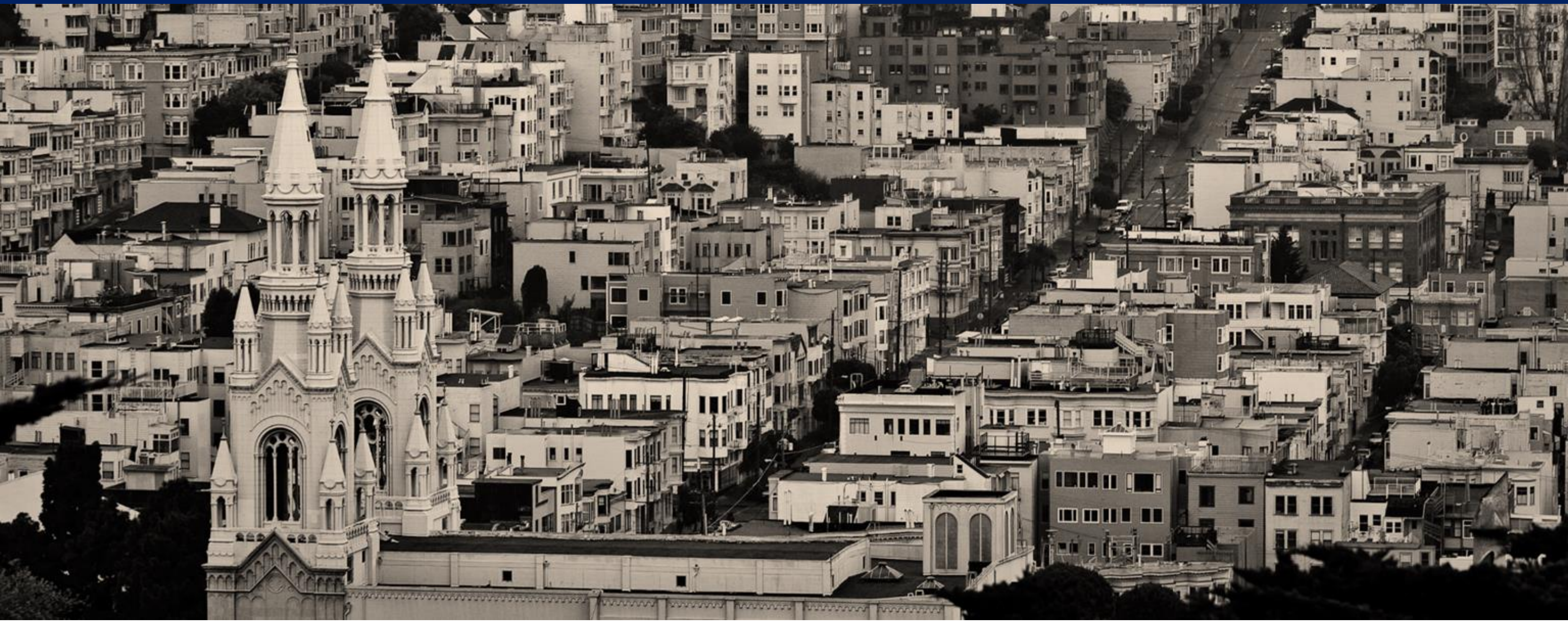
((CENTRIC))
In Balance

Centric Data & Analytics

Data Quality Monitoring & Testing Framework



Centric's Data Quality Monitoring (DQM) and Testing Framework compares actual and expected datasets at a granular level. Each test is evaluated and preserved in a *management database* for future reference. The framework supports test-driven-development of ETL as well ongoing monitoring of data quality in a production environment.

Table of Contents

Terms of Use

Notwithstanding anything to the contrary herein or in any agreement, Centric Consulting shall retain all right, title and interest in the Centric Software. Subject to payment of all amounts due under the Agreement, Centric Consulting hereby grants CLIENT a non-exclusive, royalty-free, world-wide, non-sublicensible/non-transferable, perpetual License to use, reproduce, and modify the Centric Software for its own internal operations.

The foregoing License shall be limited to the current version of the Centric Software, as of the date of this Statement of Work. Client shall have no right to future versions, patches or updates.

The foregoing terms (retention of Centric Software ownership by Centric and license to CLIENT) shall apply to any work done by Centric for CLIENT in which the Centric Software is utilized.

CENTRIC IS PROVIDING THE CENTRIC SOFTWARE ON A “WHERE IS” “AS IS” BASIS. THEREFORE, CENTRIC MAKES NO REPRESENTATIONS OR WARRANTIES WITH REGARD TO THE CENTRIC SOFTWARE, WHETHER EXPRESS OR IMPLIED BY LAW.

IN NO EVENT SHALL CENTRIC BE LIABLE TO FOR CONSEQUENTIAL, SPECIAL, OR INCIDENTAL DAMAGES ARISING OUT OF OR RELATING TO THE CENTRIC SOFTWARE. UNDER NO CIRCUMSTANCES SHALL CENTRIC BE LIABLE FOR ACTUAL DAMAGES OR LOSSES IN EXCESS OF ONE DOLLAR ($1.00).

THE PARTIES AGREE AND ACKNOWLEDGE THAT THE FOREGOING LIMITATION OF LIABILITY IS REASONABLE IN LIGHT OF THE FACT THAT THE CENTRIC SOFTWARE IS BEING PROVIDED FREE OF CHARGE.

Quick Start Guide

The following Quick Start guide will instruct the user on installation and initial configuration of the framework.

Installation

1. Install **Java v.6** or higher on the operating system in which the framework will run. Java can be downloaded here: http://java.com/en/download/

2. Navigate to http://github.com/jkanel/centric-dqm. This is the GitHub repository in which the software is maintained.

3. Click the Download ZIP button to download the software in a Zip file format. The file downloaded will be called **centric-dqm-master.zip**

4. Extract the contents of **centric-dqm-master.zip** to a target folder from which the application will be run.

5. The **jar_exec_win.bat** is a windows batch file that should be used to run the executable JAR file. Note that the batch file contains example command line parameters that may differ from desired settings. Please review the documentation below in order to properly configure the command line parameters.

   The batch file references a folder “addons” in the same director as the JAR file. This folder is used to hold DLLs that are called by Java libraries inside the JAR. It is important that, if these libraries will be called during execution of the JAR file, the “addons” folder be specified as a command line parameter inside the batch file as seen below:

```
java -Djava.library.path=".\addons" -jar "com.centric.dqm.jar"
```

6. Note that the **com.centric.dqm.jar** file is the executable JAR which comprises the framework software.

7. Note that the **com.centric.dqm.properties** is a required configuration file used to direct the application to its management database.

Properties Configuration

The **com.centric.dqm.properties** file contains properties that direct the application on how to connect to the management database. This file must be configured before the application will work.

1. Create the management database. We recommend that this database be setup on the same system as the software.

   NOTE: *At this time only SQL Server is supported to host the management database.*

2. Create the user and password. Grant the user DDL privileges on the management database at least until the database tables have been bootstrapped.

The following code block shows contents of an example properties file. Database and user names below are only an example.

```
driver=com.microsoft.sqlserver.jdbc.SQLServerDriver
url=jdbc:sqlserver://localhost;databaseName=dqm_mgr
user=dqm_admin
password=123$dqm_admin_password
maxrows=0
```

3. Configure the `driver` property. This value is the JDBC driver string corresponding to the management database.

4. Configure the `url` property. This value is the JDBC connection URL that specifies the server and management database.

5. Configure the `user` property. This value is the login for the management database.

   NOTE: *If an LDAP authenticated login is used, the url must include the appropriate security reference and the user account executing the application will be the principal.*

   NOTE: *If you wish to allow the software to automatically create the management database, this user must have DDL and read/write privilege.*

6. Configure the `password` property. This value is the corresponding password to access the management database.

7. We recommend that `maxrows` be assigned a zero value by default. Customization of this value is discussed later in the document.

8. Run the executable JAR file with no command line parameters set. Provided that database login user has sufficient privileges, ***the software will bootstrap the management database***.

```
cd {installation directory}
java –jar “com.centric.dqm.jar”
```

Scenario Configuration

Scenarios are test definitions. These must be configured along with their corresponding connections. A scenario pulls actual and expected datasets from their respective “comparison databases”.

1. Create a `connection` table record for each comparison database whose data will be compared in a test. The following fields are required:

    a. `connection_uid` is a manually created string field that uniquely identifies the connection throughout the framework.
    b. `jdbc_driver` is the JDBC driver string corresponding to the comparison database.
    c. `jdbc_url` is the JDBC url string corresponding to the comparison database.
    d. `username` is the database login name used to connect to the comparison database.
    e. `password` is the database password used to connect to the comparison database.

All other `connection` table columns can initially use their default values.

2. Create a `scenario` table record for each combination of actual/expected query that will be tested.

    a. `scenario_uid` is a manually created string field that uniquely identifies the scenario throughout the framework.
    b. `expected_connection_uid` is the identifier of the connection (from the connection table) used to execute the expected dataset.
    c. `expected_command` is the query command (SQL statement or stored procedure call) that returns the expected dataset.
    d. `actual_connection_uid` is the identifier of the connection (from the connection table) used to execute the actual dataset.

e. `actual_command` is the query command (SQL statement or stored procedure call) that returns the expected dataset.

f. `active_flag` should be set to `“Y”` for those scenarios that will be automatically run as tests.

All other `scenario` table columns can initially use their default values.

Framework Execution

The framework may be executed by running the executable JAR, from the command line. This will run all the scenarios as tests.

```
cd {installation directory}
java -jar “com.centric.dqm.jar”
```

The results of the execution can be seen in the command window or in the **com.centric.dqm.log** file.