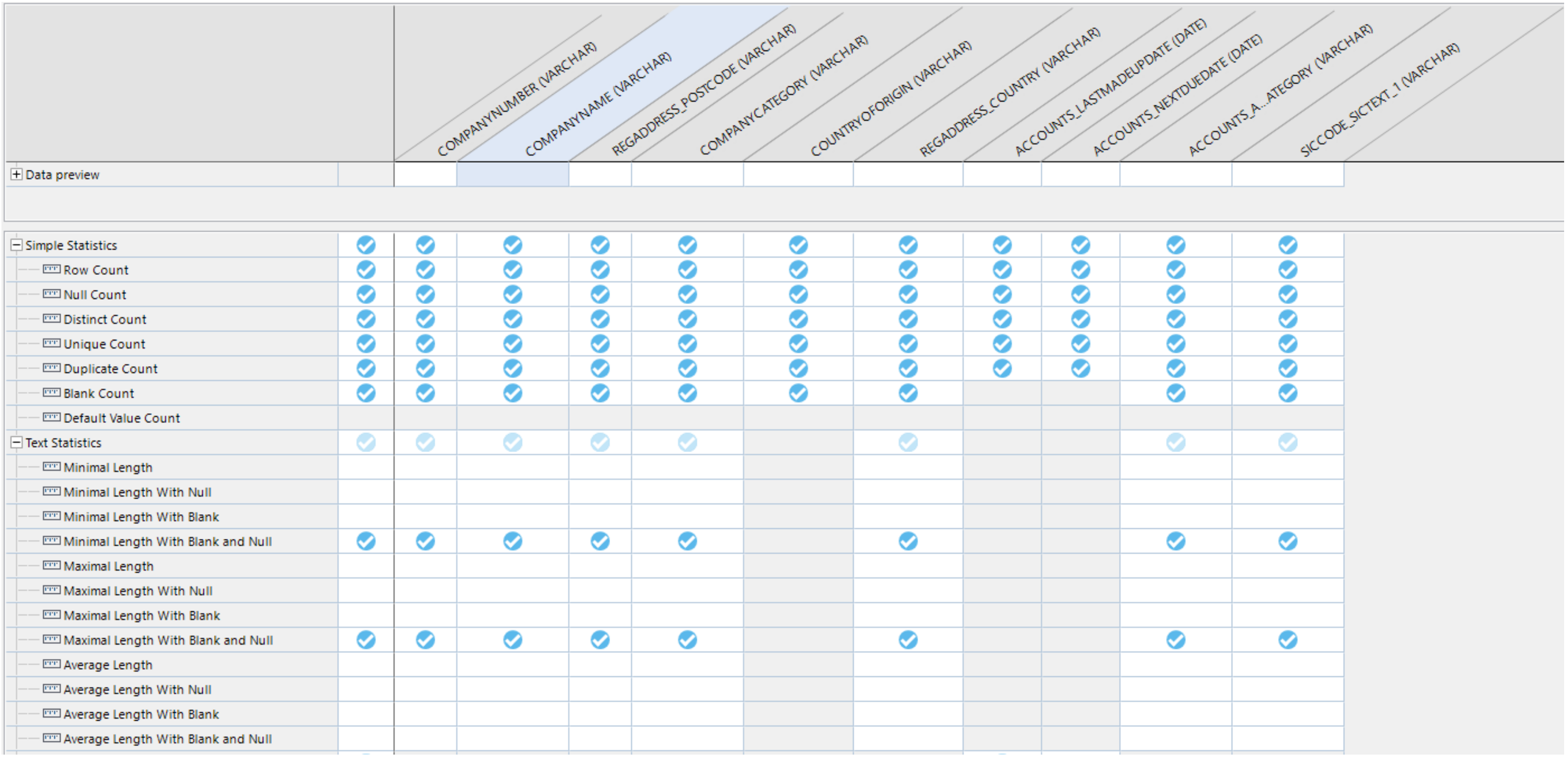
- □ DQF_MART_V11
- ■ DQF_RAW_V11
  - ■ Tables (10)
    - □ ACCOUNT_CATEGORY_HASDATE_CONSISTENCY
    - □ DQF_COMPANYCATEGORY_LKP
    - ☑ DQF_DEMO
    - □ DQF_DEMO2
    - □ DQF_DEMO3
    - □ DQF_DEMO4
    - □ DQF_DEMO5
    - □ DQF_DEMO_PD
    - □ DQF_ISOCOUNTRY_LKP
    - □ DQF_SIC_LKP
  - □ Views (1)
- □ information_schema

- ☑ COMPANYNUMBER(VARCHAR)
- ☑ COMPANYNAME(VARCHAR)
- ☑ REGADDRESS_POSTCODE(VARCHAR)
- ☑ COMPANYCATEGORY(VARCHAR)
- ☑ COUNTRYOFORIGIN(VARCHAR)
- ☑ REGADDRESS_COUNTRY(VARCHAR)
- ☑ ACCOUNTS_LASTMADEUPDATE(DATE)
- ☑ ACCOUNTS_NEXTDUEDATE(DATE)
- ☑ ACCOUNTS_ACCOUNTCATEGORY(VARCHAR)
- ☑ SICCODE_SICTEXT_1(VARCHAR)
- □ CRC(BIGINT)

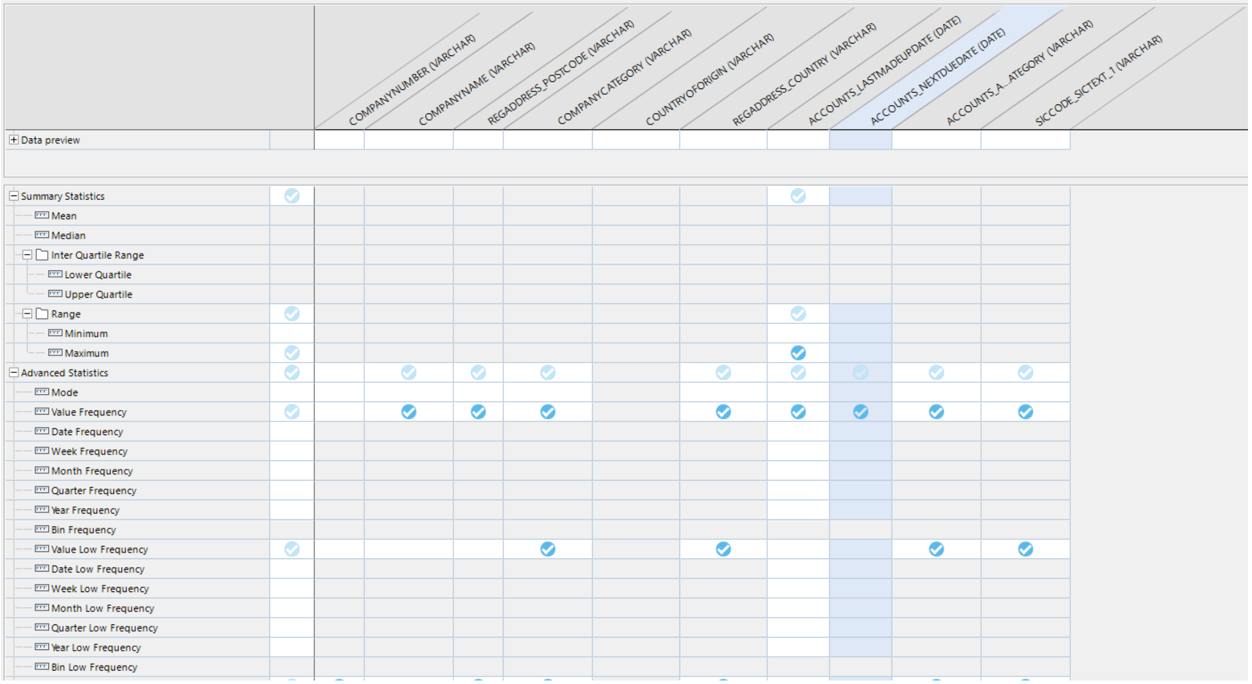
Table filter: type filter text

Column filter: type filter text

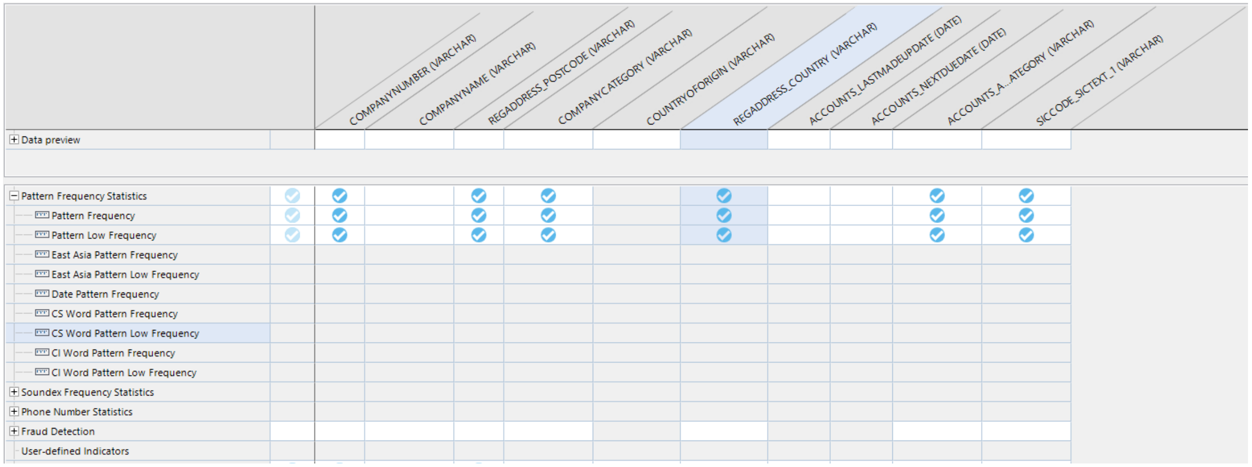
10. Configure the analysis as follows:



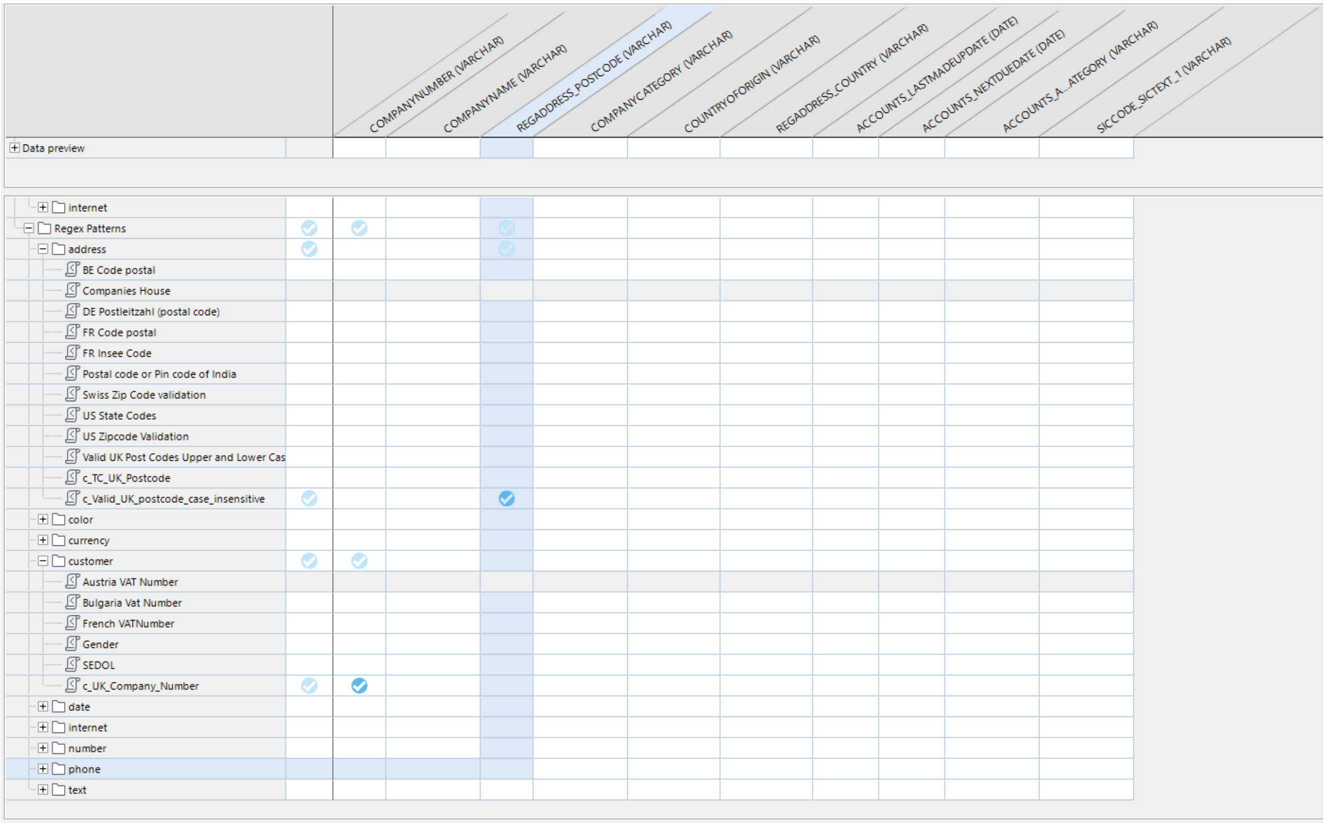
		COMPANYNUMBER (VARCHAR)	COMPANYNAME (VARCHAR)	REGADDRESS_POSTCODE (VARCHAR)	COMPANYCATEGORY (VARCHAR)	COUNTRYOFORIGIN (VARCHAR)	REGADDRESS_COUNTRY (VARCHAR)	ACCOUNTS_LASTMADEUPDATE (DATE)	ACCOUNTS_NEXTDUEDATE (DATE)	ACCOUNTS_A...ATEGORY (VARCHAR)	SICCODE_SICTEXT_1 (VARCHAR)
⊞ Data preview											
⊟ Simple Statistics	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Row Count	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Null Count	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Distinct Count	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Unique Count	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Duplicate Count	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Blank Count	✓	✓	✓	✓	✓	✓	✓			✓	✓
Default Value Count											
⊟ Text Statistics	✓	✓	✓	✓	✓		✓			✓	✓
Minimal Length											
Minimal Length With Null											
Minimal Length With Blank											
Minimal Length With Blank and Null	✓	✓	✓	✓	✓		✓			✓	✓
Maximal Length											
Maximal Length With Null											
Maximal Length With Blank											
Maximal Length With Blank and Null	✓	✓	✓	✓	✓		✓			✓	✓
Average Length											
Average Length With Null											
Average Length With Blank											
Average Length With Blank and Null											



		COMPANYNUMBER (VARCHAR)	COMPANYNAME (VARCHAR)	REGADDRESS_POSTCODE (VARCHAR)	COMPANYCATEGORY (VARCHAR)	COUNTRYOFORIGIN (VARCHAR)	REGADDRESS_COUNTRY (VARCHAR)	ACCOUNTS_LASTMADEUPDATE (DATE)	ACCOUNTS_NEXTDUEDATE (DATE)	ACCOUNTS_A...ATEGORY (VARCHAR)	SICCODE_SICTEXT_1 (VARCHAR)
⊞ Data preview											
⊟ Summary Statistics	✓							✓			
Mean											
Median											
⊟ Inter Quartile Range											
Lower Quartile											
Upper Quartile											
⊟ Range	✓							✓			
Minimum											
Maximum	✓							✓			
⊟ Advanced Statistics	✓		✓	✓	✓		✓	✓	✓	✓	✓
Mode											
Value Frequency	✓		✓	✓	✓		✓	✓	✓	✓	✓
Date Frequency											
Week Frequency											
Month Frequency											
Quarter Frequency											
Year Frequency											
Bin Frequency											
Value Low Frequency	✓				✓		✓			✓	✓
Date Low Frequency											
Week Low Frequency											
Month Low Frequency											
Quarter Low Frequency											
Year Low Frequency											
Bin Low Frequency											



		COMPANYNUMBER (VARCHAR)	COMPANYNAME (VARCHAR)	REGADDRESS_POSTCODE (VARCHAR)	COMPANYCATEGORY (VARCHAR)	COUNTRYOFORIGIN (VARCHAR)	REGADDRESS_COUNTRY (VARCHAR)	ACCOUNTS_LASTMADEUPDATE (DATE)	ACCOUNTS_NEXTDUEDATE (DATE)	ACCOUNTS_A...ATEGORY (VARCHAR)	SICCODE_SICTEXT_1 (VARCHAR)
⊞ Data preview											
⊟ Pattern Frequency Statistics	✓	✓		✓	✓		✓			✓	✓
Pattern Frequency	✓	✓		✓	✓		✓			✓	✓
Pattern Low Frequency	✓	✓		✓	✓		✓			✓	✓
East Asia Pattern Frequency											
East Asia Pattern Low Frequency											
Date Pattern Frequency											
CS Word Pattern Frequency											
CS Word Pattern Low Frequency											
CI Word Pattern Frequency											
CI Word Pattern Low Frequency											
⊞ Soundex Frequency Statistics											
⊞ Phone Number Statistics											
⊞ Fraud Detection											
User-defined Indicators											

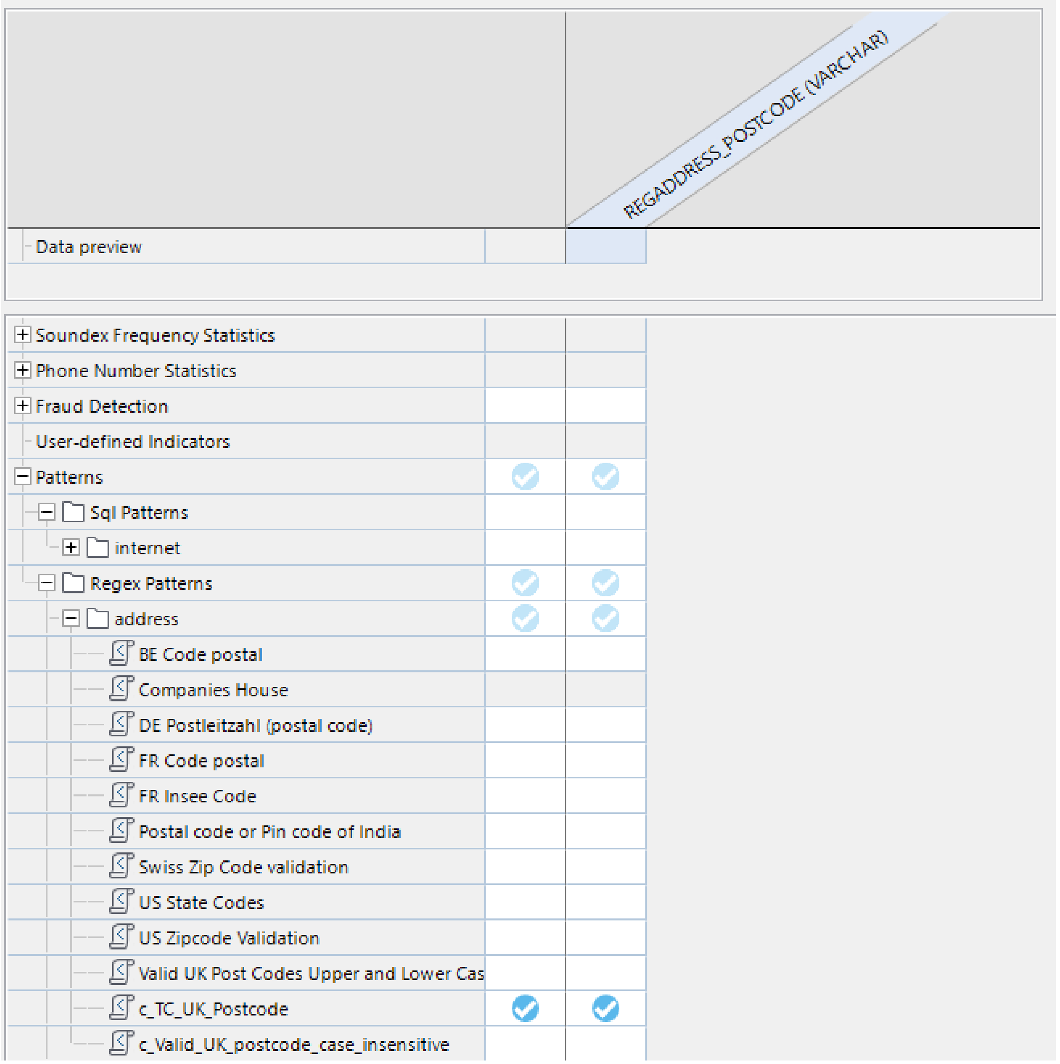


		COMPANYNUMBER (VARCHAR)	COMPANYNAME (VARCHAR)	REGADDRESS_POSTCODE (VARCHAR)	COMPANYCATEGORY (VARCHAR)	COUNTRYOFORIGIN (VARCHAR)	REGADDRESS_COUNTRY (VARCHAR)	ACCOUNTS_LASTMADEUPDATE (DATE)	ACCOUNTS_NEXTDUEDATE (DATE)	ACCOUNTS_A...ATEGORY (VARCHAR)	SICCODE_SICTEXT_1 (VARCHAR)
⊞ Data preview											
⊞ internet											
⊟ Regex Patterns	✓	✓		✓							
⊟ address	✓			✓							
BE Code postal											
Companies House											
DE Postleitzahl (postal code)											
FR Code postal											
FR Insee Code											
Postal code or Pin code of India											
Swiss Zip Code validation											
US State Codes											
US Zipcode Validation											
Valid UK Post Codes Upper and Lower Cas											
c_TC_UK_Postcode											
c_Valid_UK_postcode_case_insensitive	✓			✓							
⊞ color											
⊞ currency											
⊟ customer	✓	✓									
Austria VAT Number											
Bulgaria Vat Number											
French VATNumber											
Gender											
SEDOL											
c_UK_Company_Number	✓	✓									
⊞ date											
⊞ internet											
⊞ number											
⊞ phone											
⊞ text											

11. Perform a test run of the analysis and ensure it executes correctly.

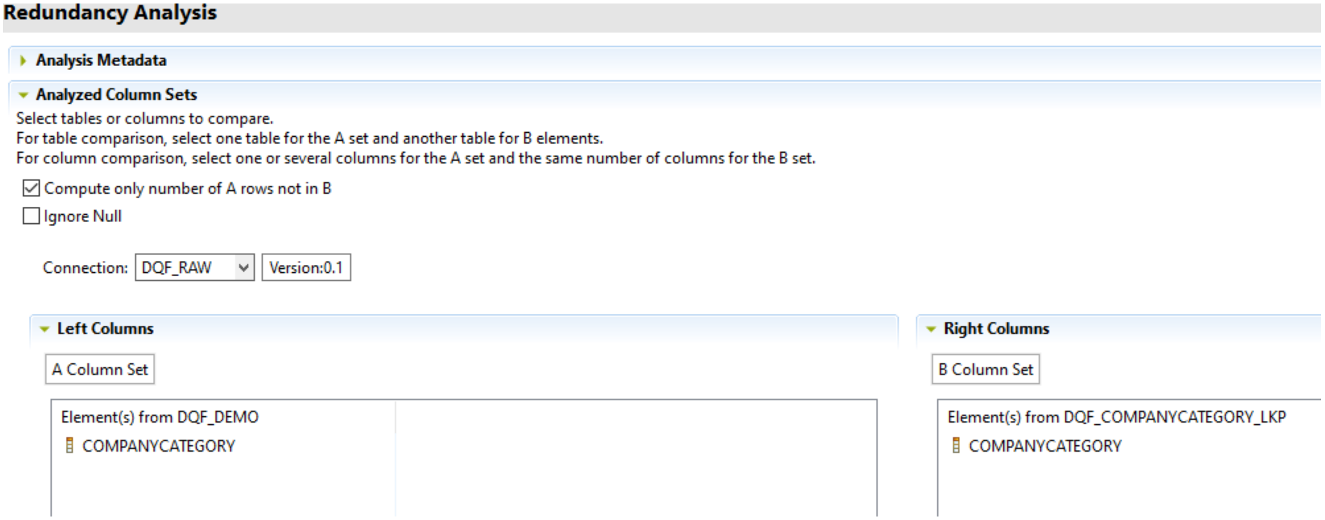
12. Change the connection from **DQF_RAW_OLD** to **DQF_RAW** for each of the other provided analyses and configure them as follows:

A02 – Table: DQF_DEMO



		REGADDRESS_POSTCODE (VARCHAR)
Data preview		
Soundex Frequency Statistics		
Phone Number Statistics		
Fraud Detection		
User-defined Indicators		
Patterns	✓	✓
Sql Patterns		
internet		
Regex Patterns	✓	✓
address	✓	✓
BE Code postal		
Companies House		
DE Postleitzahl (postal code)		
FR Code postal		
FR Insee Code		
Postal code or Pin code of India		
Swiss Zip Code validation		
US State Codes		
US Zipcode Validation		
Valid UK Post Codes Upper and Lower Cas		
c_TC_UK_Postcode	✓	✓
c_Valid_UK_postcode_case_insensitive		

A03 – Tables as shown


Redundancy Analysis

Select tables or columns to compare.
For table comparison, select one table for the A set and another table for B elements.
For column comparison, select one or several columns for the A set and the same number of columns for the B set.

☑ Compute only number of A rows not in B

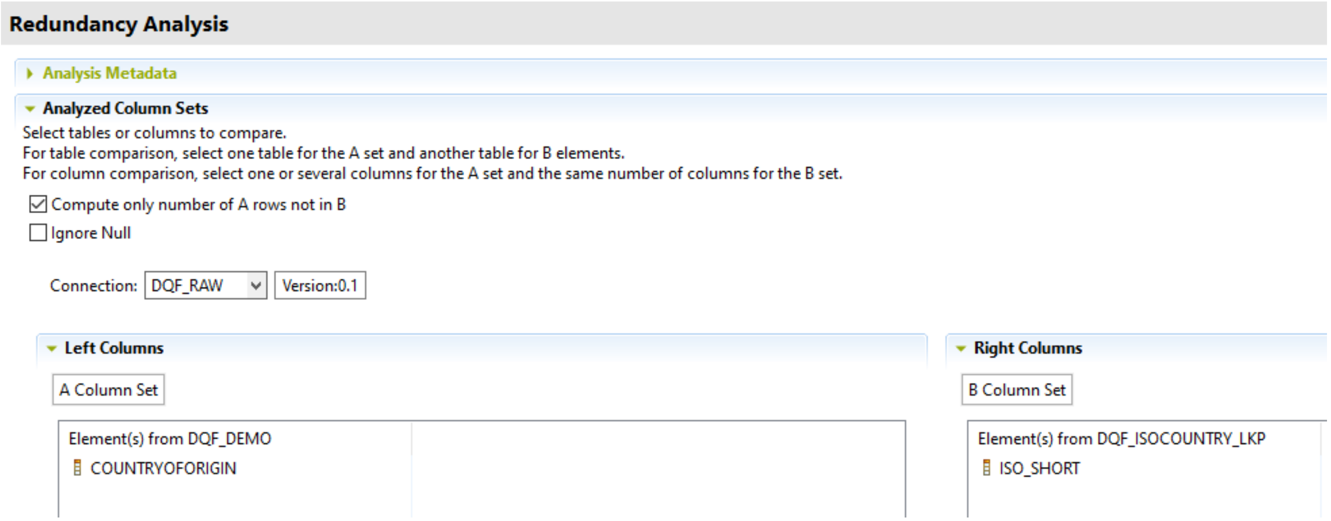
□ Ignore Null

Connection: DQF_RAW Version:0.1

Element(s) from DQF_DEMO
COMPANYCATEGORY

Element(s) from DQF_COMPANYCATEGORY_LKP
COMPANYCATEGORY

A04 – Tables as shown


Redundancy Analysis

Select tables or columns to compare.
For table comparison, select one table for the A set and another table for B elements.
For column comparison, select one or several columns for the A set and the same number of columns for the B set.

☑ Compute only number of A rows not in B

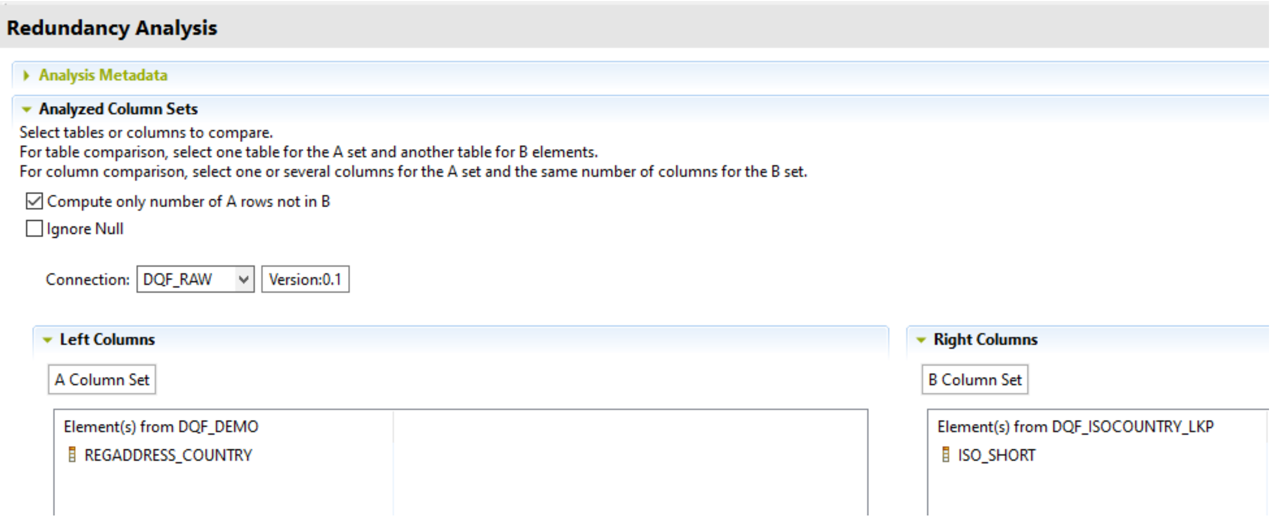
□ Ignore Null

Connection: DQF_RAW Version:0.1

Element(s) from DQF_DEMO
COUNTRYOFORIGIN

Element(s) from DQF_ISOCOUNTRY_LKP
ISO_SHORT

A05 – Tables as shown



Redundancy Analysis

Analysis Metadata

Analyzed Column Sets

Select tables or columns to compare.
For table comparison, select one table for the A set and another table for B elements.
For column comparison, select one or several columns for the A set and the same number of columns for the B set.

☑ Compute only number of A rows not in B

□ Ignore Null

Connection: DQF_RAW Version:0.1

Left Columns

A Column Set

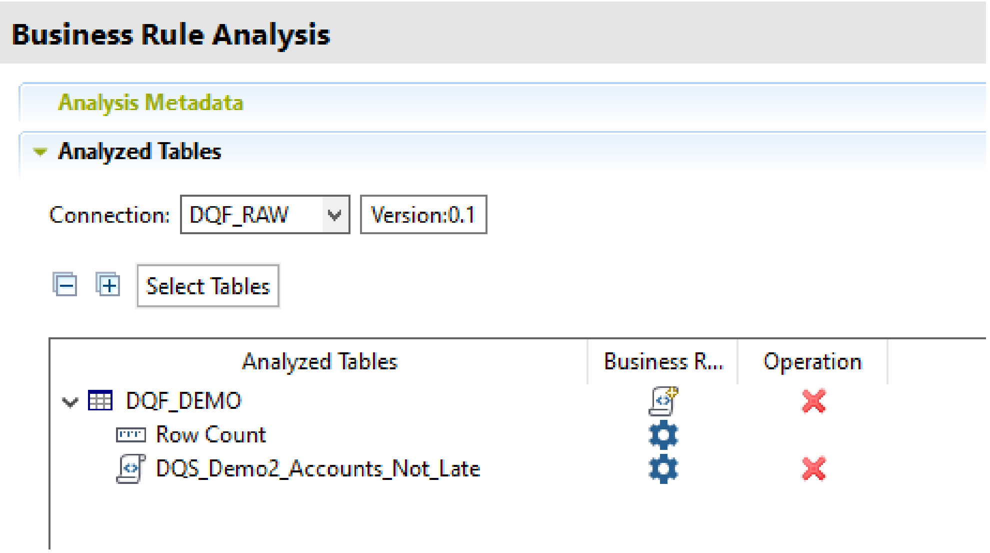
Element(s) from DQF_DEMO
REGADDRESS_COUNTRY

Right Columns

B Column Set

Element(s) from DQF_ISOCOUNTRY_LKP
ISO_SHORT

A06 – Table as shown



Business Rule Analysis

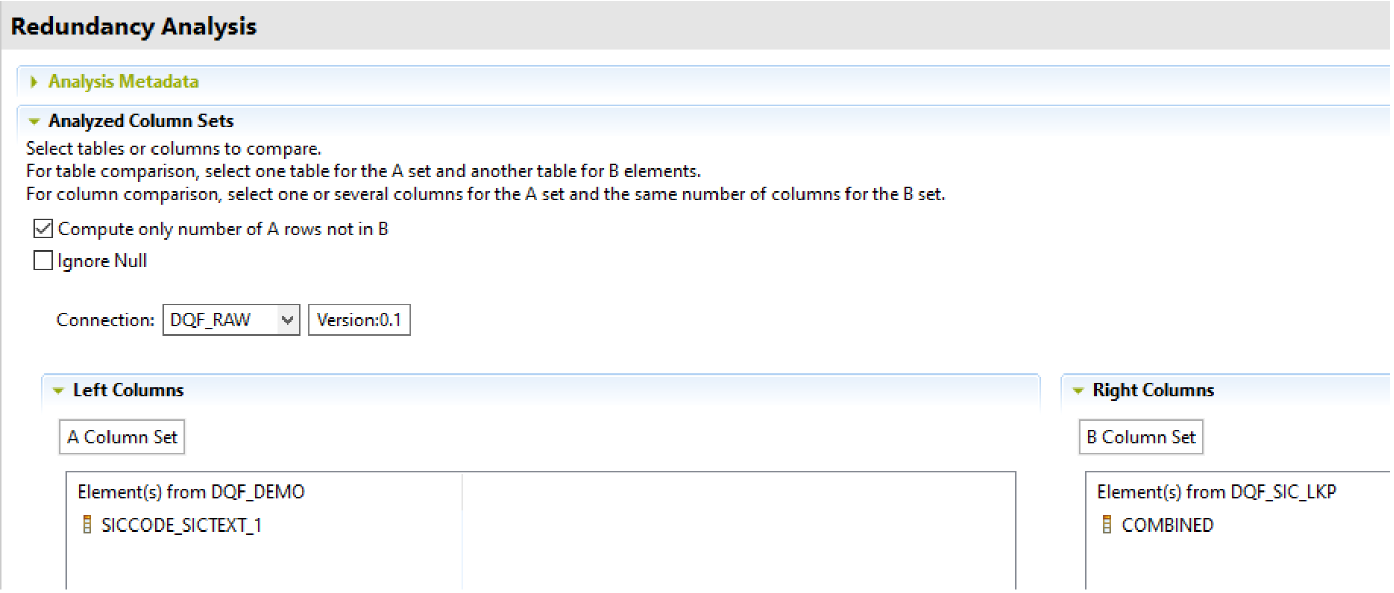
Analysis Metadata

Analyzed Tables

Connection: DQF_RAW Version:0.1

Analyzed Tables	Business R...	Operation
DQF_DEMO		✗
Row Count		
DQS_Demo2_Accounts_Not_Late		✗

A07 – Tables as shown



Redundancy Analysis

Analyzed Column Sets

Select tables or columns to compare.
For table comparison, select one table for the A set and another table for B elements.
For column comparison, select one or several columns for the A set and the same number of columns for the B set.

☑ Compute only number of A rows not in B

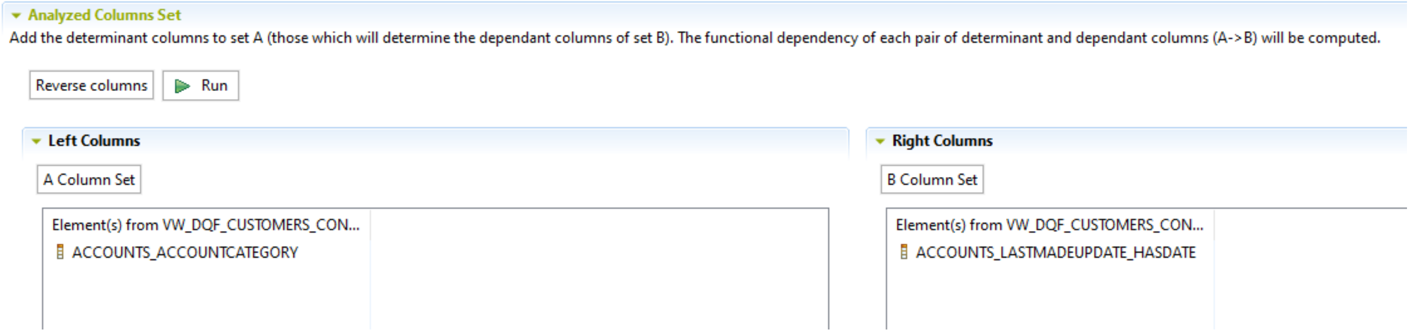
☐ Ignore Null

Connection: DQF_RAW Version:0.1

Element(s) from DQF_DEMO
SICCODE_SICTEXT_1

Element(s) from DQF_SIC_LKP
COMBINED

A08 – View as shown

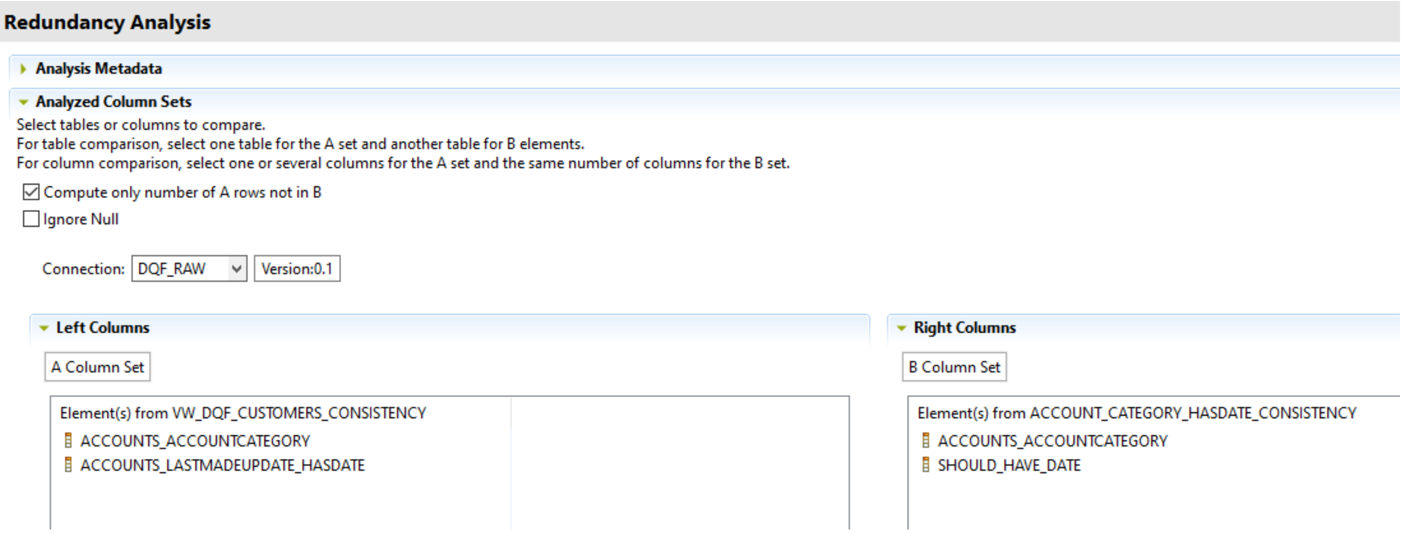


Add the determinant columns to set A (those which will determine the dependant columns of set B). The functional dependency of each pair of determinant and dependant columns (A->B) will be computed.

Element(s) from VW_DQF_CUSTOMERS_CON...
ACCOUNTS_ACCOUNTCATEGORY

Element(s) from VW_DQF_CUSTOMERS_CON...
ACCOUNTS_LASTMADEUPDATE_HASDATE

A09 – Tables as shown



Redundancy Analysis

- Analysis Metadata
- Analyzed Column Sets

Select tables or columns to compare.
For table comparison, select one table for the A set and another table for B elements.
For column comparison, select one or several columns for the A set and the same number of columns for the B set.

☑ Compute only number of A rows not in B

□ Ignore Null

Connection: DQF_RAW Version:0.1

Left Columns

A Column Set

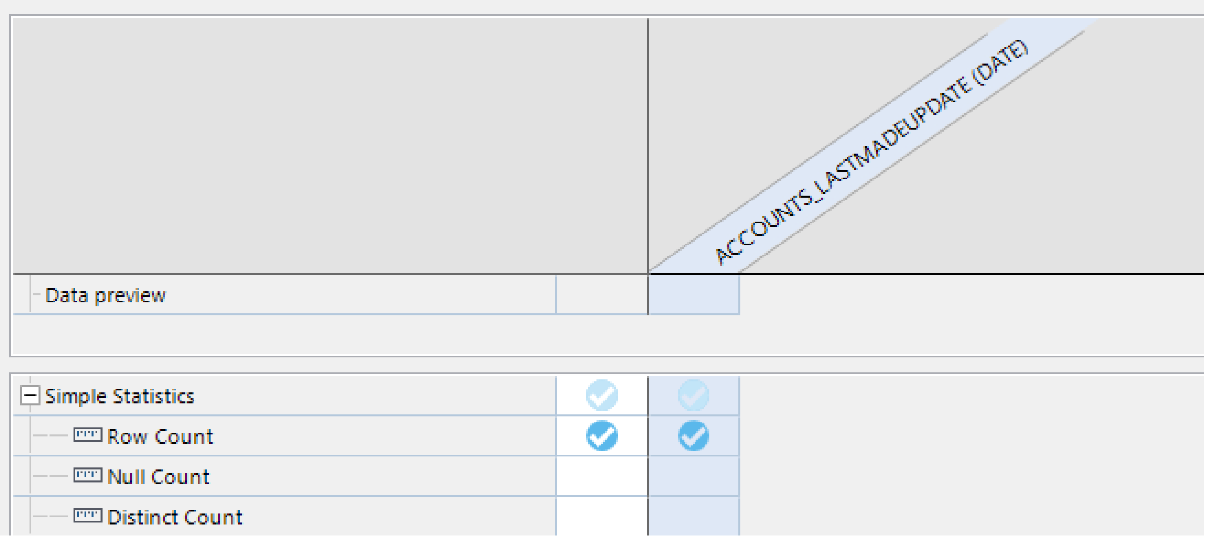
Element(s) from VW_DQF_CUSTOMERS_CONSISTENCY
- ACCOUNTS_ACCOUNTCATEGORY
- ACCOUNTS_LASTMADEUPDATE_HASDATE

Right Columns

B Column Set

Element(s) from ACCOUNT_CATEGORY_HASDATE_CONSISTENCY
- ACCOUNTS_ACCOUNTCATEGORY
- SHOULD_HAVE_DATE

A10 – Table: DQF Demo



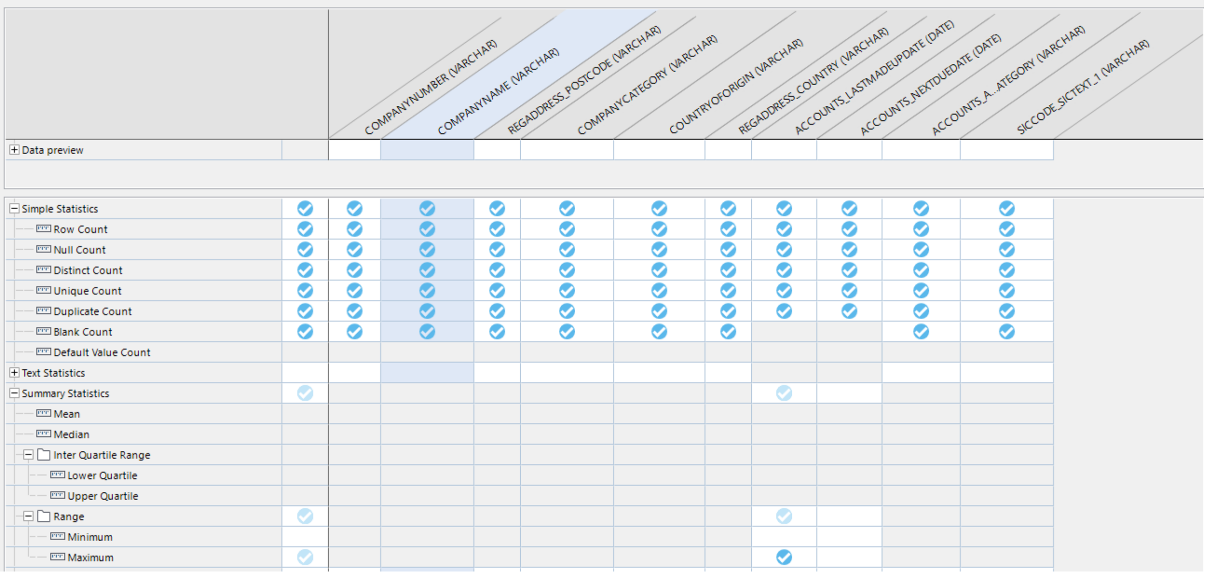
		ACCOUNTS_LASTMADEUPDATE (DATE)
Data preview		
Simple Statistics	✓	✓
Row Count	✓	✓
Null Count		
Distinct Count		

For Snowflake, leave the **where** clause of the analysis unchanged.

MySQL where clause:

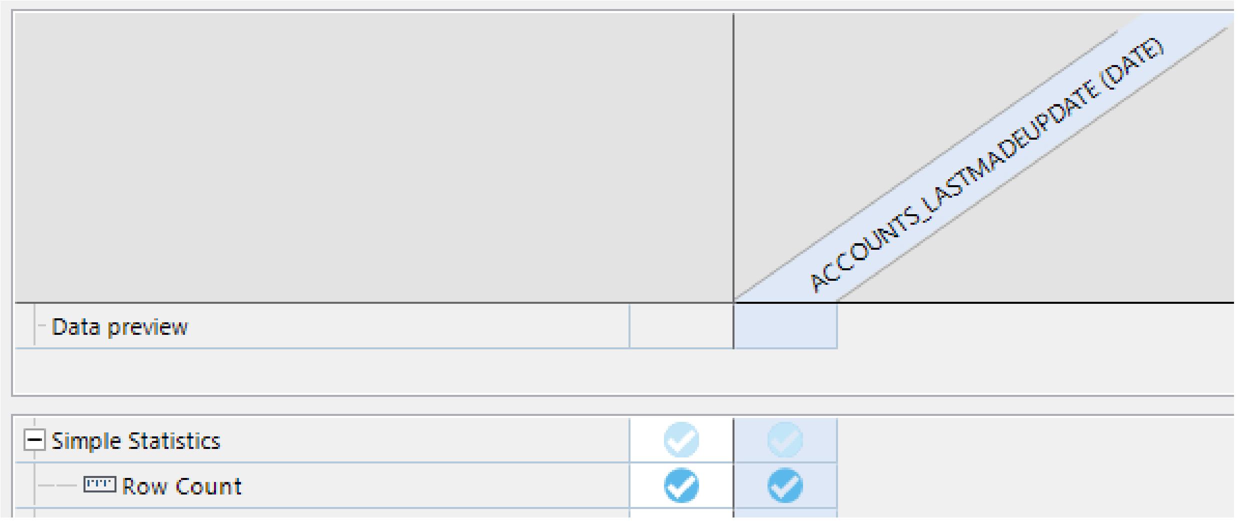
```
datediff(ACCOUNTS_LASTMADEUPDATE, date_add(SYSDATE(), INTERVAL -2 MONTH)) > 0
```

B01 (DQF_DEMO2), C01 (DQF_DEMO3), D01 (DQF_DEMO4), E01 (DQF_DEMO5)



		COMPANYNUMBER (VARCHAR)	COMPANYNAME (VARCHAR)	REGADDRESS_POSTCODE (VARCHAR)	COMPANYCATEGORY (VARCHAR)	COUNTRYOFORIGIN (VARCHAR)	REGADDRESS_COUNTRY (VARCHAR)	ACCOUNTS_LASTMADEUPDATE (DATE)	ACCOUNTS_NEXTDUEDATE (DATE)	ACCOUNTS_A...ATEGORY (VARCHAR)	SICCODE_SICTEXT_1 (VARCHAR)
⊞ Data preview											
⊟ Simple Statistics	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Row Count	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Null Count	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Distinct Count	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Unique Count	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Duplicate Count	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Blank Count	✓	✓	✓	✓	✓	✓	✓			✓	✓
Default Value Count											
⊞ Text Statistics											
⊟ Summary Statistics	✓							✓			
Mean											
Median											
⊟ Inter Quartile Range											
Lower Quartile											
Upper Quartile											
⊟ Range	✓							✓			
Minimum											
Maximum	✓							✓			

B10 (DQF_DEMO2), C10 (DQF_DEMO3), D10 (DQF_DEMO4), E10 (DQF_DEMO5)

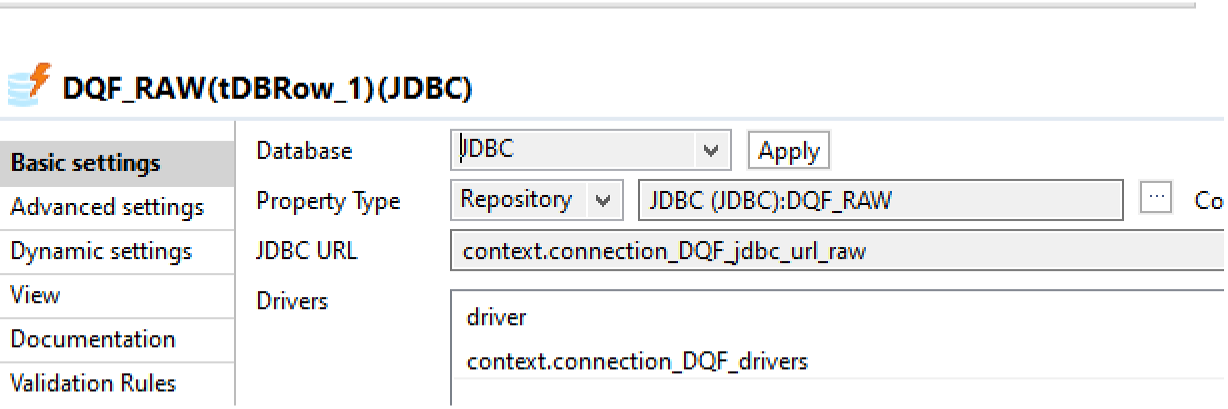


For Snowflake, leave the **where** clause of the analysis unchanged.

MySQL where clause:

```
datediff(ACCOUNTS_LASTMADEUPDATE, date_add(SYSDATE(), INTERVAL -2 MONTH)) > 0
```

13. Switch to the Data Integration perspective and open the **DQF_Template_Jobs\A00_Truncate_Pipeline_Designer_Target_Table** Job. Change the connection from **DQF_RAW_OLD** to **DQF_RAW** and click **Save** or **Close**.



DQF_RAW(tDBRow_1)(JDBC)

Database: JDBC

Property Type: Repository — JDBC (JDBC):DQF_RAW

JDBC URL: context.connection_DQF_jdbc_url_raw

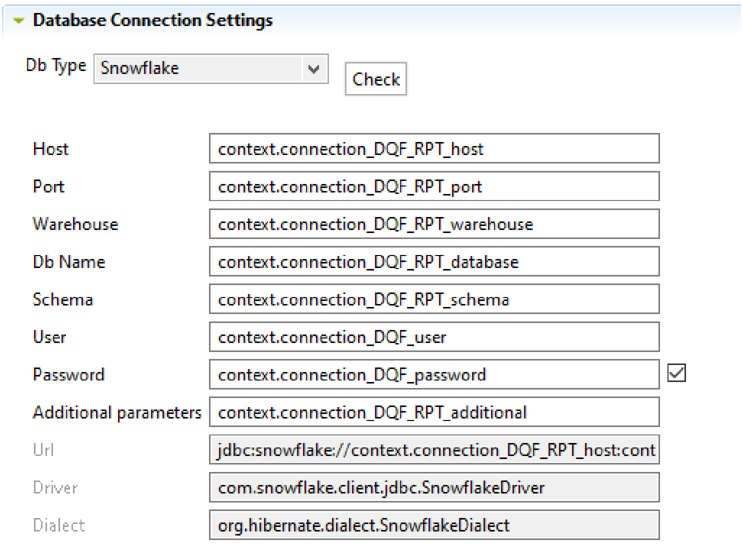
Drivers:

driver
context.connection_DQF_drivers

14. Perform an Impact Analysis on the **DQF_RAW_OLD** connection. There should be zero results, but do some troubleshooting if not.

15. Delete the **DQF_RAW_OLD** connection and empty the recycle bin.

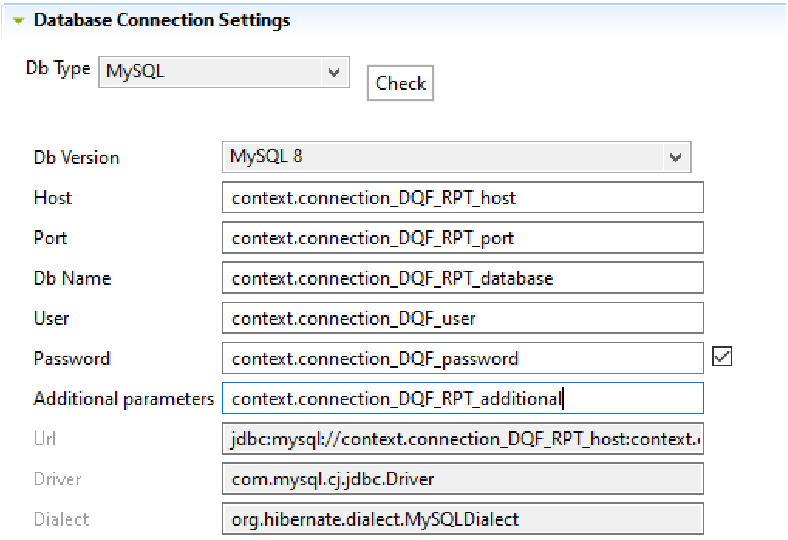
16. **MySQL Only**: In the Profiling perspective, open the **DQF_DEMO_DATASET1_RPT** report. Change the report database settings from Snowflake to MySQL as follows:



Database Connection Settings

Db Type: Snowflake · Check

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 ☑
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



Database Connection Settings

Db Type: MySQL · Check

Db Version	MySQL 8
Host	context.connection_DQF_RPT_host
Port	context.connection_DQF_RPT_port
Db Name	context.connection_DQF_RPT_database
User	context.connection_DQF_user
Password	context.connection_DQF_password ☑
Additional parameters	context.connection_DQF_RPT_additional
Url	jdbc:mysql://context.connection_DQF_RPT_host:context.
Driver	com.mysql.cj.jdbc.Driver
Dialect	org.hibernate.dialect.MySQLDialect

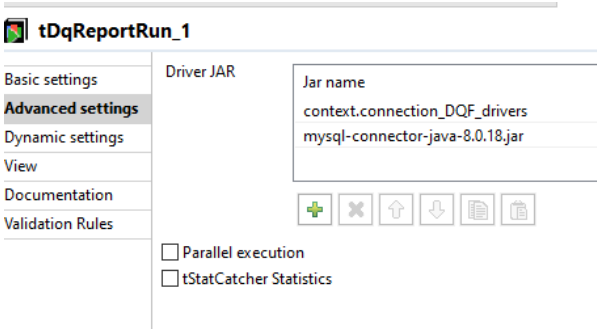
17. **MySQL Only**: Repeat step 16 for the other four reports.

18. In the Data Integration perspective, open the **DQF_Demo_Jobs/ A20_Run_Dataset1_RPT** Job.

19. Click the **tDQReportRun** component and click **Browse Reports**. Select the **DQF_DEMO_DATASET1_RPT** report then click **OK**.

20. Set the Output folder to *"/tmp"*.

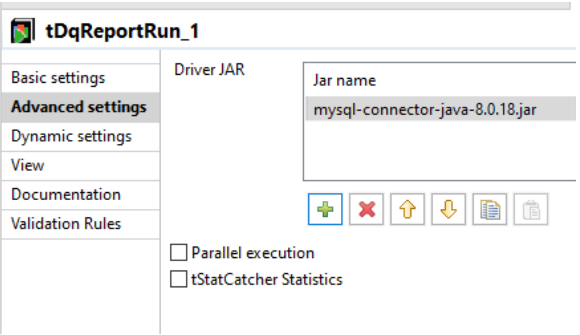
21. Switch to the Advanced settings tab and remove the **context.connection_DQF_drivers** entry from the Driver JAR table:



Driver JAR

Jar name
context.connection_DQF_drivers
mysql-connector-java-8.0.18.jar

□ Parallel execution
□ tStatCatcher Statistics



Driver JAR

Jar name
mysql-connector-java-8.0.18.jar

□ Parallel execution
□ tStatCatcher Statistics

22. Save the Job and close it.

23. Repeat steps 18-22 for the other report Jobs:

Job Name	Report Name
B20_Run_Dataset2_RPT	DQF_DEMO_DATASET2_RPT
C20_Run_Dataset3_RPT	DQF_DEMO_DATASET3_RPT
D20_Run_Dataset4_RPT	DQF_DEMO_DATASET4_RPT
E20_Run_Dataset5_RPT	DQF_DEMO_DATASET5_RPT

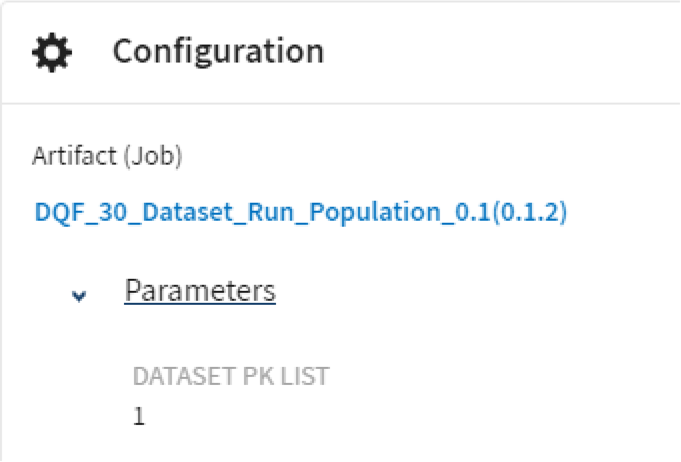
Using the demo to perform an end-to-end test of the framework

In this section, you perform a manual initial test of the framework with a single dataset. Subsequent steps perform an automated test of the framework with multiple datasets.

1. In TMC, execute the **DQF_10_Initialise_DQF_Run** task. This should write a row to the **DQF_RUN** table in the DQF Mart Schema.

2. In Studio, execute the **DQF_Demo_Jobs\A20_Run_Dataset1_RPT** Job.

3. In TMC, open the **DQF_30_Dataset_Run_Population** task and set the **DATASET_PK_LIST** parameter to the value **1**.



Configuration

Artifact (Job)

DQF_30_Dataset_Run_Population_0.1(0.1.2)

Parameters

DATASET PK LIST

1

4. In TMC, execute the **DQF_30_Dataset_Run_Population** task. This should write a row to the **DQF_DATASET_RUN** table in the DQF Mart Schema.

5. In TMC, execute the **DQF_40_Failed_Rows_ELT** task. This should write numerous rows to the **DQF_FAILED_ROW_TMP** and **DQF_FAILED_ROW** tables in the DQF Mart Schema.

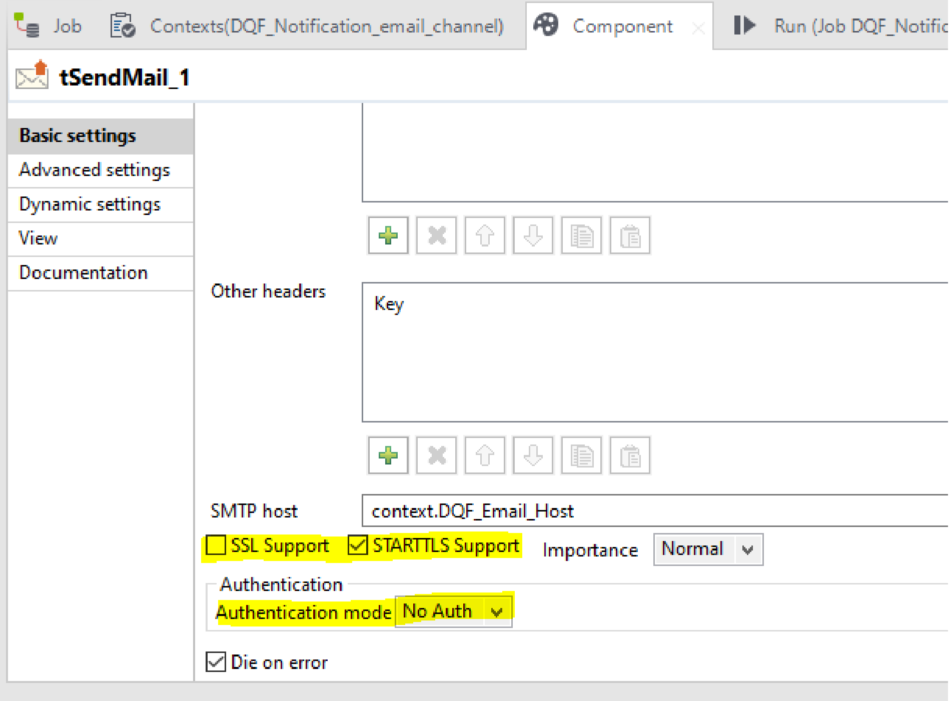
6. In TMC, execute the **DQF_50_In_Scope_Rows_ELT** task. This should write numerous rows to the **DQF_ROW_RULE_HISTORY_TMP** and **DQF_ROW_RULE_HISTORY** tables in the DQF Mart Schema.

7. In TMC, execute the **DQF_60_Generate_Failed_Rows_Report** task. This should write numerous rows to the **Z_CUSTOMER_DS1** table in the DQF Mart Schema.

8. In TMC, execute the **DQF_70_PopulateNotifications** task. This should write numerous rows to the **DQF_NOTIFICATION** table in the DQF Mart Schema.

9. In Studio, open the **DQF_Template_Jobs\notifications\channels\DQF_Notification_email_channel** Job.

10. Configure the **tSendMail** security and authorization settings to values appropriate for your email server.



SMTP host: context.DQF_Email_Host

☐ SSL Support ☑ STARTTLS Support Importance: Normal

Authentication

Authentication mode: No Auth

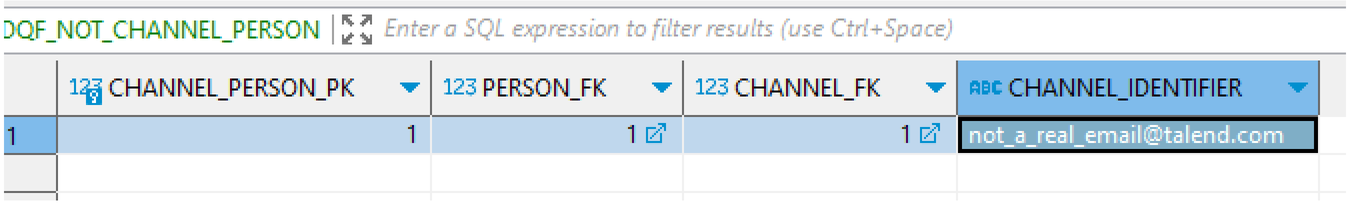
☑ Die on error

11. Save and close the **DQF_Notification_email_channel** Job.

12. Open the **DQF_Notification_message_database_update** Joblet and ensure that the tDBOutput1 component has a context parameter set for the password field. If it does not, set the Property Type fields to Built-in and **Repository**, which should resolve the issue.

13. Open the **DQF_Notification_email_channel** and **DQF_Notification_slack_channel** Jobs and save them to ensure that the Joblet has been updated in each.

14. Browse to the **DQF_NOT_CHANNEL_PERSON** table in your SQL client tool. Enter a valid email address in the **CHANNEL_IDENTIFIER** field for row 1.



DQF_NOT_CHANNEL_PERSON | Enter a SQL expression to filter results (use Ctrl+Space)

	123 CHANNEL_PERSON_PK	123 PERSON_FK	123 CHANNEL_FK	ABC CHANNEL_IDENTIFIER
1	1	1	1	not_a_real_email@talend.com

15. Open the **DQF_Template_Jobs\DQF_80_Notification_Send** Job in Studio and execute it. This should send an email to the address entered in the previous step.

16. Execute the **DQF_85_Refresh_Materialized_Views** task in TMC. This updates the materialized views (MV_ tables). This step only performs actions on MySQL.

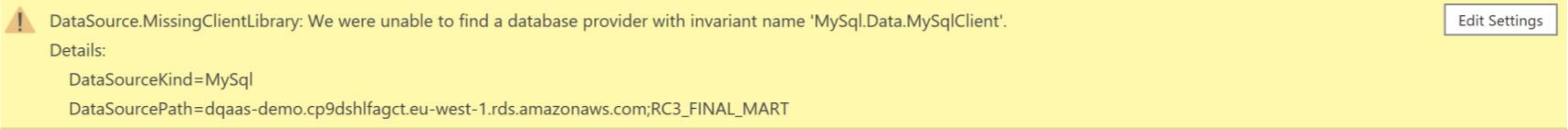
17. Execute the **DQF_90_Finalize_Run_Complete** task in TMC. This updates the existing row in the **DQF_RUN** table in the DQF Mart Schema to denote the first run as complete.

Configuring the demo PowerBI dashboard to point to the DQF Mart Schema

1. Open the **DQF demo dashboard PowerBI Snowflake v1.x.pbix** file or **DQF demo dashboard PowerBI MySQL v1.x.pbix** file in the PowerBI desktop, as appropriate.

2. Follow the steps in the **DQF - How to replace Snowflake data source in powerbi v1.x.mp4** video to point the dashboard to the DQF Mart Schema. Data from the first run should be displayed in the dashboard.

Note: If you receive the following error:



```
DataSource.MissingClientLibrary: We were unable to find a database provider with invariant name 'MySql.Data.MySqlClient'.
Details:
    DataSourceKind=MySql
    DataSourcePath=dqaas-demo.cp9dshlfagct.eu-west-1.rds.amazonaws.com;RC3_FINAL_MART
```

Install the MySQL Connector/NET (available as an option in the installer or standalone) from https://dev.mysql.com/downloads/. During testing it was found that there is a known issue with MySQL and Power BI that requires version **8.0.28** or earlier of the driver to be used.

Deploying the remaining Jobs to TMC and creating an execution plan

1. Right-click the **DQF_Demo_Jobs\A20_Run_Dataset1_RPT** Job in Studio and select **Publish to Cloud**. Publish it to the same environment and workspace as the DQF core Jobs.

2. Right-click the **DQF_Demo_Jobs\B20_Run_Dataset2_RPT** Job in Studio and select **Publish to Cloud**. Publish it to the same environment and workspace as the DQF core Jobs.

3. Right-click the **DQF_Demo_Jobs\C20_Run_Dataset3_RPT** Job in Studio and select **Publish to Cloud**. Publish it to the same environment and workspace as the DQF core Jobs.

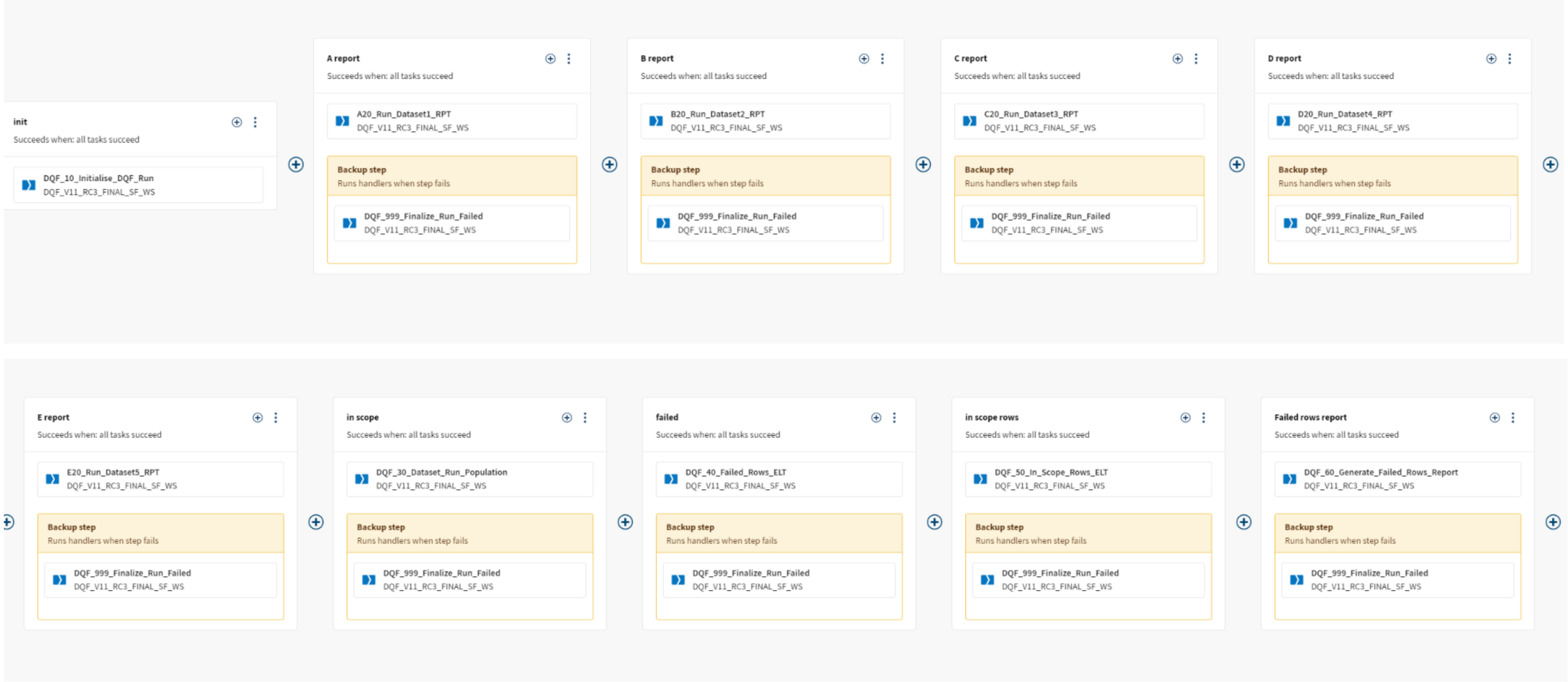
4. Right-click the **DQF_Demo_Jobs\D20_Run_Dataset4_RPT** Job in Studio and select **Publish to Cloud**. Publish it to the same environment and workspace as the DQF core Jobs.

5. Right-click the **DQF_Demo_Jobs\E20_Run_Dataset5_RPT** Job in Studio and select **Publish to Cloud**. Publish it to the same environment and workspace as the DQF core Jobs.

6. Right-click the **DQF_Template_Jobs\DQF_80_Notification_Send** Job in Studio and select **Publish to Cloud**. Publish it to the same environment and workspace as the DQF core Jobs.

7. Finish creating tasks for the newly deployed Jobs in TMC using the **DQF** connection.

8. In TMC, create an execution plan that executes the Jobs as individual steps (as shown below). Add the **DQF_999_Finalize_Run_Failed** Job as an error handler in case of failure.



init
Succeeds when: all tasks succeed
DQF_10_Initialise_DQF_Run
DQF_V11_RC3_FINAL_SF_WS

A report
Succeeds when: all tasks succeed
A20_Run_Dataset1_RPT
DQF_V11_RC3_FINAL_SF_WS
Backup step
Runs handlers when step fails
DQF_999_Finalize_Run_Failed
DQF_V11_RC3_FINAL_SF_WS

B report
Succeeds when: all tasks succeed
B20_Run_Dataset2_RPT
DQF_V11_RC3_FINAL_SF_WS
Backup step
Runs handlers when step fails
DQF_999_Finalize_Run_Failed
DQF_V11_RC3_FINAL_SF_WS

C report
Succeeds when: all tasks succeed
C20_Run_Dataset3_RPT
DQF_V11_RC3_FINAL_SF_WS
Backup step
Runs handlers when step fails
DQF_999_Finalize_Run_Failed
DQF_V11_RC3_FINAL_SF_WS

D report
Succeeds when: all tasks succeed
D20_Run_Dataset4_RPT
DQF_V11_RC3_FINAL_SF_WS
Backup step
Runs handlers when step fails
DQF_999_Finalize_Run_Failed
DQF_V11_RC3_FINAL_SF_WS

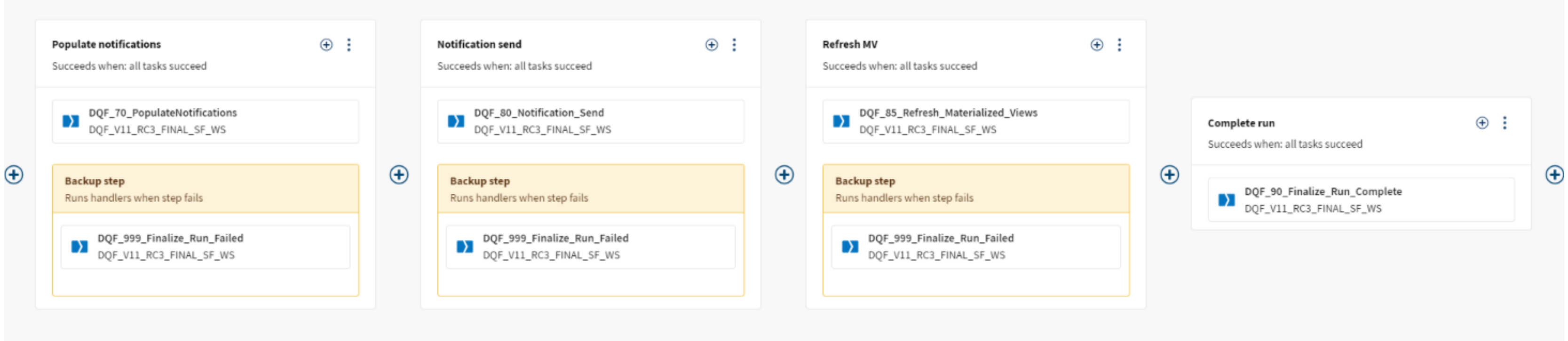
E report
Succeeds when: all tasks succeed
E20_Run_Dataset5_RPT
DQF_V11_RC3_FINAL_SF_WS
Backup step
Runs handlers when step fails
DQF_999_Finalize_Run_Failed
DQF_V11_RC3_FINAL_SF_WS

in scope
Succeeds when: all tasks succeed
DQF_30_Dataset_Run_Population
DQF_V11_RC3_FINAL_SF_WS
Backup step
Runs handlers when step fails
DQF_999_Finalize_Run_Failed
DQF_V11_RC3_FINAL_SF_WS

failed
Succeeds when: all tasks succeed
DQF_40_Failed_Rows_ELT
DQF_V11_RC3_FINAL_SF_WS
Backup step
Runs handlers when step fails
DQF_999_Finalize_Run_Failed
DQF_V11_RC3_FINAL_SF_WS

in scope rows
Succeeds when: all tasks succeed
DQF_50_In_Scope_Rows_ELT
DQF_V11_RC3_FINAL_SF_WS
Backup step
Runs handlers when step fails
DQF_999_Finalize_Run_Failed
DQF_V11_RC3_FINAL_SF_WS

Failed rows report
Succeeds when: all tasks succeed
DQF_60_Generate_Failed_Rows_Report
DQF_V11_RC3_FINAL_SF_WS
Backup step
Runs handlers when step fails
DQF_999_Finalize_Run_Failed
DQF_V11_RC3_FINAL_SF_WS



Populate notifications
Succeeds when: all tasks succeed
DQF_70_PopulateNotifications
DQF_V11_RC3_FINAL_SF_WS
Backup step
Runs handlers when step fails
DQF_999_Finalize_Run_Failed
DQF_V11_RC3_FINAL_SF_WS

Notification send
Succeeds when: all tasks succeed
DQF_80_Notification_Send
DQF_V11_RC3_FINAL_SF_WS
Backup step
Runs handlers when step fails
DQF_999_Finalize_Run_Failed
DQF_V11_RC3_FINAL_SF_WS

Refresh MV
Succeeds when: all tasks succeed
DQF_85_Refresh_Materialized_Views
DQF_V11_RC3_FINAL_SF_WS
Backup step
Runs handlers when step fails
DQF_999_Finalize_Run_Failed
DQF_V11_RC3_FINAL_SF_WS

Complete run
Succeeds when: all tasks succeed
DQF_90_Finalize_Run_Complete
DQF_V11_RC3_FINAL_SF_WS

9. Edit the **DQF_30_Dataset_Run_Population** task and set the **DATASET PK LIST** parameter to **1,2,3,4,5**. This includes all five demo datasets in the DQF run.

10. Execute the execution plan. Troubleshoot any errors and restart from the beginning of the plan if needed.

11. Refresh the data in the PowerBI dashboard. All five demo datasets should now display.

Performing additional runs

The DQF Raw Schema contains multiple data snapshots for each of the demo datasets to aid testing. By copying data from these snapshot tables to the analyzed tables, you can simulate the progression of the datasets over time.

The sequence for this is:

1. Truncate the analyzed table, for example:

```
TRUNCATE TABLE DQF_DEMO;
```

2. Copy data to the analyzed table, for example:

```
Insert into DQF_DEMO Select * from DQF_DEMO_RUN_2;
```