



ihiTM Data Quality

Installation and User's Guide

Version 5.2.0 | May 2023

Contents

1. Introducing ibi™ Data Quality	11
Overview	11
What's Included in ibi Data Quality?	11
Benefits	12
Save Cost & Time.....	12
Empower Citizen Users.....	12
Accelerate Cloud Migration Products.....	12
Improve Business Outcomes.....	13
Promote Accountability.....	13
Key Features	13
Profiling & Classification.....	13
Rules Driven Domain Analysis.....	13
Data Quality Observability.....	14
Intuitive Web UIs	14
Seamless Integration.....	14
Complementary Tools and Services.....	14
2. Design and Architecture	17
High Level Design	17
Consumer-Focused Components.....	19
Producer-Focused Components.....	19
Internal Components.....	20
Containers.....	21
Security Features.....	22
Concepts	23
Data Class.....	23
Rule.....	24
Rules Catalog.....	25
Authoring.....	25
Service.....	26
Service Catalog.....	27
Authoring.....	27
Registration.....	27

Execution.....	27
Multiple Rules Using One Service.....	28
Data Profiling.....	29
Data Quality (DQ) Analysis.....	30
Scoring.....	30
Profile Score.....	31
Data Quality (DQ) Score.....	32
3. Deploying ibi™ Data Quality	33
Configurations	33
Network Ports.....	33
Valet User Interface.....	34
WSO2 Identity Server.....	34
Valet API Server.....	36
Watchdog.....	37
Postgres Database.....	38
User Interface.....	38
Loqate API Access Keys	39
Docker Deployment	40
Prerequisites.....	40
Installing Docker Desktop or Docker Engine with Compose.....	40
Installing Loqate With Data Packs.....	41
Installing WebFOCUS DSML Services Container Edition.....	41
Installing ibi Data Quality.....	42
Kubernetes Deployment	44
Kubernetes Overview.....	44
Hardware Requirements.....	44
Installation.....	46
Building Docker Images for Kubernetes.....	46
Deploying ibi Data Quality to a Kubernetes Cluster.....	47
Deploying Infrastructure Components.....	47
Verifying and Testing.....	48
Confirming Services and Components are Running.....	48

Logging in to the ibi Data Quality Console.....	49
JDBC Driver Setup.....	50
New Valet Services Container.....	50
Existing Valet Services Container.....	50
High Availability.....	51
Updating Loqate License Keys.....	51
4. Navigating and Using ibi Data Quality.....	53
Signing In to ibi Data Quality.....	53
Supported Web Browsers.....	54
Supported Resolution and Display Layout.....	55
Uploading Data.....	55
Uploading Data From a Text File.....	56
Uploading Data From a JDBC Data Source.....	58
New JDBC Connection.....	58
Existing JDBC Connection.....	59
Sample JDBC Connection Strings.....	59
Tracking Upload Progress.....	60
Profiling Data.....	60
Viewing and Downloading a Profile.....	61
Variables.....	62
Duplicates.....	64
Correlations.....	65
Adding Correlations.....	66
Supported Correlations.....	68
Pearson Correlation.....	68
Spearman Correlation.....	69
Kendall Correlation.....	70
Comparison.....	71
Advanced Insights.....	72
Cluster Analysis.....	72
K-Means Clustering.....	73
Analyzing Data Quality.....	75

Adding Rules.....	76
Single Variable Rules.....	78
Multi-Variable Rules.....	78
Preassigned Rules.....	79
Tracking Analysis Progress.....	81
Viewing a Summary of Analysis Results.....	81
Viewing Results.....	83
Results Output.....	83
Rerun Analysis With Different Rules.....	87

5. Management and Monitoring 89

Managing Services.....	89
Packaged (Built-in) Services.....	89
Adding New Services.....	89
Service Requirements.....	89
Request Method.....	90
Authentication Method.....	90
Service Inputs.....	90
Service Outputs.....	90
Service Parameters.....	91
Service Registration.....	91
Service Definition JSON.....	91
Service Registration Endpoint.....	94
Authoring Services Using Omni-Gen Data Quality Server.....	94
Project Structure and Deployment.....	95
Managing Data Classes.....	96
Packaged (Built-in) Data Classes.....	96
Defining New Data Classes.....	98
Regular Expression.....	99
Pattern or Mask.....	99
Lookup.....	102
Verifying New Data Classes.....	102
Editing Data Classes.....	103

Managing Rules	104
Adding New Rules.....	104
Defining a New Rule.....	104
Verifying a New Rule.....	108
Managing Users	109
Adding New Users.....	110
Managing Existing Users.....	113
Recommended Groups by User Roles.....	115
Monitoring Data Quality Metrics	117
DQ Metrics Views.....	117
Profile Summary.....	117
Profile Details.....	119
DQ Summary.....	122
DQ Details.....	124
DQ Transaction Summary.....	126
Managing Artifacts	128
Exporting Artifacts.....	128
Importing Artifacts.....	129
A. API Interactions	133
High Level Flow	133
Authorize.....	134
Upload Data Set.....	136
Profile Data.....	140
Check Profile Status.....	142
Deduplicate.....	144
Numeric Analysis.....	145
Correlation Analysis.....	147
K-Means Clustering Analysis.....	149
Download Profile.....	151
Download Correlation Analysis Results.....	152
Download K-Means Clustering Results.....	153
Get List of Rules.....	154

Match Rules	156
DQ Analysis With Rules	157
Check DQ Analysis Status	160
Download All Results	162
Download Results Summary	162
Delete Analysis Results	163
Delete Entire Dataset	164
Transactional Requests	166
B. JSON Schemas	169
Profiling Results JSON Schema	170
K-Means Cluster Analysis Results JSON Schema	176
Correlation Analysis Results JSON Schema	177
DQ Analysis Summary Results JSON Schema	178
C. Creating Python Services	181
Understanding the Service Requirements	181
Creating Your Python Scripts	182
Creating a New Cleanse Service	182
Exposing a Cleanse Service as an API	183
Testing Your Web Application	183
Service Registration	183
D. Packaged DQ Services	187
Analyze Text Sentiment	187
Cleanse Address Using Loqate	189
Cleanse Australian Bank Account Number	191
Cleanse Australian Business Number	192
Cleanse Australian Phone Number	193
Cleanse Date	194
Cleanse Email	196
Cleanse Email With Loqate	197
Cleanse Payment Card	198
Cleanse Phone Using Loqate	199
Cleanse USA DEA	201

Cleanse USA NPI	202
Cleanse USA Phone	203
Cleanse USA SSN	205
Compare Pair Values	206
Detect Outliers Using Interquartile Range	208
Detect Outliers Using Z-Score	209
Impute Missing Numeric Values	210
Verify Age	211
Verify Value in Range	212
Verify Value of Data Type	214
Verify With Regular Expression	215
E. Troubleshooting	217
Locating System Logs	217
Docker Desktop.....	217
Linux.....	220
Copying Logs From a Container.....	220
Common Problems	221
Legal and Third-Party Notices	223

Introducing ibi™ Data Quality

This chapter provides an introduction to ibi™ Data Quality and describes benefits, key features, and key concepts.

In this chapter:

- [Overview](#)
 - [What's Included in ibi Data Quality?](#)
 - [Benefits](#)
 - [Key Features](#)
-

Overview

ibi™ Data Quality connects “data people” with measurable, reportable, and actionable facts about their data. Users who rely on data to make everyday business decisions often deal with poor quality or untrusted data. ibi Data Quality provides these users with a complete view of the trustworthiness of their data, improves the quality of the data that drives business outcomes, delivers a rich user experience, and enables seamless communication between the different systems and stakeholders involved in the data value chain.

What's Included in ibi Data Quality?

This section provides a summary of the products and add-ons that are included with ibi Data Quality.

Products

- ibi™ Data Quality
- ibi™ Omni-Gen®**
- ibi™ iWay® Service Manager
- ibi™ Data Migrator

**ibi Omni-Gen is included for current Omni-Gen DQ customers for upward compatibility.

Add-Ons

- ibi™ DQ Service Provider - DQS Add-on***

- ibi™ DQ Address Engine
- ibi™ DQ Address for [Country]
- ibi™ DQ United States - CASS Add-on
- ibi™ DQ Canada - SERP Add-on
- ibi™ DQ Australia - AMAS Add-on
- ibi™ DQ Address for Asia Pacific
- ibi™ DQ Address for North America
- ibi™ DQ Email Validation (API calls metered)
- ibi™ DQ Phone Validation (API calls metered)

***DQS is included for current Omni-Gen DQ customers for upward compatibility.

Benefits

ibi Data Quality provides users with access to high quality data with faster time to value.

Save Cost & Time

Use out-of-the-box rules that implement complex workflows to analyze, fix, and augment data for several data domains: customer, employee, partner, and more. With lots of built-in features, free up your valuable resources for higher value tasks.

Empower Citizen Users

Provide a low/no-code user interface and unified experiences for non-technical users to accomplish more with less. Simple intuitive browser-based UIs enable citizen users, data analysts, and data scientists to use a wide range of data quality services without the need for IT support and without wasting time on manual and repetitive data validation tasks.

Accelerate Cloud Migration Products

Get the most out of your cloud investment by preventing bad data from entering. Improve accuracy, reduce maintenance, and govern efficiently by adding data quality firewalls to data migration jobs. Improve the reliability and accuracy of data to result in only high quality data.

Improve Business Outcomes

With data quality rules and API services, deliver only trusted data to decision points. Enable the desired business outcomes: growth, customer satisfaction & retention, process & resource optimization, cost reduction, risk mitigation, increased shareholder value.

Promote Accountability

Enable collaboration between data managers and data consumers with up-to-date and reliable data quality metrics. With business users participating in data governance programs, everyone in the organization feels responsible for implementing data quality best practices and maintaining high data quality.

Key Features

ibi Data Quality provides an extensive set of tools and facilities, all which are accessible through a user-friendly, browser-based interface.

Profiling & Classification

Understand your data from several perspectives: what it is, and how it can be used and improved.

- ❑ **Exploratory Data Analysis.** Profile data to explore and understand its characteristics such as data type, unique values, missing values, frequent values, stats, scores, and more.
- ❑ **Advanced Profiling Options.** Perform advanced analysis such as duplicate detection, correlation analysis, clustering analysis, etc.
- ❑ **Data Classification.** Automatically discover and categorize data values into known data classes, adding business context and aiding in the identification of sensitive data.
- ❑ **Define Custom Data Classes.** Define rules to classify new and/or proprietary data for your business.

Rules Driven Domain Analysis

Verify your data against known rules to ascertain if that data accurately represents the real-world entity.

- ❑ **Prepackaged Rules and Services.** Browse through the Rules Catalog and attach your data with generic, domain, or business specific rules that automatically verify, cleanse, standardize, and enrich your data.
- ❑ **Stats, Scores and Results.** Preview the verification and cleansing results, evaluate data quality scores, and ingest the standardized and enriched output.

- ❑ **Create Custom Rules.** Create custom rules that meet your business needs and publish them into a centralized Rules Catalog for everyone to reuse.
- ❑ **Register New Services.** Create workflows in any tool and register them as services so they can be associated with Rules and reused by everyone in the enterprise.

Data Quality Observability

Collaborate with data experts and improve visibility of data quality initiatives by monitoring and tracking progress.

- ❑ Track, trend, and monitor data quality stats over time.
- ❑ Assess trustworthiness of your data by comparing scores against benchmarks.
- ❑ Monitor KPIs and setup alerts that send notifications to stakeholders whenever those metrics fall outside the normal thresholds.

Intuitive Web UIs

Enable your “data people” to self-serve powerful data analysis features that automatically detect and remedy data quality issues via an intuitive browser-based user interface.

Seamless Integration

Use REST API services to introduce DQ Rules anywhere in your data pipeline and publish standardized and enriched data to downstream applications.

Complementary Tools and Services

ibi Data Quality comes with “batteries included” - all tools and services needed to efficiently and effectively organize and implement data quality services in your enterprise architecture.

- ❑ Authoring tool that includes a library of pre-configured components and a rich set of parsing, cleansing, and enrichment functions for technical users to author workflows as reusable web services.
- ❑ A highly scalable enterprise service bus, with fully integrated service design-time environment and web services for orchestrating flows that embed data quality functions.
- ❑ A broad category of tools to facilitate and automate the extraction of data from various sources, transformation of data using data quality rules and the load of clean data into target.
- ❑ Address verification with geocoding for over 240 countries.

- ❑ Email verification to reduce email bounce rates with realtime email address checking.
- ❑ Phone number verification for fast and efficient communications by identifying the location of a phone number and the carrier.

Chapter 2

Design and Architecture

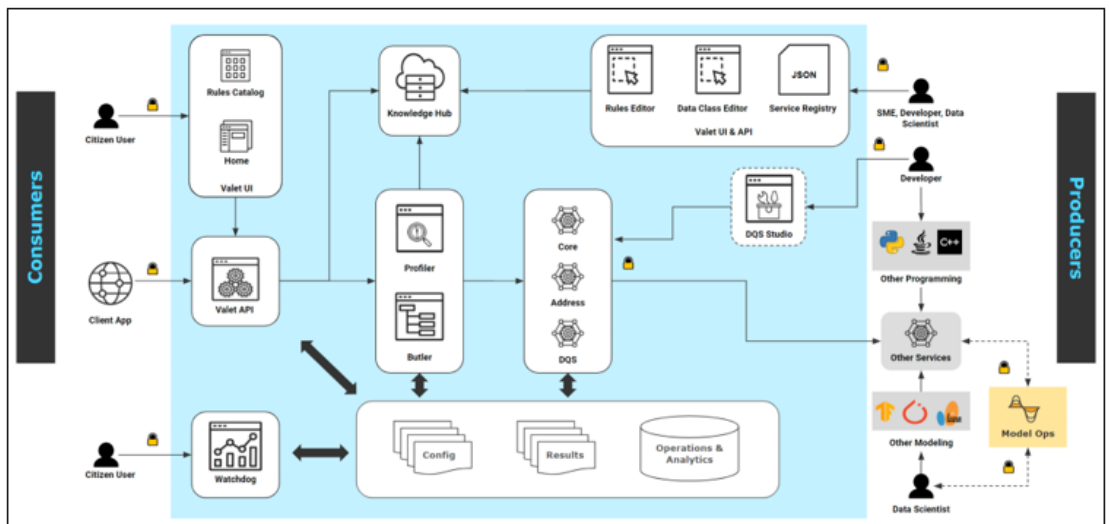
This chapter describes the high level design and architecture of ibi™ Data Quality.

In this chapter:

- ❑ [High Level Design](#)
- ❑ [Concepts](#)

High Level Design

ibi Data Quality has several application components that provide a rich set of services to the consumers of data quality services, as well as the producers of those services.

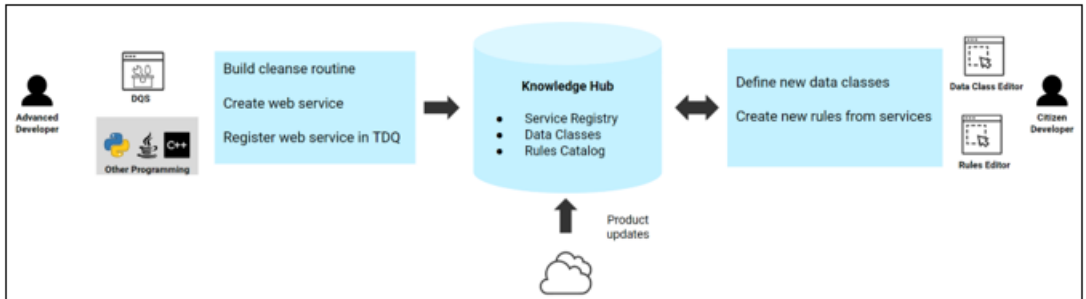


Producers

Producers are users who play the role of citizen developer or advanced developer. These users will interact with the application to create new Services, define new Data Classes, or author new Rules. For more information on Data Classes, Rules, and Services, see [Concepts](#) on page 23.

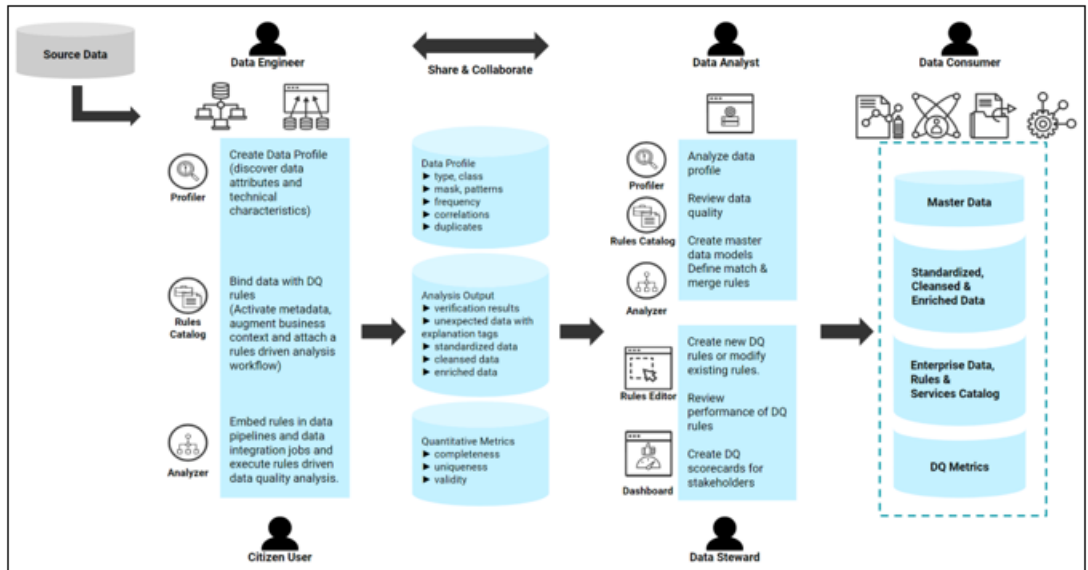
ibi Data Quality has an open Service authoring architecture (i.e., developers have several different options to create new Services):

- Option 1.** Use built-in Services to author new Rules.
- Option 2.** Use ibi Omni-Gen Data Quality Server (DQS) as an authoring tool to create new or custom cleanse plans.
- Option 3.** Create and register a RESTful Service using any other tool or language, such as Python, Java, etc.



Consumers

Consumers are users who play the role of citizen user, data engineer, data analyst, data scientist, data steward, etc. These users will interact with the application to discover data attributes and other technical characteristics, discover attribute relationships, detect outliers and anomalies, embed data quality analysis in data pipelines and data integration jobs, evaluate and review data quality scores and monitor and govern their data assets on different data quality dimensions.



Consumer-Focused Components

Valet API

This is the main pillar of ibi Data Quality. It powers all the other components of the solution. It also provides API services for programmatic interaction with ibi Data Quality.

Valet UI

This is a no-code user interface that enables citizen users to upload data, run profiling analysis, associate Rules with the data to execute data quality analysis, and download the results.

Rules Catalog

This is a discovery-focused user interface that enables data analysts to search for Rules applicable to their data.

Watchdog

This is a reporting user interface that allows data stewards to track, monitor, and report data quality metrics.

Producer-Focused Components

Rules Editor

This is a no-code user interface that allows data experts and data owners to author and publish new Rules from existing Services.

Data Class Editor

This is a no-code user interface that allows data experts and data owners to define new data classes using character masks/patterns, regular expressions and lookups.

Service Registry

This is a set of API services that allows developers to test new Services and register those Services in their ibi Data Quality instance.

DQ Studio

This is a Service authoring tool that allows developers to build new Services.

Internal Components

Knowledge Hub

This is the central repository of definitions and related artifacts for Data Classes, Services, and Rules.

Profiler

This is a scalable engine that performs comprehensive technical analysis of data.

Butler

This is an orchestration engine that interprets incoming requests and routes those requests to Service Providers.

Service Providers (Core, Address, DQS)

This is a set of Service Providers that accept requests from the Butler, execute data quality analyses, and respond with output results.

Analytics Repository

This is a relational data store used for persisting profiling and data quality stats, metrics, and scores.

Results Repository

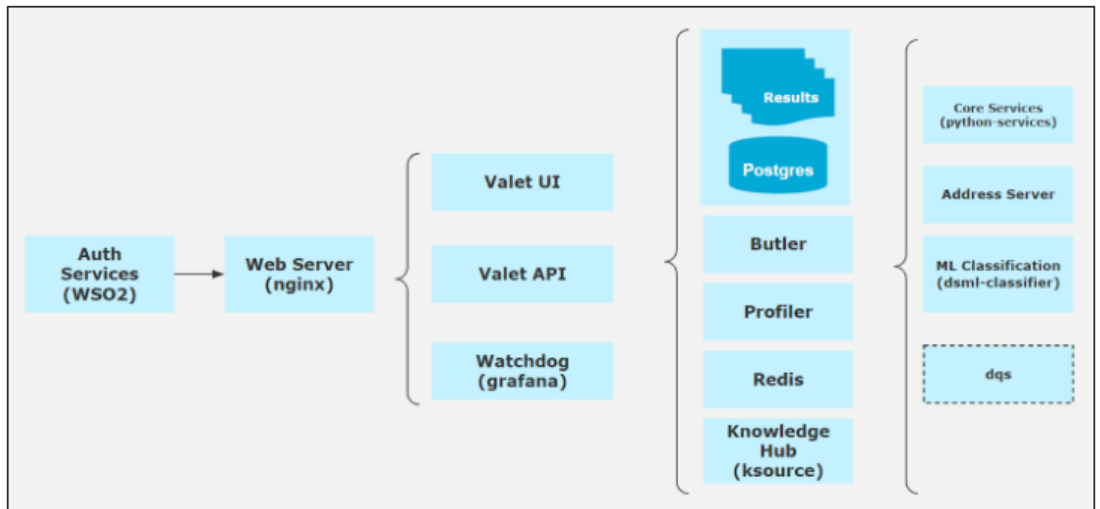
This is a persistent file storage system used for storing profiling and data analysis results (input data, deduplicated data, cleansed output, detailed results, and summarized reports).

Containers

The following table lists and describes the containers that comprise ibi Data Quality.

Container Name	Description
address-server	Performs address cleansing using Loqate.
butler	Orchestrates DQ analysis by interpreting Rules and routing requests to underlying services (python or DQS).
dqs	(Optional) Runs DQS workflows if the customer opts to use DQS to author and run data quality workflows.
dsml-classifier	Provides DSML models to classify data.
grafana	Hosts the Watchdog app that provides reports and dashboards on DQ metrics.
ksource	Provides API to manage data classes, rules and services.
nginx	Provides main ingress for ibi Data Quality.
postgres	Provides relational data store for shared application configuration and operational data.
profiler	Performs technical analysis on input data to generate data profile.
python-services	Performs data quality analyses on input data to generate data quality results.
redis	Provides a simple queue service for profiler.
valet-services	Provides back-end services for the Valet UI and APIs for programmatic interactions with ibi Data Quality.

Container Name	Description
valet-ui	Provide user interface for citizen users to interact with ibi Data Quality via web browser.



Security Features

All external interactions with ibi Data Quality use HTTPS. For more information on steps to change the default certificate for the Kubernetes ingress or the *nginx* proxy used by the Docker Compose, see [Deploying ibi™ Data Quality](#) on page 33.

ibi Data Quality requires that *valet-ui*, *valet-services*, and *watchdog* be exposed to the external network, along with endpoints in the WS02 Identity Server that are required for OAuth2 and OpenID Connect. For convenience when running in trusted environments, Postgres and the WS02 Identity Server console are exposed by default. In production or untrusted environments, you can disable access to these services by following the steps described in [Deploying ibi™ Data Quality](#) on page 33.

Authentication and Authorization in ibi Data Quality use the OAuth2 and OpenID Connect protocols, with signed JSON web tokens. Any access to the application requires authentication, and specific operations require roles or permissions as described in [Managing Users](#) on page 109.

The *valet-services* API provides an authentication service that allows an API user to acquire a bearer token. You must supply this token in the HTTP Authorization header for all requests made through the API. The bearer token expires after one hour. For more information, see [Authorize](#) on page 134.

Concepts

This section describes ibi Data Quality key concepts, facilities, and terminology you should familiarize yourself with prior to using the product.

Data Class

In ibi Data Quality, a Data Class represents a real-world entity.

Example: SSN, Credit Card, Email

Data classes are associated with a Sensitive Data flag to identify data classes that represent data objects containing confidential information.

The ibi Data Quality Profiler uses metadata classification algorithms to classify data attributes and associate them with known Data Classes. The ibi Data Quality Knowledge Hub provides the definitions for all the Data Classes that can be recognized by the Profiler. Users can review a list of Data Classes from the Data Class tab on the user interface. Users can also add new or custom Data Classes using the Data Class Editor.

For more information, see [Managing Data Classes](#) on page 96.

A sample definition for the built-in Data Class called *us_ssn* is shown below.

Data Class (us_ssn)

Name *	Sensitive *
us_ssn	true

Description *

United States Social Security Number



Regular Expression

Regular Expression *

```
^\d{3}(\s|^\s|_|)(0,1)\d{2}(\s|^\s|_|)(0,1)\d{4}$
```

Rule

In ibi Data Quality, a Rule represents an implementation of a Service that can validate, cleanse, and/or enrich a class of data.

- Rules are implemented via Services.
- Rules are created by configuring Service parameters.
- Rules are associated with metadata such as description, data class, entity, industry, country, etc.
- Multiple Rules can be created using a single Service.

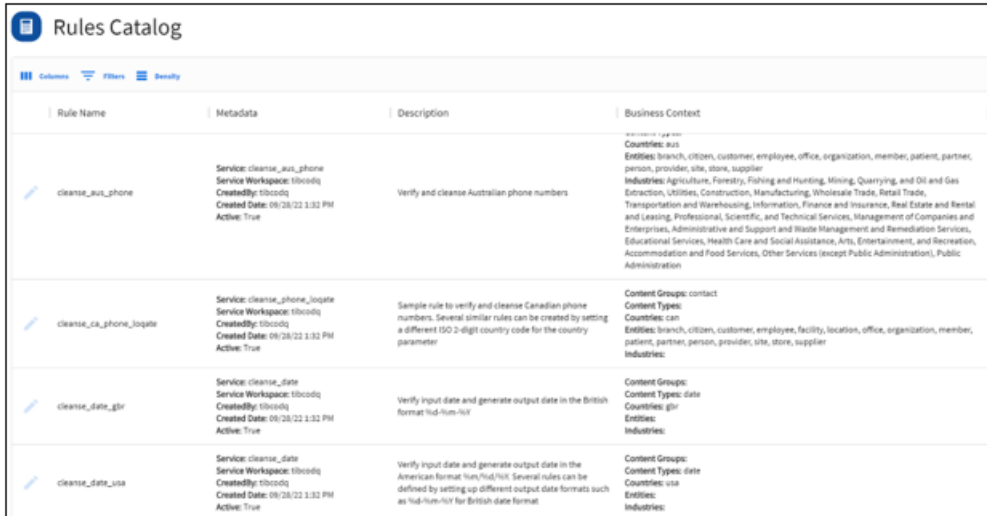
Example: cleanse_ssn, cleanse_payment_card, cleanse_email

ibi Data Quality delivers a set of sample Rules for known Data Classes. Users can author new Rules using the Rules Editor. For more information, see [Managing Rules](#) on page 104.



Rules Catalog

Users can search the Rules Catalog and find Rules to associate with different data attributes. Rules are configured with Rule metadata that provides users additional context to find the most appropriate Rules for their data. Rule metadata generally includes Description, Data Classes, Industries, Geographies, Entities, etc.



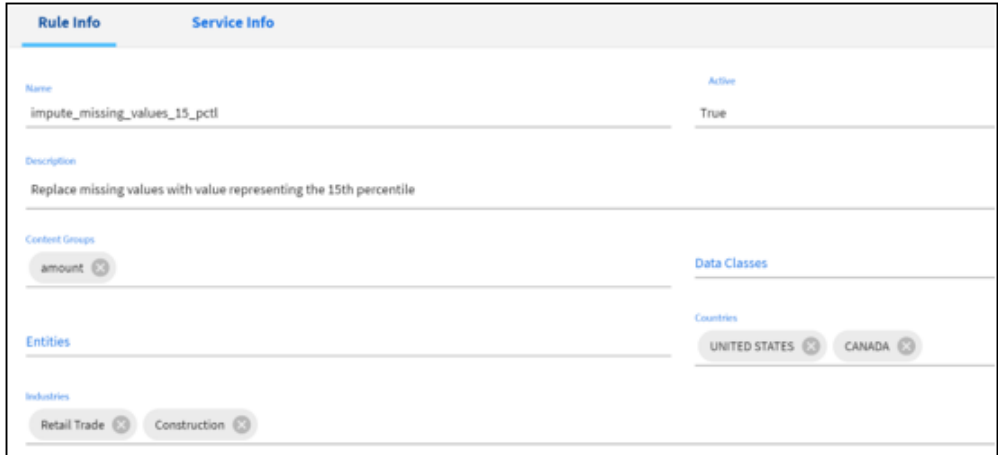
Rule Name	Metadata	Description	Business Context
cleanse_aus_phone	Service: cleanse_aus_phone Service Workspace: tibcooq Created By: tibcooq Created Date: 09/28/22 1:32 PM Active: True	Verify and cleanse Australian phone numbers	Country: aus Entities: branch, citizen, customer, employee, office, organization, member, patient, partner, person, provider, site, store, supplier Industries: Agriculture, Forestry, Fishing and Hunting, Mining, Quarrying, and Oil and Gas Extraction, Utilities, Construction, Manufacturing, Wholesale Trade, Retail Trade, Transportation and Warehousing, Information, Finance and Insurance, Real Estate and Rental and Leasing, Professional, Scientific, and Technical Services, Management of Companies and Enterprises, Administrative and Support and Waste Management and Remediation Services, Educational Services, Health Care and Social Assistance, Arts, Entertainment, and Recreation, Accommodation and Food Services, Other Services (except Public Administration), Public Administration
cleanse_ca_phone_locale	Service: cleanse_phone_locale Service Workspace: tibcooq Created By: tibcooq Created Date: 09/28/22 1:32 PM Active: True	Sample rule to verify and cleanse Canadian phone numbers. Several similar rules can be created by setting a different ISO 3-digit country code for the country parameter	Content Group: contact Content Type: can Entities: branch, citizen, customer, employee, facility, location, office, organization, member, patient, partner, person, provider, site, store, supplier Industries:
cleanse_date_gbr	Service: cleanse_date Service Workspace: tibcooq Created By: tibcooq Created Date: 09/28/22 1:32 PM Active: True	Verify input date and generate output date in the British format %d-%m-%Y	Content Group: date Content Type: date Country: gbr Entities:
cleanse_date_usa	Service: cleanse_date Service Workspace: tibcooq Created By: tibcooq Created Date: 09/28/22 1:32 PM Active: True	Verify input date and generate output date in the American format %m/%d/%Y. Several rules can be defined by setting up different output date formats such as %d-%m-%Y for British date format	Content Group: date Content Type: date Country: usa Entities: Industries:

Authoring

Users can define and add new rules using the Rules Editor.

Rules are generally associated with metadata to provide the context in which it should be executed. Rules are created and managed in the Rules Editor where the author specifies the rule metadata, selects a Service and configures the Service parameters. Multiple Rules can be created using a single Service.

For more information, see [Managing Rules](#) on page 104.



Service

In ibi Data Quality, a Service represents a workflow that contains the logic to cleanse, validate, and enrich input data.

- Services can be generic or domain-specific.

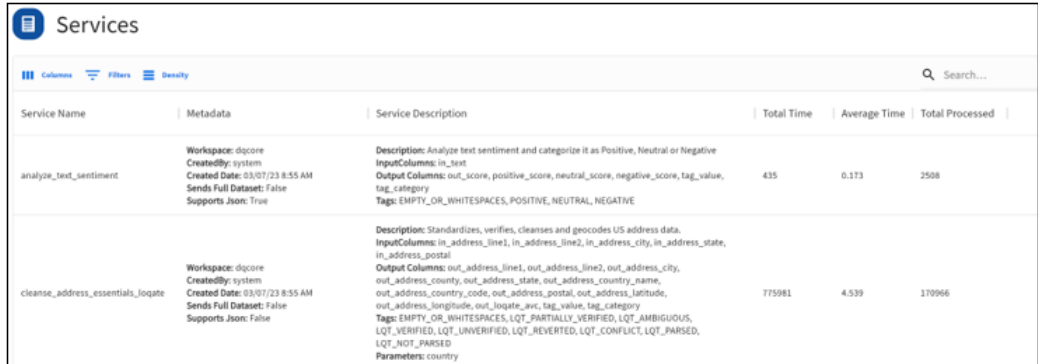
Examples:

- Generic:** cleanse_date, compare_values
- Domain-specific:** cleanse_ssn, cleanse_payment_card, cleanse_email
- ibi Data Quality delivers a set of prebuilt services.
- ibi Data Quality ships with containers that host product delivered Services and DQS Services (if the customer opts to add DQS as a Service Producer).
- All other Services have to be hosted outside ibi Data Quality and published as a RESTful Service that can be accessed by ibi Data Quality.
- Users can test and register new Services using the API.

For more information, see [Managing Services](#) on page 89.

Service Catalog

Users can search and find the different services registered in their environment. The Service Catalog shows service description, metadata, and operational metrics such as number of requests executed and the execution time.



Service Name	Metadata	Service Description	Total Time	Average Time	Total Processed
analyze_text_sentiment	Workspace: dqcore Created By: system Created Date: 03/07/23 8:55 AM Sends Full Dataset: False Supports Json: True	Description: Analyze text sentiment and categorize it as Positive, Neutral or Negative Input Columns: in_text Output Columns: out_score, positive_score, neutral_score, negative_score, tag_value, tag_category Tags: EMPTY_OR_WHITESPACES, POSITIVE, NEUTRAL, NEGATIVE	435	0.173	2508
cleanse_address_essentials_loqate	Workspace: dqcore Created By: system Created Date: 03/07/23 8:55 AM Sends Full Dataset: False Supports Json: False	Description: Standardizes, verifies, cleanses and geocodes US address data. Input Columns: in_address_line1, in_address_line2, in_address_city, in_address_state, in_address_postal Output Columns: out_address_line1, out_address_line2, out_address_city, out_address_country, out_address_state, out_address_country_name, out_address_country_code, out_address_postal, out_address_latitude, out_address_longitude, out_loqate_src, tag_value, tag_category Tags: EMPTY_OR_WHITESPACES, LQT_PARTIALLY_VERIFIED, LQT_AMBIGUOUS, LQT_VERIFIED, LQT_UNVERIFIED, LQT_REVERTED, LQT_CONFLICT, LQT_PARSED, LQT_NOT_PARSED Parameters: country	775981	4.539	170966

Authoring

Developers can author a data quality workflow in any language or tool and deploy it as a RESTful Service. Services have to comply with ibi Data Quality's service requirements. For more information, see [Service Requirements](#) on page 89.

Customers can also opt to purchase a license for ibi Omni-Gen Data Quality Server (DQS) to build DQ plans and deploy them into the DQS container. For more information, see [Authoring Services Using Omni-Gen Data Quality Server](#) on page 94.

Registration

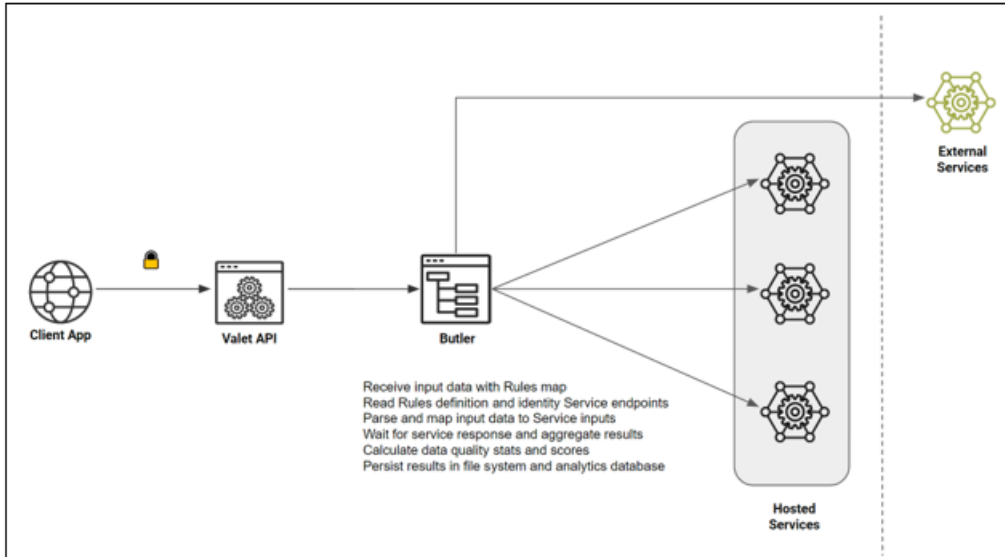
Developers can test and register new Services using the Register Services API. For more information, see [Service Registration](#) on page 91.

Execution

In ibi Data Quality, client applications can execute a DQ Service request via Rules. Clients cannot send direct requests to Service endpoints.

Note: In order to execute a DQ Service, you must create a Rule that references the Service.

When a client application submits a request to execute a set of Rules against input data attributes, ibi Data Quality reads the Rule definitions, parses and maps the data attributes to Service inputs, routes the requests to the corresponding Service endpoints, aggregates the responses for all the Service requests, generates data quality stats and metrics, and persists the results in file system and the analytics database.



Multiple Rules Using One Service

This is a simple illustration of a single Service used to create multiple Rules in ibi Data Quality. We are going to use a built-in Service named *cleanse_date* that takes input date values in any format, automatically infers the input date format, verifies if the date values represent valid calendar dates, and produces output dates in the user-specified format.

Service Details:

- Name:** `cleanse_date`
- Inputs:** `in_date`
- Outputs:** `out_date`, `date_format`, `tag_value`, `tag_category`
- Parameters:**
 - default_date:** A default date value that is used to replace invalid or missing date values in the output.

- ❑ **date_format:** Date format to represent valid or cleansed output dates.

We are going to define two Rules by specifying different values for the Service Parameters.

1. *cleanse_date_usa* will generate output dates in “%m/%d/%Y” format and will default invalid or missing dates to 01/01/1950.
2. *cleanse_data_gbr* will generate output dates in “%d-%m-%Y” format and will default invalid or missing dates to 01-01-1950.

As shown below, both of these Rules refer to the same Service, but are configured with different values for the Service parameters. For more information on how to create new Rules, see [Adding New Rules](#) on page 104.

The screenshot displays two rule configuration panels. The top panel is for 'Rule (cleanse_date_gbr)' and the bottom panel is for 'Rule (cleanse_date_usa)'. Both panels show 'Service Workspace' as 'Tibco DQ' and 'Service' as 'cleanse_date'. Under 'Service Parameters', the top rule has 'date_format' as '%d-%m-%Y' and 'default_date' as '01-01-1950'. The bottom rule has 'date_format' as '%m/%d/%Y' and 'default_date' as '01/01/1950'.

Data Profiling

ibi Data Quality's Profiler performs technical analysis of data to generate output, such as:

- ❑ Number of columns, data types, data classes.

- ❑ Completeness, uniqueness, patterns, masks, etc.
- ❑ Advanced options: Dedup data, discover correlations, identify outliers.
- ❑ Profiling scores: A score that reflects the trustworthiness of the data both at the column level and at the data set level.

When users upload a new data set, they have the ability to add the source metadata. They also have the ability to select the data attributes for profiling analysis and can set up various data expectations. User defined data expectations for an input data set factor into the calculation of Profile and DQ Scores for that data set.

Data expectations that can be set by the user:

- ❑ Column values should be unique.
- ❑ Column values can be null.
- ❑ Column values have high, medium, or low business impact.

ibi Data Quality's Profiler generates detailed results in JSON format and stores the detailed results in the file system. It also persists key statistics and metrics in the analytics repository. ibi Data Quality's Profiler cannot be customized by users.

Data Quality (DQ) Analysis

In ibi Data Quality, Rules are used to perform Data Quality (DQ) analysis. DQ analysis involves cleansing (removing junk or unexpected characters), standardization (reformatting to a consistent format), verification against a series of tests to ascertain that the values accurately represent the intended real-world entity, and data enrichment (imputing missing values or augmenting data from reference data sources).

In general, DQ analysis of an input data set results in the following output values:

- ❑ **output value.** Cleansed, fixed, or enriched value whenever issues are detected.
- ❑ **tag_value.** Tags to represent issues or reportable facts identified during the DQ analysis.
- ❑ **tag_category.** Final outcome of the analysis (VALID, INVALID, MISSING, or CLEANSED).
- ❑ **score.** A score that reflects the trustworthiness of the data at the column level and at the data set level.

Scoring

ibi Data Quality delivers two sets of scores: Profile Scores and Data Quality (DQ) Scores

In order to get accurate scores, it is recommended that users provide the source metadata and configure their expectations of the data.

Source Metadata

- Source Name.** Name of the data source.
- Source Type.** An indicator whether the data source is “INTERNAL” or “EXTERNAL” to the business entity.
- Application Name.** Name of the application within the source that is generating the data (for example, ERP-Prod).
- Industry.** Optional industry reference for the data (for example, Retail).
- Entity.** Optional business entity reference for the data (for example, Product).

Data expectations that can be set by the user:

- Column values should be unique.
- Column values can be null.
- Column values have high, medium, or low business impact.

Profile Score

ibi Data Quality’s Profiler compares the profiling results against data expectations set by the user to calculate individual scores for each data attribute and a summarized score for the complete data set.

Users are required to set up the following data expectations to get accurate profile scores:

- Column values should be unique.** User expects values in the data attribute to be unique.
- Column values can be null.** User does not always expect a value for the data attribute.
- Business impact.** An indicator of how the quality of data in this data attribute affects downstream business applications (HIGH, MEDIUM, or LOW).

Profiling stats for individual data attributes are scored against the first two expectations. Overall profile score is calculated as a weighted average score that applies business impact as weights to the individual data attribute's profiling scores.

Data Quality (DQ) Score

ibi Data Quality calculates DQ scores for each data attribute or a group of data attributes that is mapped to a Rule and also generates a summarized score for the complete data set. Users are required to set up the following expectation to get accurate DQ scores:

- ❑ **Business impact.** An indicator of how the quality of data in this data attribute affects downstream business applications (HIGH, MEDIUM, or LOW).

DQ stats for individual data attributes are scored against tag categories. The overall DQ score is calculated as a weighted average score that applies business impact as weights to the individual data attribute's DQ score.

Chapter 3

Deploying ibi™ Data Quality

This chapter describes how to deploy and run ibi™ Data Quality using Docker Compose and in a Kubernetes Cluster.

In this chapter:

- [Configurations](#)
- [Loqate API Access Keys](#)
- [Docker Deployment](#)
- [Kubernetes Deployment](#)
- [JDBC Driver Setup](#)
- [High Availability](#)
- [Updating Loqate License Keys](#)

Configurations

This section describes the configurations used by the ibi Data Quality containers and how to modify the default values if required.

Network Ports

In the Docker Compose deployment, the network ports listed in the following table are configured by default:

Container	Port
Valet User Interface	9800
WSO2 Identity Server	9801
Valet API Server	9803
Watchdog	9807
Postgres Database	9543

These ports can be reconfigured by following the steps in the corresponding sections that follow.

Note: After any changes are made to the port settings, you must rebuild the images and restart containers.

Valet User Interface

Edit *install/docker-compose.yaml* to change the nginx proxy configuration to use a different port.

```
services:
  dq-nginx:
    build:
      context: dq-nginx/dockerRoot
    container_name: dq-nginx
    ports:
      - "9800:80" # dq-valet-ui
```

So, to run the Valet User Interface on port 8900, edit as follows:

```
services:
  dq-nginx:
    build:
      context: dq-nginx/dockerRoot
    container_name: dq-nginx
    ports:
      - "8900:80" # dq-valet-ui
```

Then, edit the following properties in *install/dq-valet-ui/.env.development.tmpl* to use the new port:

```
REACT_APP_AUTH_POST_LOGOUT_REDIRECT_URI=https://$HOSTDOMAIN:9800/index.html
REACT_APP_AUTH_REDIRECT_URI=https://$HOSTDOMAIN:9800/callback
```

WSO2 Identity Server

Edit *install/docker-compose.yaml* to change the nginx proxy configuration to use a different port.

```
services:
  dq-nginx:
    build:
      context: dq-nginx/dockerRoot
    container_name: dq-nginx
    ports:
      - "9801:81" # dq-wso2
```

So, to run the WSO2 Identity Server on port 8901, edit as follows:

```

services:
  dq-nginx:
    build:
      context: dq-nginx/dockerRoot
    container_name: dq-nginx
    ports:
      ...
      - "8901:81" # dq-wso2

```

Also in *docker-compose.yaml*, change the following settings for the dq-grafana service to use the new port:

```

dq-grafana:
  ...
  Environment:
  ...
  - GF_AUTH_SIGNOUT_REDIRECT_URL=https://${HOSTDOMAIN}:9801/oidc/logout?
post_logout_redirect_uri=https%3A%2F%2F%7BHOSTDOMAIN%7D%3A9807
  ...
  - GF_AUTH_GENERIC_OAUTH_AUTH_URL=https://${HOSTDOMAIN}:9801/oauth2/
authorize

```

Then, edit the following properties in *install/dq-valet-ui/.env.development.tmpl* to use the new port:

```

REACT_APP_AUTH_AUTHORITY_URL=https://$HOSTDOMAIN:9801/oauth2/token

```

Then, edit *install/dq-wso2/dockerRoot/config/repository/conf/deployment.toml* to set the new port here:

```

[transport.https.properties]
proxyPort = 9801

```

To remove access to the WS02 console, edit *install/dq-nginx/dockerRoot/server-wso2.tmpl*. Find the following section of the file:

```
location / {
    # Reject requests with unsupported HTTP method
    resolver 127.0.0.1;

    if ($request_method !~ ^(GET|POST|PUT|DELETE|HEAD|OPTIONS|PATCH)$) {
        return 405;
    }

    proxy_pass https://${LOCATION_HOSTNAME}:${LOCATION_PORT}/;
    proxy_pass_header Server;
    proxy_set_header Host $server_name;
    proxy_set_header X-Forwarded-Host $host;
    proxy_set_header X-Forwarded-Server $host;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_read_timeout 5m;
    proxy_send_timeout 5m;
}
}
```

Edit as follows:

```
location / {
    # Reject requests with unsupported HTTP method
    resolver 127.0.0.1;

    if ($request_method !~ ^(GET|POST|PUT|DELETE|HEAD|OPTIONS|PATCH)$) {
        return 405;
    }

    proxy_pass https://${LOCATION_HOSTNAME}:${LOCATION_PORT}/;
    proxy_pass_header Server;
    proxy_set_header Host $server_name;
    proxy_set_header X-Forwarded-Host $host;
    proxy_set_header X-Forwarded-Server $host;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_read_timeout 5m;
    proxy_send_timeout 5m;
}

location /carbon/ {
    # Reject requests to admin console
    return 401;
}
}
```

Valet API Server

Edit *install/docker-compose.yml* to change the nginx proxy configuration to use a different port.

```

services:
  dq-nginx:
    build:
      context: dq-nginx/dockerRoot
    container_name: dq-nginx
    ports:
      ...
      - "9803:83" # dq-valet-services

```

So, to run the Valet API Server on port 8903, edit as follows:

```

services:
  dq-nginx:
    build:
      context: dq-nginx/dockerRoot
    container_name: dq-nginx
    ports:
      ...
      - "8903:83" # dq-valet-services

```

Then, edit the following properties in *install/dq-valet-ui/.env.development.tmpl* to use the new port.

```

# Base url for API requests
REACT_APP_API_BASE_URL=https://$HOSTDOMAIN:9803

```

Watchdog

Edit *install/docker-compose.yaml* to change the nginx proxy configuration to use a different port.

```

services:
  dq-nginx:
    build:
      context: dq-nginx/dockerRoot
    container_name: dq-nginx
    ports:
      ...
      - "9807:87" # dq-grafana

```

So, to run the Watchdog on port 8907, edit as follows:

```

services:
  dq-nginx:
    build:
      context: dq-nginx/dockerRoot
    container_name: dq-nginx
    ports:
      ...
      - "8907:87" # dq-grafana

```

Also in *docker-compose.yml*, change the following settings for the *dq-grafana* service to use the new port:

```
environment:
  ...
  - GF_SERVER_ROOT_URL=https://${HOSTDOMAIN}:9807
  ...
  - GF_AUTH_SINGNOUT_REDIRECT_URL=https://${HOSTDOMAIN}:9801/oidc/logout?
post_logout_redirect_uri=https%3A%2F%2F%3A9807
```

Then, edit the following properties in *install/dq-valet-ui/.env.development.tmpl* to use the new port:

```
# Watchdog url
REACT_APP_WATCHDOG_URL=https://$HOSTDOMAIN:9807
```

Postgres Database

The Postgres database is exposed on port 9543 for the convenience of users who might want to use it for reporting or analysis purposes. The application does not require that it be exposed and for security reasons, some users may wish not to expose it. To change the exposed port or remove it, edit *install/docker-compose.yml*.

```
dq-postgres:
  ...
  ports:
    - "9543:5432"
```

To change the port for the Postgres database to 8943, you would edit as follows:

```
dq-postgres:
  ...
  ports:
    - "8943:5432"
```

To remove the port, either delete those two lines, or comment them:

```
dq-postgres:
  ...
  #ports:
  # - "9543:5432"
```

User Interface

You can configure the following environment variables for User Interface by editing the *install/dq-valet-ui/.env.development.tmpl* file.

Note: Changes will not take effect until the *dq-valet-ui* image is rebuilt.

Timezone

```
# Setup timezone for user information display
REACT_APP_USER_TIMEZONE=America/New_York
```

Date Format

```
# Setup Date format for user display
REACT_APP_DATE_FORMAT="MM/DD/YY h:mm A"
```

File Upload Limit

```
# Setup File upload limit in bytes (allows users to upload upto max this
size)
REACT_APP_TOTAL_FILE_UPLOAD_LIMIT_BYTES="5368709120"
```

Values for Business Impact

```
# Setup numeric values for Business Impact used for Data Profile and DQ
scoring
REACT_APP_BUSINESS_IMPACT_HIGH_VALUE=10
REACT_APP_BUSINESS_IMPACT_MEDIUM_VALUE=5
REACT_APP_BUSINESS_IMPACT_LOW_VALUE=1
```

Profile Score Thresholds

```
# Setup Profile score thresholds (Note: between values is a warning)
# i.e if the 2 values are 50 and 74; 0-49=Bad; 50-74=Warning; 75-100=Good
REACT_APP_PROFILE_SCORE_BAD_WHEN_LT_VALUE=50
REACT_APP_PROFILE_SCORE_GOOD_WHEN_GT_VALUE=74
```

DQ Score Thresholds

```
# DQ scoring thresholds (Note: between values Get Docker | Docker
Documentation is a warning)
REACT_APP_DQ_SCORE_BAD_WHEN_LT_VALUE=50
REACT_APP_DQ_SCORE_GOOD_WHEN_GT_VALUE=74
```

Observation Limit for Correlation Analysis

```
# Setup max number of observations for correlation analysis (# of
Variables X Row Count)
REACT_APP_TOTAL_OBSERVATIONS_LIMIT="500000"
```

Loqate API Access Keys

To use the Loqate phone number and email address services, you must supply a valid API key. First, stop and remove the `dq-python-services` service:

```
$ docker compose stop dq-python-services
[+] Running 1/1
   □ Container dq-python-services   Stopped
$ docker compose rm dq-python-services
? Going to remove dq-python-services Yes
[+] Running 1/0
   □ Container dq-python-services   Removed
```

Then, edit `install/dq-python-services/config/loqate-service-config.json` to change values for `_email_api_key` and `_phone_api_key`:

```
{  "_email_api_url": "https://api.addressy.com/EmailValidation/Interactive/
Validate/v2.00/xmla.ws?",
  "_email_api_key": "AAAA-BBBB-1234-1234",
  "_email_api_timeout": "15000",
  "_phone_api_url": "https://api.addressy.com/PhoneNumberValidation/
Interactive/Validate/v2.20/xmla.ws?",
  "_phone_api_key": "AAAA-BBBB-1234-1234",
  "_phone_api_timeout": "15000"
}
```

Rebuild and restart the `dq-python-services` service:

```
docker compose up -d --build --force-recreate dq-python-services
```

Docker Deployment

This section provides prerequisites and describes how to deploy ibi Data Quality using Docker Compose.

Prerequisites

Before continuing, ensure your system has the following prerequisites:

- Docker Desktop or Docker Engine with Compose
- Loqate (Linux installation with data packs)
- WebFOCUS DSML Services Container Edition

Installing Docker Desktop or Docker Engine with Compose

Install Docker as documented for your environment. To install Docker Desktop, follow the instructions found on the following website:

<https://docs.docker.com/get-docker/>

Installing Loqate With Data Packs

Install and configure Loqate as documented in the *Omni-Gen® Address Cleansing (Loqate®) Installation and Configuration Guide*:

https://docs.tibco.com/pub/og/3.16.0/doc/pdf/TIB_og_3.16.0_address_cleansing_install.pdf

Installing WebFOCUS DSML Services Container Edition

To install WebFOCUS DSML Services Container Edition:

Download and Configure

1. Go to TIBCO eDelivery and download the *TIB_dsmlice_1.0.0_linux_x86_64.tar* file.
2. Untar the file to a local directory.
3. Go to the *TIB_dsmlice_1.0.2_linux_x86_64/dsm-build-context/metadata* directory and edit Dockerfile.
4. Locate the following COPY command section in the file:

```
ENV DSML_USER=dsm1
RUN adduser --disabled-password --gecos '' $DSML_USER

WORKDIR /metadata
COPY --chown=$DSML_USER:$DSML_USER metadata /metadata/run_service
```

5. Add the following line above the COPY command:

```
RUN chown -R $DSML_USER:$DSML_USER /metadata
```

The resulting file should look like the following:

```
#Create User
ENV DSML_USER=dsm1
RUN adduser --disabled-password --gecos '' $DSML_USER

WORKDIR /metadata
RUN chown -R $DSML_USER:$DSML_USER /metadata
COPY --chown=$DSML_USER:$DSML_USER metadata /metadata/run_service
```

6. Do **not** change anything else in the file and save your changes.

Build Metadata Service Image

1. Using the command line, change directory to the *TIB_dsmlice_1.0.2_linux_x86_64/dsm-build-context/metadata* directory.
2. Run the following command:

```
docker build --no-cache -t ibi2020/webfocus:metadata-9.0-1.0.0 .
```

Note: The dot (.) at the end is part of the command and must be included.

3. To verify that the image has been built, execute the following command:

```
$ docker image ls ibi2020/webfocus
REPOSITORY          TAG                IMAGE ID           CREATED           SIZE
ibi2020/webfocus   metadata-9.0-1.0.0  fd19b79b06d8     3 weeks ago     295MB
```

Installing ibi Data Quality

To install ibi Data Quality:

1. Download and unzip the *ibi_tdq_5.2.0_container.zip* file.
2. Edit the *install/.env* file.
 - a. Change the *HOSTDOMAIN*. This is required if you intend to access ibi Data Quality from a machine other than the one where it is installed.
 - b. Change the location of your Loqate Linux installation. This is required if you intend to use the Loqate address cleansing services.

For example, if you extracted a Linux installation of Loqate in the *C:\Loqate_Linux* folder, then change the value of *LOQATE_HOME* in the *.env* file to:

```
LOQATE_HOME=C:/loqate_linux
```

- c. By default, ibi Data Quality starts three replicas of the *dq-profiler* container. This improves performance and scalability when profiling large data sets, at the cost of increased resource use. If you do not expect to profile large data sets, then you can limit resource consumption by setting *PROFILER_SCALE* to 1 or 2.

Note: If you are using Docker Compose version 2.x and require profiler scaling, then you must edit the *docker-compose.yaml* file.

By default, the *docker-compose.yaml* file sets the scaling properties used by Docker Compose version 1.x:

```
dq-profiler:
  build:
    ...
    scale: ${PROFILER_SCALE}
# comment scale and use deploy section instead for compose
# version 2.x
#   deploy:
#     mode: replicated
#     replicas: ${PROFILER_SCALE}
```

For Docker Compose version 2.x, edit this section in the *docker-compose.yaml* file as follows:

```

dq-profiler:
  build:
    ...
  #   scale: ${PROFILER_SCALE}
  #   comment scale and use deploy section instead for compose
  #   version 2.x
  deploy:
    mode: replicated
    replicas: ${PROFILER_SCALE}

```

d. Other settings in the `.env` file should not be changed.

3. Open Windows PowerShell or any Linux shell.
4. Change your directory to the folder where you extracted the `ibi_tdq_5.2.0_container.zip` file.
5. Change your directory to the `/install` folder.
6. Run the following command:

```
docker-compose up --build
```

7. Wait to confirm that WS02 Identity Server is started and running.
8. Open the WS02 console at <https://localhost:9801> and accept the self-signed certificate.

Note: ibi Data Quality is packaged with a self-signed certificate for HTTPS. It is recommended that you replace this with your own certificate. If you have a signed certificate and key, replace `tdq-cert.crt` and `tdq-cert.key` in the `install/dq-nginx/dockerRoot` folder with your own files, using those file names. If OpenSSL is available, then you can generate your own self-signed certificate by editing `tdq-nginx-cert.conf` and executing the command found in the `gen-cert-script.txt` file.

9. Enter the following URL in your browser to access the ibi Data Quality valet console:

```
https://localhost:9800/
```

The default login credentials (user ID and password) are:

- User ID: **dqadmin**
- Password: **dqadmin**

Note: It is recommended that you change the default login credentials in the User Management Console. For more information, see [Managing Existing Users](#) on page 113.

10. To restart ibi Data Quality after the first time, remove the `-build` switch from the command line. Simply run:

```
docker-compose up
```

Kubernetes Deployment

This section provides prerequisites and describes how to deploy ibi Data Quality in a Kubernetes cluster.

Kubernetes Overview

Kubernetes® (K8s) is an open-source system for running, managing, and orchestrating containerized applications in a cluster of servers (known as a Kubernetes cluster). Kubernetes clusters can run in any cloud environment (e.g., private, public, hybrid) or on-premises.

A Kubernetes cluster consists of a set of worker machines, called nodes, that run containerized applications. Every cluster has at least one worker node.

The size of a Kubernetes cluster (determined by the number of nodes) depends on the application workload(s) that will be running. For example, each node can represent an 8 core / 64 GB RAM system. Pods are the basic objects in a Kubernetes cluster, which consists of one or more software containers. The worker node(s) host the Pods that are the components of the application workload. Usually, only one container runs in a Pod. However, multiple containers can be run in a Pod if needed (depending on specific environment requirements). If one Pod fails, Kubernetes can automatically replace that Pod with a new instance.

Key benefits of Kubernetes include automatic scalability, efficient load balancing, high availability, failover/fault tolerance, and deploying updates across your environment without any disruptions to users.

Hardware Requirements

ibi Data Quality is a collection of microservices running as separate containers in a private cloud environment such as Kubernetes. Your Kubernetes cluster must have at least the following:

With Horizontal Pod Autoscaler (HPA):

- 42.5 vCPUs
- 33.7 GB of memory

Without Horizontal Pod Autoscaler (HPA):

- 22.5 vCPUs
- 23.7 GB of memory

Both cases would require 500 GB of persistent volume to be available.

The following deployment resource spec table lists the recommended CPU, memory, and volume requirements for each of the containers.

Container Name	Replicas		CPU (Per Replica)		Memory (Per Replica)	
	Min	Max	Requests	Limits	Requests	Limits
dq-address-server	1	1	2000m	2000m	6Gi	8Gi
dq-butler	1	2	1000m	3000m	2Gi	3Gi
dq-dqs	1	1	500m	1000m	1Gi	1Gi
dq-dsml-classifier	1	1	2000m	5000m	2Gi	4Gi
dq-grafana	1	1	1000m	2000m	1Gi	3Gi
dq-ksource	1	1	1000m	2000m	1Gi	2Gi
dq-postgres	1	1	2000m	4000m	2Gi	8Gi
dq-profiler	4	8	4000m	4000m	2Gi	4Gi
dq-python-services	3	6	4000m	4000m	2Gi	8Gi
dq-valet-services	1	3	500m	1000m	1Gi	2Gi
dq-valet-ui	1	1	500m	1000m	1Gi	2Gi
dq-wso2	1	1	2000m	4000m	2Gi	3Gi
dq-redis	1	1	500m	2000m	200Mi	1Gi
dq-log-viewer	1	1	1000m	1000m	0.5Gi	1Gi

Installation

To install ibi Data Quality:

1. Download and unzip the *ibi_tdq_5.2.0_container.zip* file.
2. Download and extract a Linux installation of Loqate into the NFS server volume where the instance of ibi Data Quality that is being deployed to a Kubernetes cluster will be utilized:

```
<NFS_Volume_Mount_Path>/loqate
```

Building Docker Images for Kubernetes

Before you deploy ibi Data Quality in a Kubernetes cluster, you must first create Docker images for the ibi Data Quality components.

1. Open Windows PowerShell or any Linux shell.
2. Change your directory to the folder where you extracted the *ibi_tdq_5.2.0_container.zip* file.
3. Change your directory to the *install/dq-k8s/scripts* folder.
4. Deploy the NGINX Ingress controller using the following command:

```
$ kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v1.3.1/deploy/static/provider/baremetal/deploy.yaml
```

This command will deploy Kubernetes objects into the *ingress-nginx* namespace.

After deployment, run the following command to extract the mapped controller HTTPS port of the cluster, which will be required in the next step:

```
$ kubectl -n ingress-nginx get service ingress-nginx-controller | grep -oP '(?<=443:).*?(?=/TCP)'
```

5. Run the *init_kubernetes_configuration.sh* script to update the domain name, the ingress node port, and container image registry URL:

```
$ ./init_kubernetes_configuration.sh <DQ_CLUSTER_DOMAIN>  
<DQ_INGRESS_CONTROLLER_HTTPS_NODEPORT>
```

where:

```
<DQ_CLUSTER_DOMAIN>
```

Is the cluster domain intended to be used for the HTTPS URL.

```
<DQ_INGRESS_CONTROLLER_HTTPS_NODEPORT>
```

Must be set to the NGINX Ingress controller's HTTPS port (from step 4).

6. Change your directory to the folder where you extracted the *ibi_tdq_5.2.0_container.zip* file to the **install/** folder.
7. Run the following command:

```
$ docker-compose --build -f docker-compose-kubernetes.yaml build
```

- Build the DSML Services Container Edition images.

For more information, see [Installing WebFOCUS DSML Services Container Edition](#) on page 41.

- (Optional) Push the Docker images to the image registry that can be accessed by the Kubernetes cluster.

Deploying ibi Data Quality to a Kubernetes Cluster

ibi Data Quality deployed in a Kubernetes cluster requires a shared PersistentVolume (PV) that is set to the *ReadWriteMany* access mode. In the following procedures, the NFS-volume is used as an example.

Prerequisites

- NFS mount must be made available to the Kubernetes nodes.
- OpenSSL tool must be installed.
- Helm package management software installed.

Deploying Infrastructure Components

This section describes how to deploy ibi Data Quality infrastructure components.

- Change your directory to the folder where you extracted the *ibi_tdq_5.2.0_container.zip* file to the *install/dq-k8s/helm/* folder.
- Create a namespace (for example, *idq-ns*) using the following command:

```
$ kubectl create namespace idq-ns
```

This will create the namespace object *tdq-ns* in the Kubernetes cluster.

- Create a secret for ingresses.

Either create a self-signed certificate using the following command, or use a trusted certificate key and certificate to create a Kubernetes secret.

The following command will create a self-signed certificate for IP 10.10.11.12 and the output files will be *dq.key* and *dq.cert*:

```
$ openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout dq.key -
out dq.cert -subj "/CN=10.10.11.12/O=customer.org/subjectAltName=DNS.1=*.
10.10.11.12"
```

The following command will insert the `dq.key` and `dq.cert` files created in the previous command in the Kubernetes secret and named `dq-tls-secret` in the `idq-ns` namespace:

```
$ kubectl create secret tls dq-tls-secret --key dq.key --cert dq.cert -n idq-ns
```

4. In the `dq/values.yaml` file, replace the following string patterns with the appropriate values:

❑ **<CLUSTER_DOMAIN>**: The cluster domain intended to be used for the HTTPS URL.

❑ **<INGRESS_NODEPORT>**: The Ingress Controller's HTTPS NodePort.

❑ **<IMAGE_REGISTRY>**: The Docker image registry URL.

❑ **<NFS_SERVER_IP>**: The NFS server IP used for Persistent Volume.

❑ **<NFS_SERVER_MOUNT_PATH>**: The exported path on the NFS server.

5. (Optional) For users who require the Horizontal Pod Autoscaler (HPA) feature, set `autoscaling.enabled` to `true` in the `dq/values.yaml` file.

HorizontalPodAutoscaler automatically scales the ibi Data Quality components to match demand based on observed metrics, such as average CPU utilization and average memory utilization.

All the HPA component targets are set to be 50% of the requested CPU and memory usage. Please check the minimum and maximum replica number for each component, as noted in the deployment resource spec table in [Hardware Requirements](#) on page 44.

6. Install the ibi Data Quality Helm chart.

The following command will install the ibi Data Quality Helm chart to the namespace with the `idq` release name:

```
$ helm upgrade --install -n idq-ns --create-namespace idq ./dq
```

Verifying and Testing

This section describes how to verify and test ibi Data Quality in a Kubernetes cluster by confirming all required services and components are running.

Confirming Services and Components are Running

To confirm ibi Data Quality services and components are running:

1. Confirm that the WSO2 Identity Server is started and running.

The following is a sample command you can use to check the status of the `dq-wso2` pod:

```
$ kubectl -n idq-ns get pod -l app.kubernetes.io/component=dq-wso2
```



```
$ kubectl -n idq-ns describe pod -l app.kubernetes.io/component=dq-wso2
```

2. Wait for approximately 15 minutes to confirm that the `dq-valet-ui` service is started and running.

The following is a sample command you can use to check the status of the `dq-valet-ui` pod:

```
$ kubectl -n idq-ns get pod -l app.kubernetes.io/components=dq-valet-ui
```

```
$ kubectl -n idq-ns describe pod -l app.kubernetes.io/components=dq-valet-ui
```

3. Verify that all other ibi Data Quality pods in the deployment are started and running.

The following is a sample command you can use to check the status of each ibi Data Quality pod:

```
$ kubectl -n idq-ns get pod
```

The following is a sample command you can use to check the status of each ibi Data Quality service:

```
$ kubectl -n idq-ns get svc
```

The following is a sample command you can use to check the status of each ibi Data Quality ingress:

```
$ kubectl -n idq-ns get ingress
```

Logging in to the ibi Data Quality Console

If a self-signed or untrusted certificate is used, there are three certificates that must be accepted by the browser before logging into the ibi Data Quality console.

1. Visit the WSO2 console at the following URL (substituting values for `<DQ_CLUSTER_DOMAIN>` and `<DQ_INGRESS_CONTROLLER_HTTPS_NODEPORT>`) and accept the self-signed certificate:

```
https://dq-wso2.<DQ_CLUSTER_DOMAIN>:<DQ_INGRESS_CONTROLLER_HTTPS_NODEPORT>
```

2. Visit the following URL in your browser (substituting values for `<DQ_CLUSTER_DOMAIN>` and `<DQ_INGRESS_CONTROLLER_HTTPS_NODEPORT>`) and accept the certificate:

```
https://dq-valet-services.<DQ_CLUSTER_DOMAIN>:<DQ_INGRESS_CONTROLLER_HTTPS_NODEPORT>/api/v1/about
```

3. Visit the following URL in your browser (substituting values for `<DQ_CLUSTER_DOMAIN>` and `<DQ_INGRESS_CONTROLLER_HTTPS_NODEPORT>`) to access the ibi Data Quality Console:

```
https://dq-valet-  
ui.<DQ_CLUSTER_DOMAIN>:<DQ_INGRESS_CONTROLLER_HTTPS_NODEPORT>
```

The default login credentials (user ID and password) are:

User ID: **dqadmin**

Password: **dqadmin**

4. Visit the following URL in your browser (substituting values for `<DQ_CLUSTER_DOMAIN>` and `<DQ_INGRESS_CONTROLLER_HTTPS_NODEPORT>`) to access the log files for ibi Data Quality:

```
https://dq-log-  
viewer.<DQ_CLUSTER_DOMAIN>:<DQ_INGRESS_CONTROLLER_HTTPS_NODEPORT>
```

The default login credentials (user ID and password) are:

User ID: **dqadmin**

Password: **dqadmin**

JDBC Driver Setup

JDBC drivers are required to connect and read data from JDBC data sources. For more information, see [Uploading Data From a JDBC Data Source](#) on page 58. JDBC drivers are configured in the Valet Services container.

New Valet Services Container

For a new installation before you run the Docker Compose, you should copy all your JDBC driver .jar files to the following location:

```
install\dq-valet-services\dockerRoot\lib
```

Existing Valet Services Container

If you are already running the Valet Services container, you can copy a new .jar file into that container using the Docker copy (cp) command. For example:

```
docker cp sqljdbc42.jar dq-valet-services:/dq-valet-services/lib/  
sqljdbc42.jar
```

After you copy the new .jar file, restart the `dq-valet-services` container.

High Availability

For Docker Compose services, you can configure restart policies to improve the availability of ibi Data Quality or use third-party products to implement clustering of Docker servers. For more information on maintaining a swarm of Docker engines, refer to:

https://docs.docker.com/engine/swarm/admin_guide/

However, for environments where high availability is required, we recommend deploying ibi Data Quality in a Kubernetes cluster. For more information, see [Kubernetes Deployment](#) on page 44.

Deploying ibi Data Quality in a Kubernetes cluster provides high availability on the deployment level within the Kubernetes cluster. Any stopped/crashed ibi Data Quality services in the Kubernetes cluster will be restarted automatically using a Kubernetes internal mechanism. All critical services are configured to run with multiple replicas to improve the reliability of those services.

It is recommended that Kubernetes administrators implement high availability on the cluster level to protect the cluster's infrastructure, data, and metadata. For more information, refer to:

<https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/high-availability/>

Updating Loqate License Keys

The following are the high-level steps required to update the Loqate license key on your ibi Data Quality address service container.

1. Create a ticket through Tech Support to obtain the new license key for your Loqate address service subscription.
2. Login to the terminal for dq-address-container.

You can do this in Docker Desktop, as shown in the following image.



You can also execute from the command line as follows:

```
docker exec -i <container name or container id> sh
```

3. Change your current directory to `/software/loqate`.

4. Run the *InstallManager.sh* script.

```
/dq-address-server # cd /software/loqate
/software/loqate # ./InstallManager.sh
Native library STLPort failed to load, ignore this if not using solaris OS.
java.lang.UnsatisfiedLinkError: no stlport in java.library.path
Welcome to the Installation Manager - version 15.0.0
Running using local API version 2.42.1.16021-4d37ba4
Does your network communication pass through a proxy server (y/n)?
n
Please enter the path to the data installation folder
/software/loqate/data
A valid current installation was found. Proceeding with a data update.

Please select from below:
Enter 1 if you have a license key
Enter 2 if you have a license pack
Enter 3 to exit
1
Please enter the license key:
XXX-XXX-XXX-XXX

Contacting license server to validate the license key.
++++
Details of your current license
-----
-----
Product                                Expiry Date
-----
Knowledge Base Common                  2023-12-31
Worldwide Geocode Dataset              2023-12-31
USA Verify Dataset CASS                2023-12-31
USA Enhance CENSUS Dataset             2023-12-31
CASS Library                           2023-12-31
Worldwide Reverse Geocode Dataset      2023-12-31
UK Enhance DPS+UDPRN Dataset           2023-12-31
Worldwide Verify Dataset               2023-12-31
-----
-----

Please select from below:
Enter 1 if you want to download the data packs but not install
Enter 2 if you want to download and install the data packs
Enter 3 if you want to install using your locally available data packs
Enter 4 to exit
```

5. Provide the path to the data installation directory:
`/software/loqate/data`
6. Enter 1 to continue.
7. Enter your Loqate license key.
8. Enter 4 to exit.

Navigating and Using ibi Data Quality

This chapter describes how to navigate and use the core facilities and tools provided by ibi Data Quality.

ibi Data Quality enables you to:

- Upload CSV data sets.
- Sample the uploaded data.
- Create data profiles based on a technical analysis of the data.
- Perform domain analysis using pre-configured rules that verify, cleanse, and enrich the data.
- Download, review, and share your data profiles and analysis results.

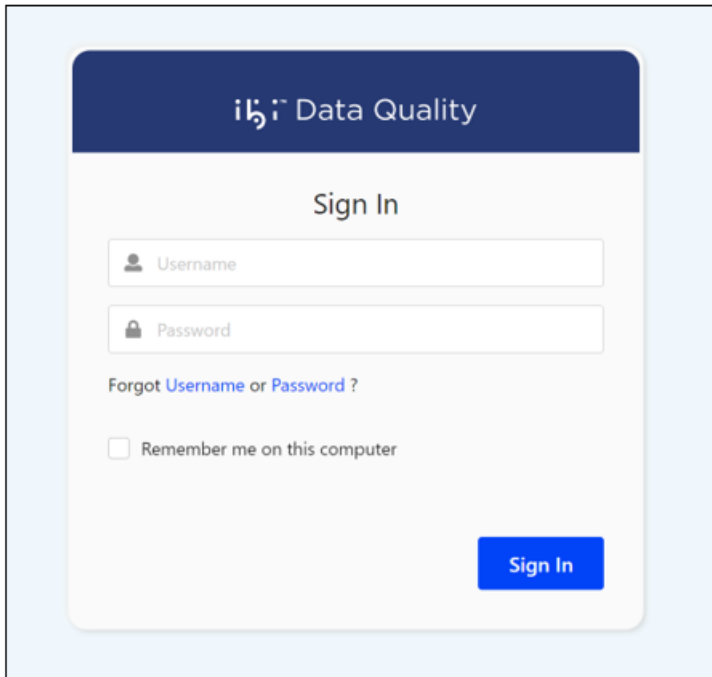
In this chapter:

- [Signing In to ibi Data Quality](#)
 - [Uploading Data](#)
 - [Profiling Data](#)
 - [Analyzing Data Quality](#)
-

Signing In to ibi Data Quality

Signing in to ibi Data Quality requires a valid user name and password, as configured by your ibi Data Quality administrator.

In a supported web browser, enter your ibi Data Quality username and password in the Sign In screen, as shown in the following image.



The image shows a sign-in interface for 'ibi Data Quality'. At the top, there is a dark blue header with the 'ibi Data Quality' logo. Below the header, the text 'Sign In' is centered. There are two input fields: 'Username' with a person icon and 'Password' with a lock icon. Below the password field is a link 'Forgot Username or Password?'. There is a checkbox labeled 'Remember me on this computer'. At the bottom right is a blue 'Sign In' button.

Supported Web Browsers

ibi Data Quality web client supports the following web browsers:

Browser	Supported Version
Chrome	Latest
Safari	Latest
Edge	Latest
Firefox	Latest

Supported Resolution and Display Layout

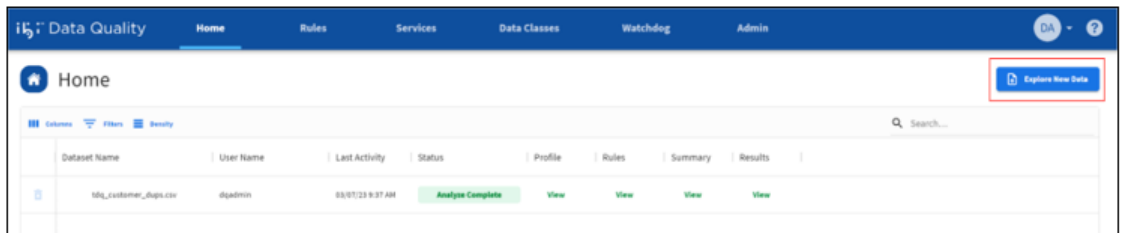
ibi Data Quality web client supports the following display resolution and screen and layout ratio to avoid screen real estate issues on Windows machines:

Display Resolution	Scale and Layout
1920 x 1080	125% or less

Uploading Data

Once signed in, the ibi Data Quality UI offers different navigational views to get you started.

To upload data, click *Explore New Data* in the upper-right corner of the Home Page, as shown in the following image.



The Explore New Data dialog opens, as shown in the following image.

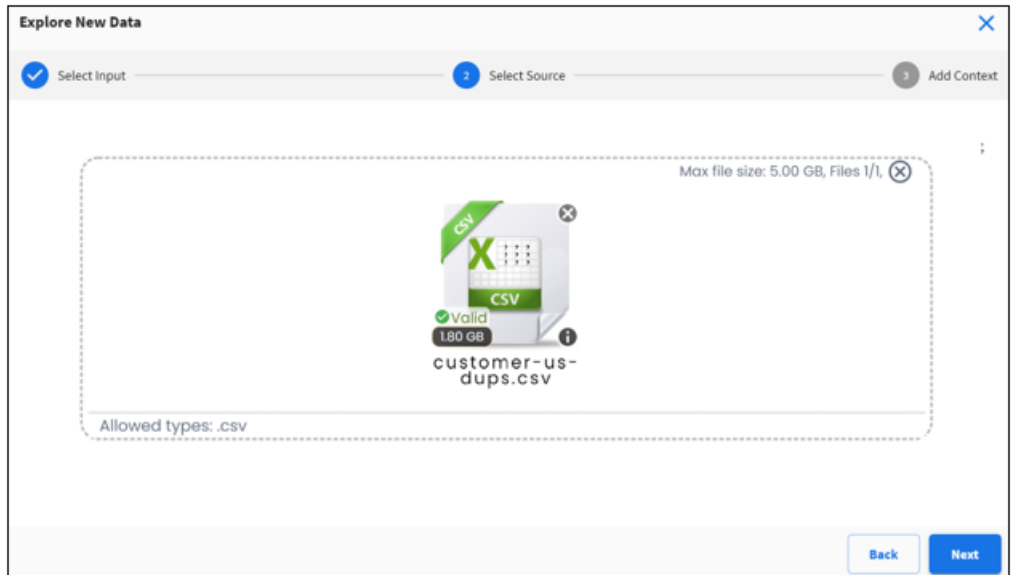


You can upload data from a text file or JDBC data source (PostgreSQL, SQL Server, etc.). Click *File Upload* or *JDBC datasource*.

Uploading Data From a Text File

To upload data from a text file (for example, .csv):

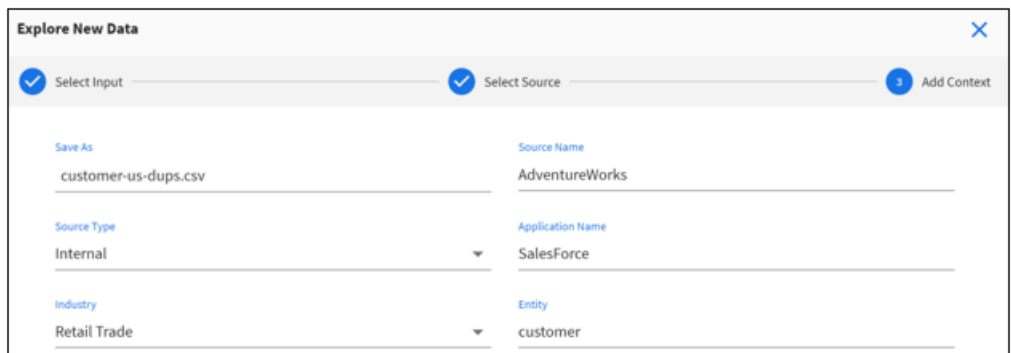
1. Drag an input text file to the upload window or browse your local file system and select an input text file.



Note: Text files up to 5GB in size are supported.

2. Click Next.

The Add Context pane opens, as shown in the following image.



3. Add business context for your input data.

Tip: Setting up the right business context helps users filter DQ statistics and metrics in the ibi Data Quality Watchdog App.

Property	Description
Save As	Data set name if different from file name.
Source Name	Name of the data source.
Source Type	Is the data source considered internal or external to the organization.
Application Name	Name of the application that generated the input data.
Industry	Select an industry represented by the data.
Entity	Name of the business entity the data represents (e.g., customer, partner, supplier, office).

4. Provide parsing options for your input data.

File Character Set	File Has Header
UTF-8	True
File Delimiter	Quote Character
Comma (,)	Quotation Mark ("")

Property	Description
File character set	Character encoding, supported types are UTF-8, UTF-16, or ISO-8859-1.
File has header	True if file has a header column, False otherwise.
File Delimiter	Field delimiter, supported delimiters are Comma (,), Pipe(), Semicolon (;), Space, or Tab.

Property	Description
Quote Character	Define the enclosing character if text within a field includes the delimiter character. Supported characters are double quote (") or grave accent (`).

Uploading Data From a JDBC Data Source

This section describes how to upload data from a JDBC data source.

New JDBC Connection

To upload data from a JDBC data source using a new JDBC connection:

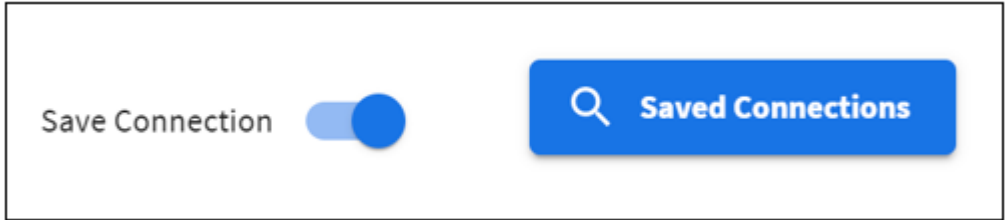
1. Provide the SQL select statement to read data from a table or a view.

The screenshot shows the 'Explore New Data' interface with the following details:

- Select Statement:** `select * from public.tdq_customer`
- JDBC Info:**
 - User Name:** pgadmin
 - Password:** [Masked]
 - JDBC Driver:** Postgresql
 - Host Name / IP:** db
 - Port:** 5432
 - Database Name:** pgtest
- JDBC Connection String:** `jdbc:postgresql://db:5432/pgtest`
- Buttons:** Save Connection (toggle), Save Connections (button), Back, Next.

2. Provide the source database user name and password.
3. Select a JDBC driver from the list of available drivers.
4. Provide the hostname, port, and database name.
5. Provide any optional JDBC connection parameters as required.

6. Toggle *Save Connection* if you intend to save and reuse this JDBC connection.



7. Click *Next* to continue with the rest of the data upload steps.

Existing JDBC Connection

To upload data from a JDBC data source using an existing JDBC connection:

1. If you want to reuse an existing JDBC connection, click *Saved Connections* and select from the list of available JDBC connection strings.



2. Provide the SQL select statement to read data from a table or a view.
3. Click *Next* to continue with the rest of the data upload steps.

Sample JDBC Connection Strings

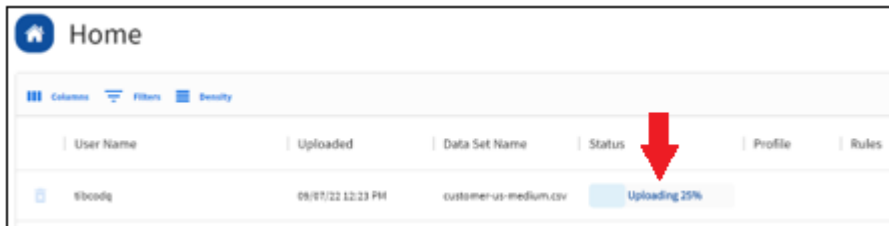
The following table provides sample JDBC connection strings for reference purposes.

Database	Format	Example
SQL Server	<code>jdbc:sqlserver://[serverName[\\instanceName][:portNumber]]</code>	<code>jdbc:sqlserver://localhost:1433;databaseName=AdventureWorks</code>
PostgreSQL	<code>jdbc:postgresql://host:port/database</code>	<code>jdbc:postgresql://localhost:5432/AdventureWorks</code>

Database	Format	Example
Oracle	<code>jdbc:oracle:thin:@host:port:database</code>	<code>jdbc:oracle:thin:@localhost:1521:AdventureWorks</code>
MySQL	<code>jdbc:mysql://localhost:port/database</code>	<code>jdbc:mysql://localhost:3306/AdventureWorks</code>

Tracking Upload Progress

The upload progress bar displays the current upload status, as shown in the following image.

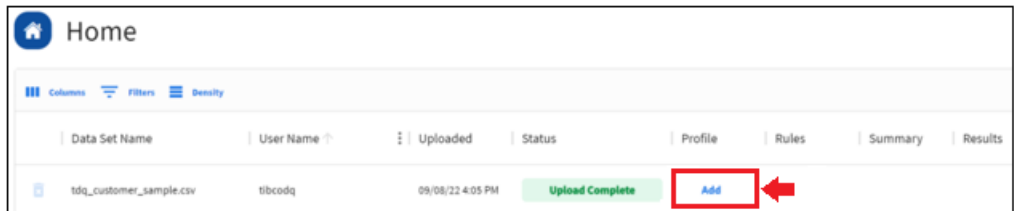


Profiling Data

Data Profiling analysis generates insights about the technical characteristics of the input data for each of the variables selected by the user.

Create a Data Profile

Click *Add* under the Profile column, as shown in the following image.



Select Variables

Select the variables to include in the data profile.

Tip: Selecting the checkbox in the upper-left corner selects all variables.

Set Data Expectations

For each variable, configure the following three data expectations:

1. Select *Unique* if the values in that column are always expected to be unique.
2. Deselect *Allow Nulls* if values in that column can never be Null.
3. Use the slider to indicate if the variable has *HIGH*, *MEDIUM*, or *LOW* impact on business outcomes.

Tip: These three settings are used to calculate the Profile Scores.



Track Profiling Progress

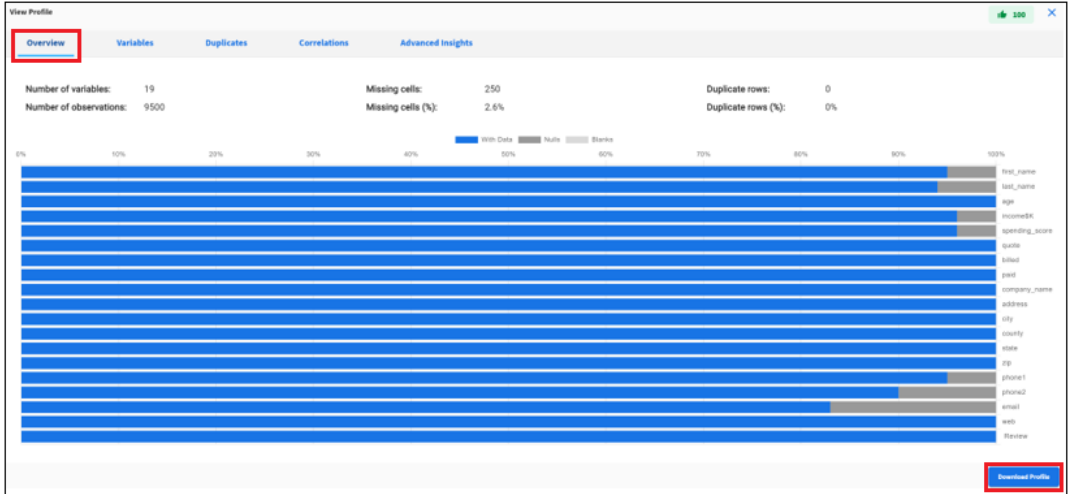
The profile progress bar displays the current processing status, as shown in the following image.



Viewing and Downloading a Profile

This report provides a high-level overview of the overall data profile.

Click the *Overview* tab in the View Profile dialog, as shown in the following image.



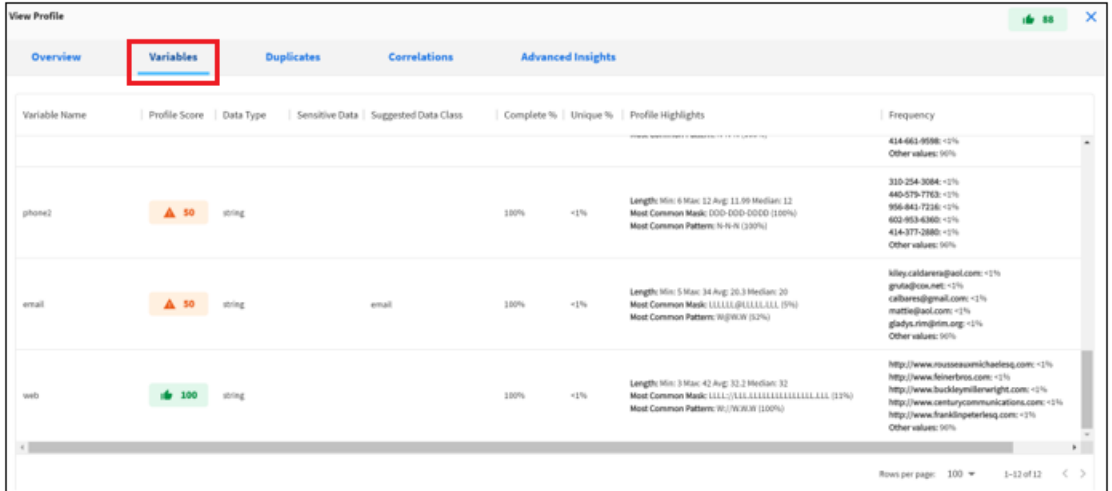
Click *Download Profile* in the lower-right corner of the *Overview* tab to download the profiling results in JSON format. This profile output is also available in the results .zip file.

Data Facts	Description
Number of variables	Number of columns in the data set.
Number of observations	Total number of data values in the data set, row times columns.
Missing Cells	Number of values that are empty or missing.
Missing Cells (%)	Percentage of values that are empty or missing.
Duplicate Rows	Total number of duplicate rows.
Duplicate rows (%)	Percentage of rows that are duplicate.
Overall Profile Score	Profile score for the data set in the range 0 to 100 (100 is the best).

Variables

This report provides a detailed view of each variable included in the profile.

Click the *Variables* tab in the View Profile dialog, as shown in the following image.



Variable Facts	Description
Variable Name	Name of the column.
Profile Score	Profile score for the variable in the range 0 to 100 (100 is the best).
Data Type	Data type identified by the profiler.
Sensitive Data	A flag to suggest if the variable contains sensitive data.
Data Class	The data class identifier by the profiler.
Complete %	Percentage of values that are not null.
Unique %	Percentage of values that are unique.
Highlights - Length	Min, max, average and median length of values in the variable.

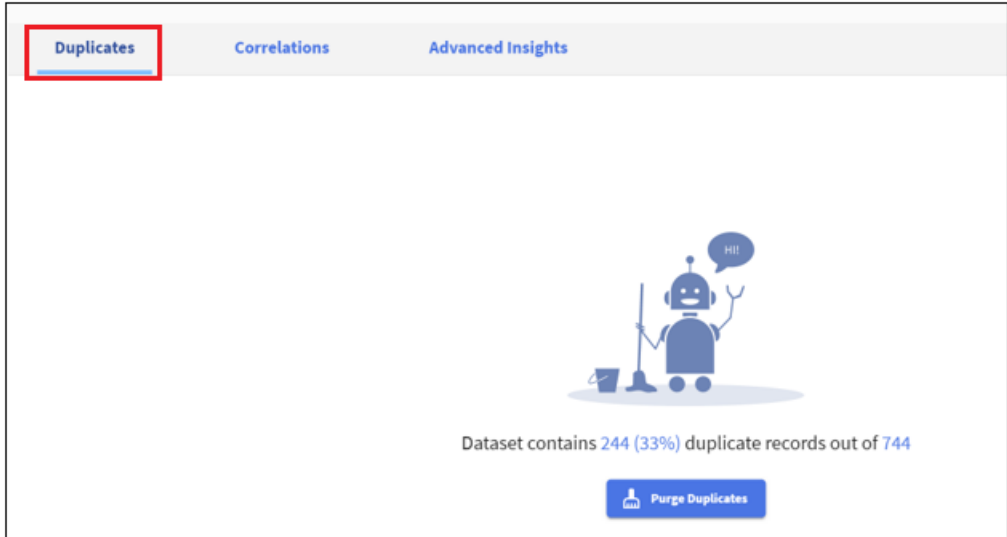
Variable Facts	Description
Highlights - Mask	<p>Most common mask discovered by the profiler.</p> <ul style="list-style-type: none"> <input type="checkbox"/> Any digit between 0 and 9 is represented as "D". <input type="checkbox"/> Any character between Aa to Zz is represented as "L". <input type="checkbox"/> All other characters are represented as actual values.
Highlights - Pattern	<p>Most common pattern discovered by the profiler.</p> <ul style="list-style-type: none"> <input type="checkbox"/> A single digit between 0 and 9 is represented as "D" (digit). <input type="checkbox"/> A single character between Aa to Zz is represented as "L" (letter). <input type="checkbox"/> A sequence of digits is represented as "N" (number). <input type="checkbox"/> A sequence of characters between Aa to Zz is represented as "W" (word). <input type="checkbox"/> All other characters are represented as actual values.
Frequency	Most common values and their frequency.

Note: Users can create definitions for classifying custom data using the masks and patterns discovered during profiling. For more information, see [Managing Data Classes](#) on page 96.

Duplicates

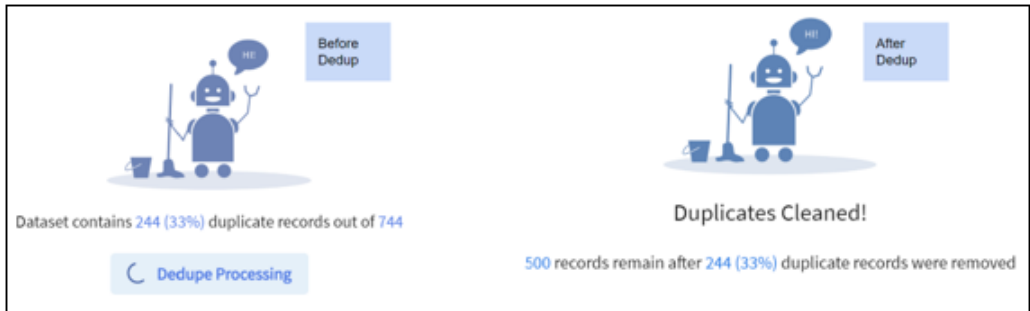
This report shows all the duplicate rows in the data. These are true duplicate rows where a row of data values exactly matches with another row in the input data.

Click the *Duplicates* tab in the View Profile dialog, as shown in the following image.



Purge Duplicates

Click *Purge Duplicates* to remove duplicate rows from input data.



Correlations

This is an optional profiling feature to perform bivariate analysis that measures the strength of association between two variables and the direction of the relationship.

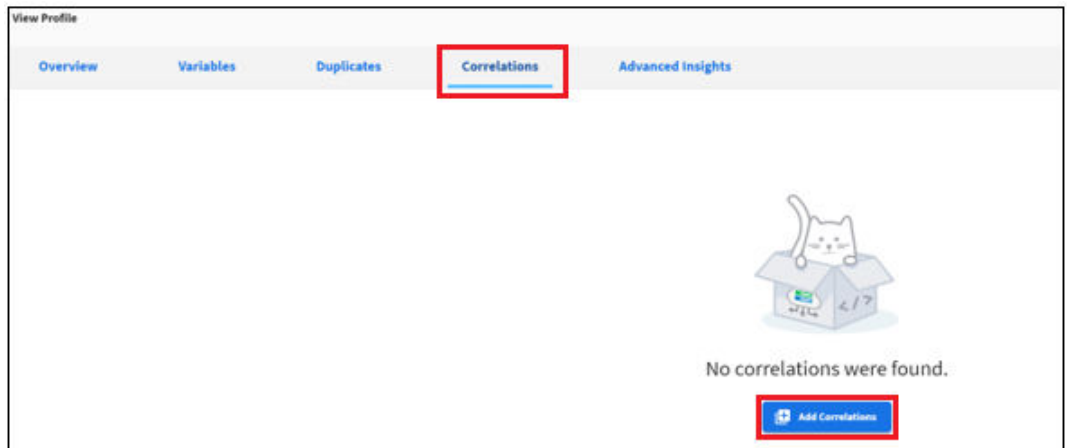
In terms of the strength of the relationship, the value of the correlation coefficient varies between +1 and -1. A value of ± 1 indicates a perfect degree of association between the two variables. As the correlation coefficient value goes towards 0, the relationship between the two variables will be weaker. The direction of the relationship is indicated by the sign of the coefficient. A plus (+) sign indicates a positive relationship and a minus (-) sign indicates a negative relationship.

Value	Correlation Between x and y
equal to 1	Perfect positive linear relationship.
greater than 0	Positive correlation.
equal to 0	No linear relationship.
less than 0	Negative correlation.
equal to -1	Perfect negative linear relationship.

Adding Correlations

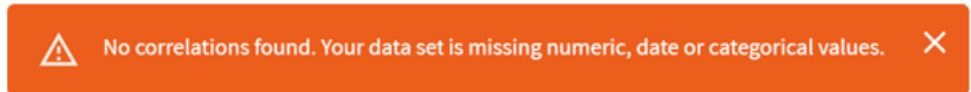
To add correlations:

1. Click the *Correlations* tab in the View Profile dialog, and then click *Add Correlations* as shown in the following image.



2. Choose correlation settings.
 - a. Select variables for correlation analysis.

Note: Correlations can be generated for variables with numeric, date or categorical values. If the selected variables do not meet this criteria, an error message displays in the upper-right corner of the screen, as shown in the following image.



- b. Select one or more correlation types.
For more information, see [Supported Correlations](#) on page 68.
 - c. Select the row count.

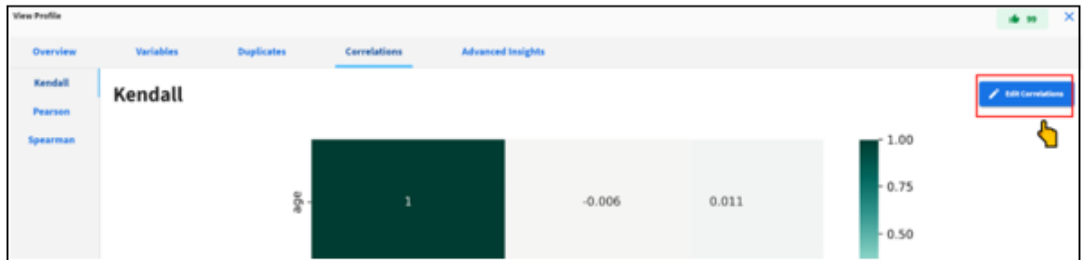
 A screenshot of the "Correlation Settings" interface. It features three sections:

- Choose Variable(s):** Three buttons labeled "age", "income\$K", and "spending_score".
- Choose Correlation(s):** Three buttons labeled "Kendall", "Pearson", and "Spearman".
- Row Count:** A text input field containing the number "500".
- Total Observations:** A text input field containing the number "1500".

Note: The max number of observations that can be uploaded for correlations is 500,000. Adjust the number of variables and the row count to reduce the total number of observations if it exceeds the max limit.

 A red-bordered box containing a warning message. At the top, it says "Total Observations" followed by a red line and the number "3000000". Below this, in red text, it says "(# of Variables X Row Count) cannot exceed 500000".

3. Click *Edit Correlations* to change the correlation settings and to rerun the correlation analysis, as shown in the following image.



Supported Correlations

This section describes the supported correlations that are available.

Pearson Correlation

The Pearson (product-moment) correlation coefficient is a measure of the linear relationship between two features. It is the ratio of the covariance of x and y to the product of their standard deviations. It is often denoted with the letter r and called Pearson's r .

This value is mathematically expressed by the following equation:

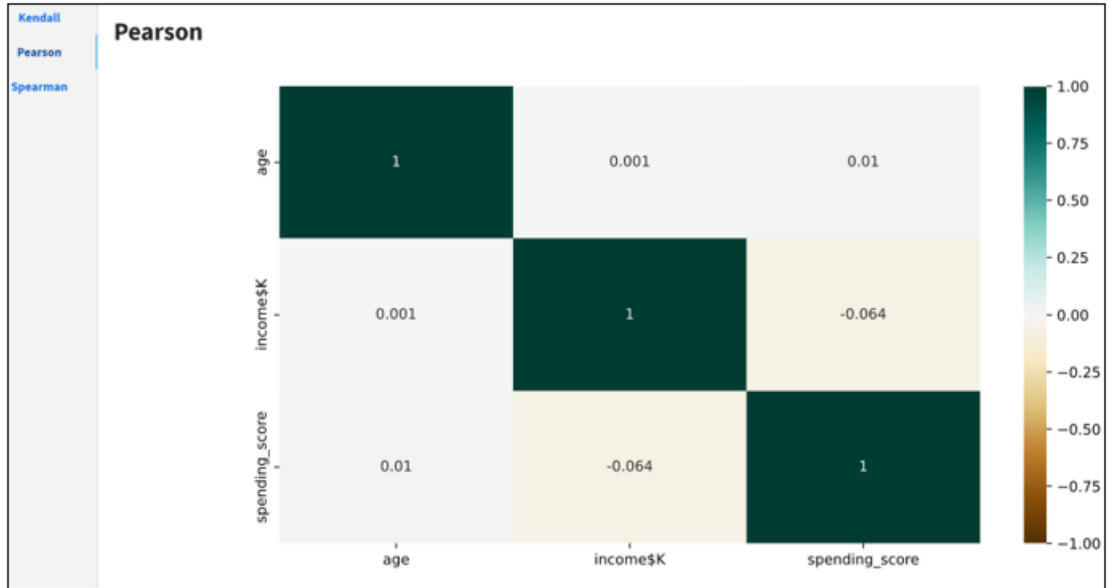
$$r = \frac{\sum(X - \bar{X})(Y - \bar{Y})}{\sqrt{\sum(X - \bar{X})^2 \cdot \sum(Y - \bar{Y})^2}}$$

Where, \bar{X} = mean of X variable
 \bar{Y} = mean of Y variable

Assumptions:

- Each observation should have a pair of values.
- Each variable should be continuous.
- It should be the absence of outliers.

- ❑ It assumes linearity and homoscedasticity.



Spearman Correlation

The Spearman correlation coefficient between two features is the Pearson correlation coefficient between their rank values. It is calculated the same way as the Pearson correlation coefficient, but takes into account their ranks instead of their values. It is often denoted with the Greek letter ρ and called Spearman's rho.

This value is mathematically expressed by the following equation:

$$\rho = \frac{\sum_{i=1}^n (R(x_i) - \overline{R(x)})(R(y_i) - \overline{R(y)})}{\sqrt{\sum_{i=1}^n (R(x_i) - \overline{R(x)})^2 \cdot \sum_{i=1}^n (R(y_i) - \overline{R(y)})^2}} = 1 - \frac{6 \sum_{i=1}^n (R(x_i) - R(y_i))^2}{n(n^2 - 1)}$$

Where, $R(x_i)$ = rank of x_i

$R(y_i)$ = rank of y_i

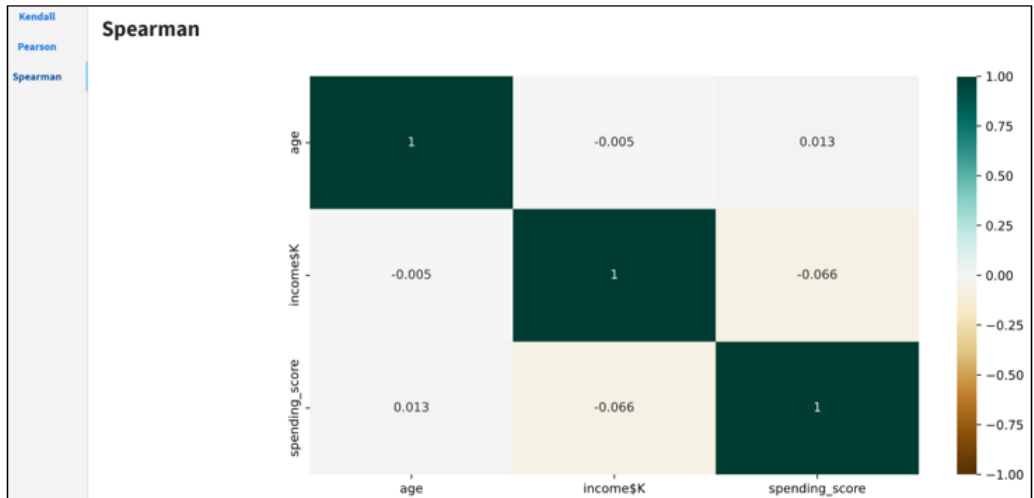
$\overline{R(x)}$ = mean rank of x

$\overline{R(y)}$ = mean rank of y

n = number of pairs

Assumptions:

- Pairs of observations are independent.
- Two variables should be measured on an ordinal, interval, or ratio scale.
- It assumes that there is a monotonic relationship between the two variables.



Kendall Correlation

The Kendall correlation coefficient compares the number of concordant and discordant pairs of data. This coefficient is based on the difference in the counts of concordant and discordant pairs relative to the number of x-y pairs. It is often denoted with the Greek letter tau (τ) and called Kendall's tau.

This value is mathematically expressed by the following equation:

$$\tau = \frac{n_c - n_d}{n_c + n_d} = \frac{n_c - n_d}{n(n - 1)/2}$$

*Where, n_c = number of concordant pairs
 n_d = number of discordant pairs
 n = number of pairs*

Assumptions:

- Pairs of observations are independent.
- Two variables should be measured on an ordinal, interval, or ratio scale.
- It assumes that there is a monotonic relationship between the two variables.

**Comparison**

This section provides a comparison of Pearson correlation vs. Spearman and Kendall correlations.

- Non-parametric correlations are less powerful because they use less information in their calculations. In the case of Pearson's correlation uses information about the mean and deviation from the mean, while non-parametric correlations use only the ordinal information and scores of pairs.
- In the case of non-parametric correlation, it is possible that the X and Y values can be continuous or ordinal, and approximate normal distributions for X and Y are not required. But in the case of Pearson's correlation, it assumes the distributions of X and Y should have normal distribution and also be continuous.
- Correlation coefficients only measure linear (Pearson) or monotonic (Spearman and Kendall) relationships.

Spearman correlation vs. Kendall correlation:

- ❑ In the normal case, Kendall correlation is more robust and efficient than Spearman correlation. It means that Kendall correlation is preferred when there are small samples or some outliers.
- ❑ Spearman's rho usually is larger than Kendall's tau.
- ❑ The interpretation of Kendall's tau in terms of the probabilities of observing the agreeable (concordant) and non-agreeable (discordant) pairs is very direct.

Advanced Insights

This is an optional profiling feature to generate advanced insights on data such as Cluster Analysis, Cohort Analysis, Time Series Analysis, etc. The current release only provides K-Means clustering analysis. Future versions will enable other insights.

Click the *Advanced Insights* tab in the View Profile dialog, and then click *Add Insights* as shown in the following image.



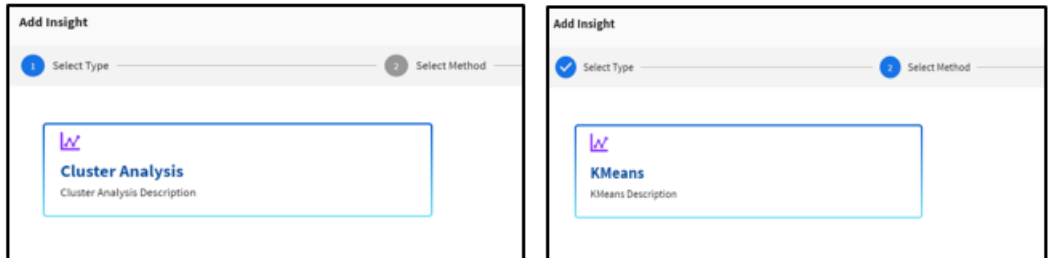
Cluster Analysis

Cluster analysis is a statistical method for organizing items into groups or clusters on the basis of how closely associated they are.

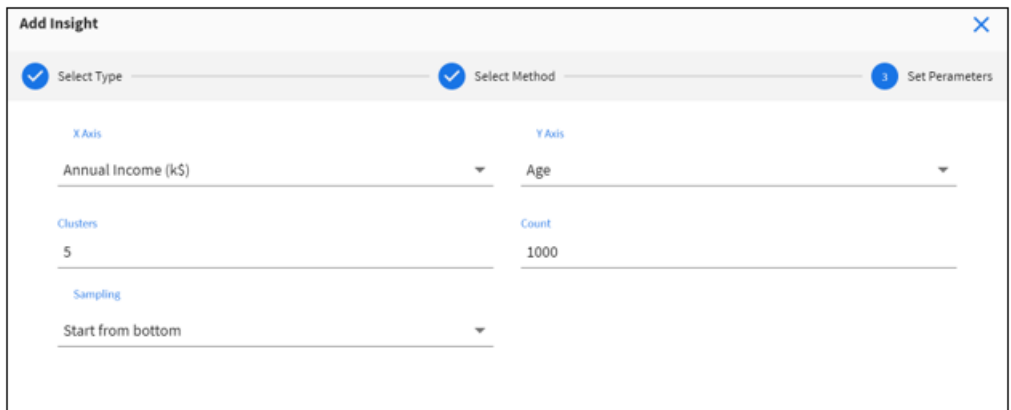
K-Means Clustering

K-means clustering is an unsupervised learning algorithm to partition N observations into K clusters in which each observation belongs to the cluster with the nearest mean (cluster centers or cluster centroid). The algorithm clusters data points based on feature similarity and works iteratively to assign each data point to one of K groups based on the features that are provided.

1. From the Add Insight dialog, click *Cluster Analysis* and select the *KMeans* method, as shown in the following image.



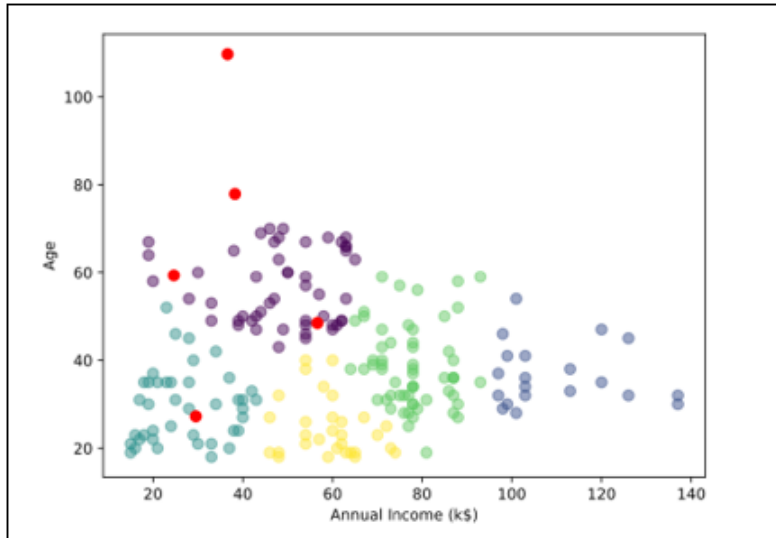
The Set Parameters pane opens, as shown in the following image.



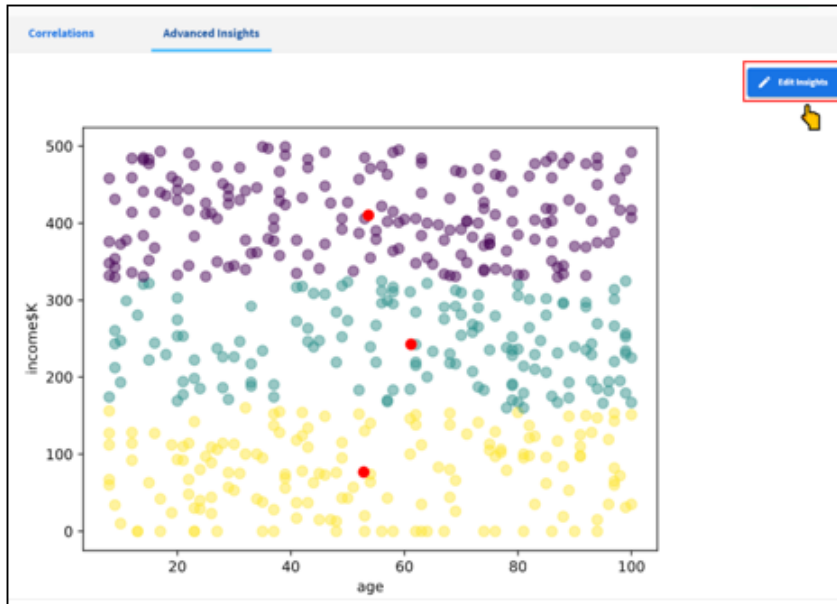
2. Set the following parameters:
 - a. Select variables for the X-axis and Y-axis.
 - b. Select the number of expected clusters in your data set (default is set to 3, min is 1, and max is 25).
 - c. Select number of rows (max is 4000).
 - d. Select sampling order (Top or Bottom).

Note: If your data set has more than 4000 rows, this order determines the top or bottom 4000 rows sampled for analysis.

The analysis is generated based on the specified parameters, as shown in the following image.



3. Click *Edit Insights* to change the parameters and to rerun the analysis, as shown in the following image.



Analyzing Data Quality

Data quality analysis is a process of verifying data values against known rules to ascertain if the values accurately represent the real-world entity. Rules are an implementation of an underlying service and rules are generally associated with a data class. There are several built-in rules available for data quality analysis. These rules can be mapped to a single data attribute or a group of data attributes.

During data analysis, one or more of the following steps are executed:

1. Data values may be cleansed by removing unexpected characters, leading, or trailing spaces.
2. Data values may be validated against data specifications using regular expression checks, blacklist/whitelist checks, lookups, LUHN validation, etc.
3. Missing or incomplete data values may be enriched using reference data or other statistical data imputation methods.
4. Data values may be standardized by reformatting the output values.
5. Data values may be normalized by detecting and replacing abnormal values.
6. Tag values may be generated to report data defects or to highlight other data facts.

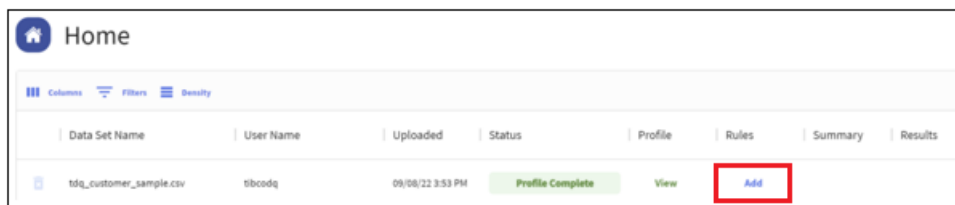
7. The overall outcome of the analysis is summarized into a tag category as:

- VALID.** Input value is either valid or has defects that were fixed (cleansed) and output value is considered valid.
- INVALID.** Input value is invalid and could not be fixed.
- CLEANSSED.** Input value has deficiencies that were fixed and output value is considered valid.
- MISSING.** Input value is missing (blank spaces or null).

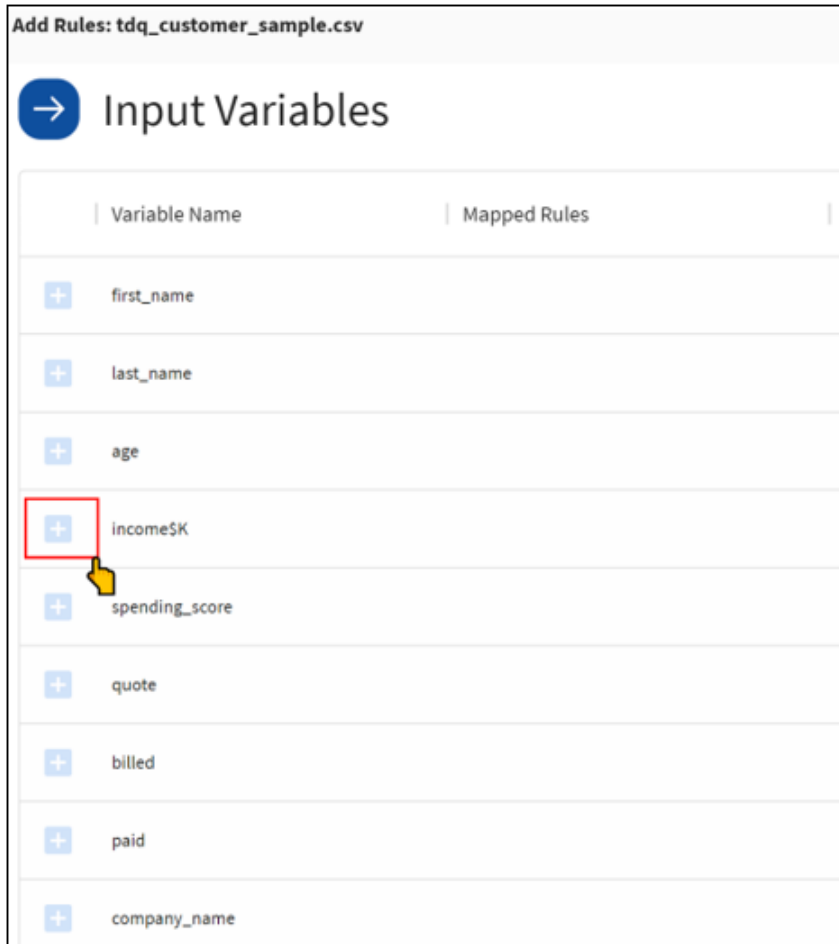
Adding Rules

To add (assign) rules when analyzing data:

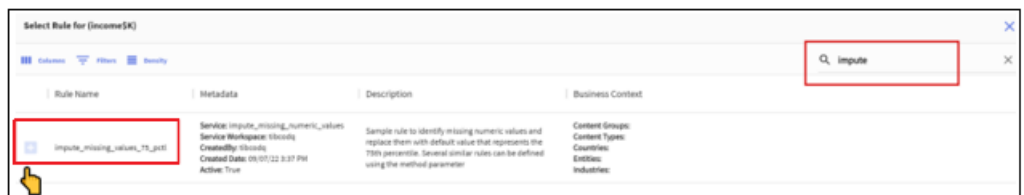
1. Click *Add* under Rules, as shown in the following image.



- In the left pane of the mapping window, click the plus (+) icon next to an input data variable (for example, income\$K), as shown in the following image.



The Select Rule dialog opens, as shown in the following image.

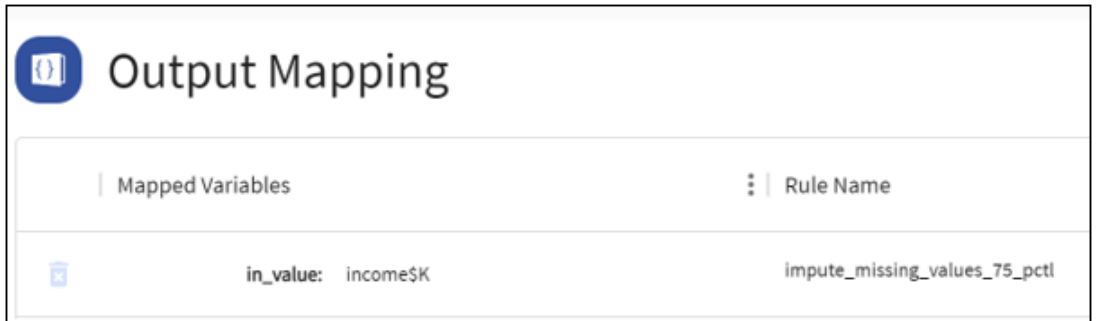


- Search the Rules Catalog and select the most appropriate rule.

4. Map input data variables to the rule input.

Single Variable Rules

These rules need one input and once selected, the input data variable is automatically mapped to the rule input (for example, income\$K).



Multi-Variable Rules

These rules require multiple inputs.

1. Map each rule input to an input variable.
2. Provide a meaningful group name.

Note: The group name is used to generate the output results file name, so make sure you provide a unique group name.

Rule Mapping: cleanse_usa_address_loqate ✕

→ Input Variables

<p><small>in_address_line1</small> address</p>	▼	<p><small>in_address_line2</small></p>
<p><small>in_address_city</small> city</p>	▼	<p><small>in_address_state</small> state</p>
<p><small>in_address_postal</small> zip</p>	▼	

🗑️ Group Name

Group Name *
home_address

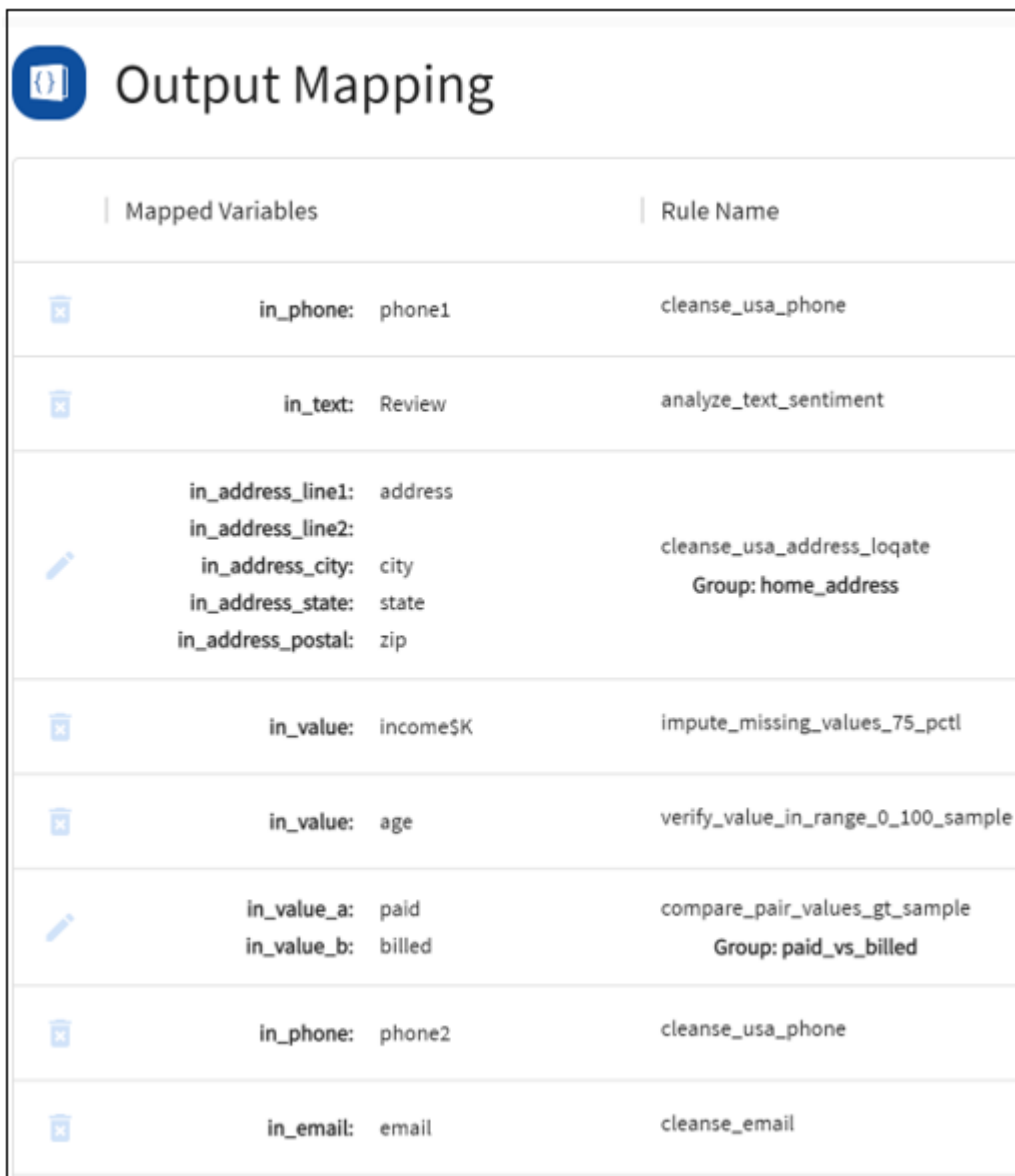
Cancel
Save

Preassigned Rules









Some columns may have a preassigned rule based on the data class discovered by the profiler. You can delete and reassign a different rule if the assigned rule is not appropriate for the input variable.



Verify all the rules associated with the input variables before submitting the data for processing.

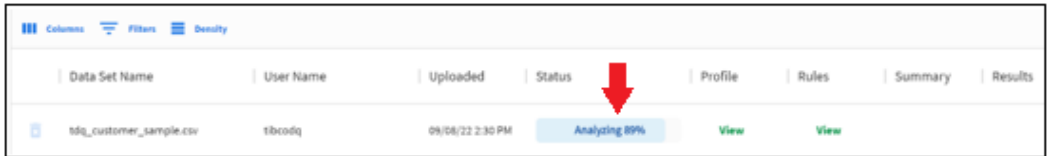


The screenshot shows the 'Output Mapping' interface with a table of mapped variables and their associated rules. The table has two columns: 'Mapped Variables' and 'Rule Name'. Each row represents a mapping, with a trash icon on the left for deletion and a pencil icon for editing. The 'Mapped Variables' column lists input variables and their values, while the 'Rule Name' column lists the corresponding rule name and group.

	Mapped Variables	Rule Name
	in_phone: phone1	cleanse_usa_phone
	in_text: Review	analyze_text_sentiment
	in_address_line1: address in_address_line2: in_address_city: city in_address_state: state in_address_postal: zip	cleanse_usa_address_loqate Group: home_address
	in_value: income\$K	impute_missing_values_75_pctl
	in_value: age	verify_value_in_range_0_100_sample
	in_value_a: paid in_value_b: billed	compare_pair_values_gt_sample Group: paid_vs_billed
	in_phone: phone2	cleanse_usa_phone
	in_email: email	cleanse_email

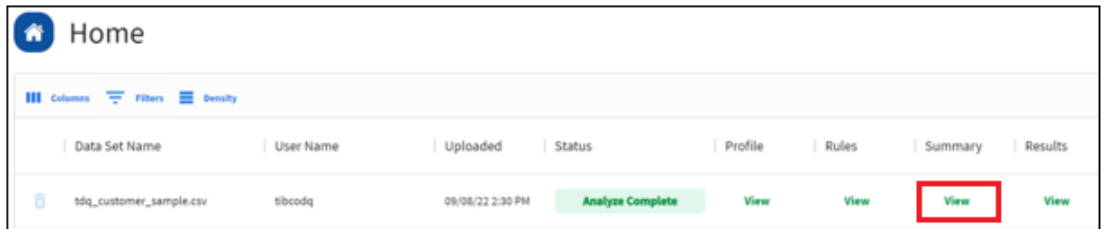
Tracking Analysis Progress

The analysis progress bar displays the current processing status, as shown in the following image.



Viewing a Summary of Analysis Results

You can view a summary of analysis results by clicking *View* under Summary, as shown in the following image.

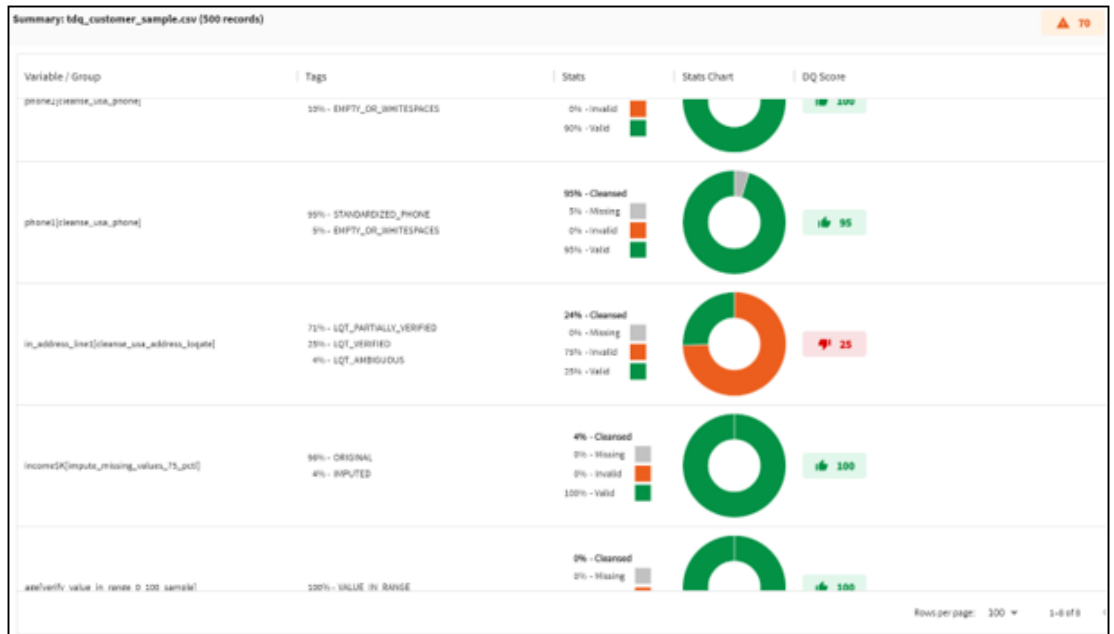


Summarized results contain:

Result	Description
Overall DQ Score	Data Quality (DQ) score for the entire data set. This score will be in the range 0 to 100 (100 is the best).
Rule based DQ Score	Data Quality (DQ) score for each rule applied to a variable or a group of variables. This score will be in the range 0 to 100 (100 is the best).
Tags	List of issues or reportable facts identified during the data quality analysis.

Result	Description
Stats & Stats Chart	<p>Summarized stats for final outcome of the analysis.</p> <ul style="list-style-type: none"> <input type="checkbox"/> VALID. Represents percentage of valid output values. <input type="checkbox"/> INVALID. Represents percentage of values that failed validation and could not be fixed. <input type="checkbox"/> MISSING. Represents percentage of values that are empty or missing. <input type="checkbox"/> CLEANSED. Represents percentage of values that were fixed - standardized, normalized, or enriched.

The following is a sample summary of analysis report.



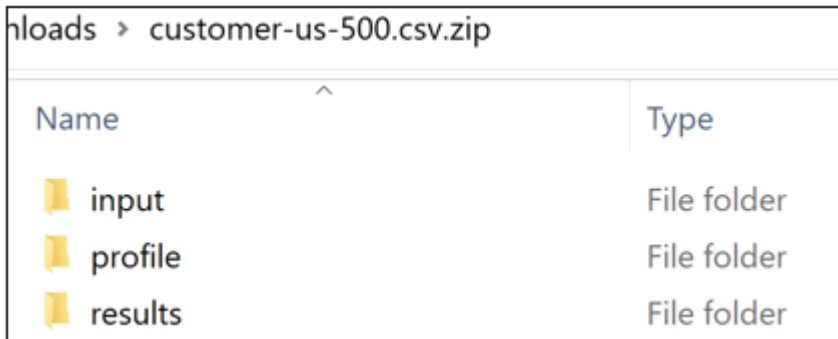
Viewing Results

You can download the detailed results of the Data Quality (DQ) analysis by clicking **View** under Results.

The .zip file that is generated contains the following folders:

- input.** Contains the input data set.
- profile.** Contains data profiling results.
- results.** Contains DQ analysis results.

For example:



Name	Type
input	File folder
profile	File folder
results	File folder

Results Output

The results folder contains analysis results.

1. One JSON file named **rules.json** with the summary of rules mapped by the user.
2. One JSON file with the naming convention **<<input_file_name>>.results.json** with the summarized results of the rule based DQ analysis.
3. One CSV file with the naming convention **<<input_file_name>>.cleansed.csv** with the same number of columns as the input data set.
4. Multiple CSV files with the naming convention **<<variable or group name>>[<<rule name>>].csv** representing an output file for each combination of variable/group and rule that is executed.

In the following example, a user submitted eight rules for execution, and the results folder contains:

- One rules JSON.
- One summarized output JSON.

- One cleansed output CSV.
- Eight output CSVs (one for each rule).

Output Mapping

Mapped Variables	Rule Name
in_phone: phone1	cleanse_usa_phone
in_text: review	analyze_text_sentiment
in_address_line0: address in_address_line0: city in_address_state: state in_address_postal: zip	cleanse_usa_address_loqate Group: home_address
in_value: income5k	impute_missing_values_75_pct1
in_value: age	verify_value_in_range_0_100_sample
in_value_k: paid in_value_k: billed	compare_pair_values_gt_sample Group: paid_vs_billed
in_phone: phone2	cleanse_usa_phone
in_email: email	cleanse_email

Name

- rules.json
- tdq_customer_sample.csv.results.json
- age[verify_value_in_range_0_100_sample].csv
- email[cleanse_email].csv
- home_address[cleanse_usa_address_loqate].csv
- income\$K[impute_missing_values_75_pct1].csv
- paid_vs_billed[compare_pair_values_gt_sample].csv
- phone1[cleanse_usa_phone].csv
- phone2[cleanse_usa_phone].csv
- Review[analyze_text_sentiment].csv
- tdq_customer_sample.csv.cleansed.csv

rules.json - Summary of rules mapped by the user.

Value	Description
Rule Name	Name of the rule selected by the user.
Group Name	Group name provided by the user for mapping multiple variables into a single Rule.
Input Map	Mapping of input variables to rule inputs.

Value	Description
Variable Options	<p>Data expectations by the user for each variable.</p> <ul style="list-style-type: none"> <input type="checkbox"/> shouldBeUnique. Set to <i>true</i> if column values are always expected to be unique. <input type="checkbox"/> allowsNulls. Set to <i>false</i> if column values can never be Null. <input type="checkbox"/> businessImpact. A number that represents HIGH, MEDIUM, or LOW impact of the variable on business outcomes.

The following is a sample *rules.json* file for reference.

```

{
  "ruleName" : "cleanse_usa_address_loqate",
  "groupName" : "home_address",
  "inputMap" : {
    "in_address_line1" : "address",
    "in_address_line2" : null,
    "in_address_city" : "city",
    "in_address_state" : "state",
    "in_address_postal" : "zip"
  },
  "ruleId" : "cleanse_usa_address_loqate"
}, {
  "ruleName" : "cleanse_email",
  "inputMap" : {
    "in_email" : "email"
  },
  "id" : "cleanse_email",
  "ruleId" : "cleanse_email"
} ],
"variableOptions" : {
  "zip" : {
    "id" : "zip",
    "businessImpact" : 10,
    "allowsNulls" : true,
    "shouldBeUnique" : false,
    "values" : [ "70116", "48116", "8014" ]
  }
}

```

<<input_data_set_name>>.results.json - JSON output with summarized results of the DQ analysis.

Value	Description
Input File	Input file that contains the input data set uploaded by the user.
Output File	Output file that contains cleansed output values of all the variables in the input data set.
Overall DQ Score	Data Quality (DQ) score for the entire data set in the range 0 to 100 (100 is the best).
Rule based DQ Score	Data Quality (DQ) score for each variable in the range 0 to 100 (100 is the best).
Tags	List of issues or reportable facts identified during the data quality analysis.
Count Processed	Total number of data values processed by a Rule.
Count Valid	Total number of valid output values generated by a Rule.
Count Invalid	Total number of values that failed validation and could not be fixed by a Rule.
Count Missing	Total number of empty or missing values.
Count Cleansed	Total number of cleansed output values generated by a Rule.

The following is a sample `<<input_data_set_name>>.results.json` file for reference.

```

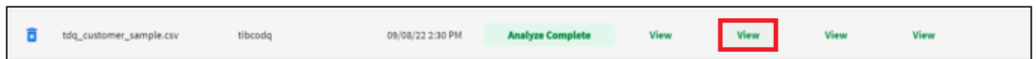
{
  "inputFile": "tibcdq/active/27aa1176-9f6f-4b95-ab53-976d861c382d/input/customer-us-500.csv",
  "outputFile": "files:///tdq-storage/tibcdq/active/27aa1176-9f6f-4b95-ab53-976d861c382d/results/2022-09-07T20.16.48.169/customer-us-500.csv.cleaned.csv",
  "dqScore": 62,
  "results": [
    {
      "fileName": "files:///tdq-storage/tibcdq/active/27aa1176-9f6f-4b95-ab53-976d861c382d/results/2022-09-07T20.16.48.169/Review[analyze_text_sentiment].csv",
      "ruleId": "analyze_text_sentiment",
      "dqScore": 100,
      "tags": {
        "POSITIVE": 268,
        "NEUTRAL": 118,
        "NEGATIVE": 114
      },
      "countCleaned": 500,
      "countNull": 0,
      "countProcessed": 500,
      "countRejected": 0,
      "countValid": 500
    },
    {
      "fileName": "files:///tdq-storage/tibcdq/active/27aa1176-9f6f-4b95-ab53-976d861c382d/results/2022-09-07T20.16.48.169/phone2[cleans_usa_phone].csv",
      "ruleId": "cleans_usa_phone",
      "dqScore": 100,
      "tags": {
        "STANDARDIZED_PHONE": 500
      },
      "countCleaned": 500,
      "countNull": 0,
      "countProcessed": 500,
      "countRejected": 0,
      "countValid": 500
    }
  ]
}

```

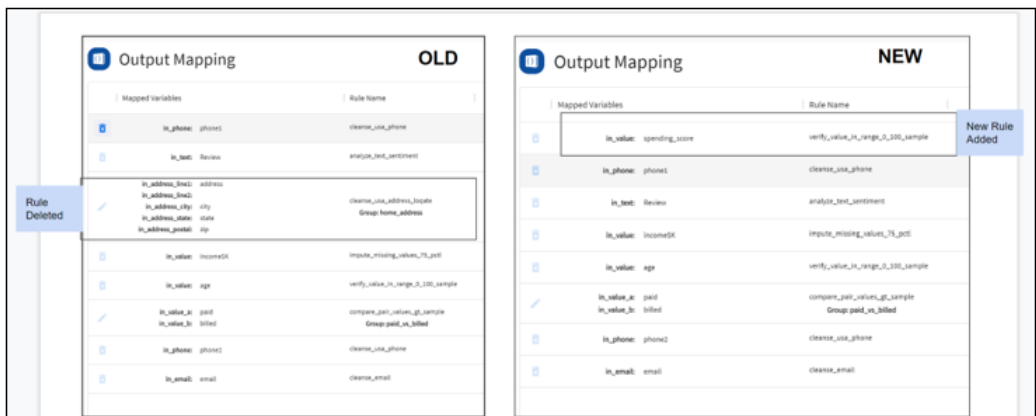
Rerun Analysis With Different Rules

After analyzing your data through Rules for the first time, you can re-analyze the same data with a different set of Rules, allowing you to skip the data upload and profiling steps. A new record is generated that displays at the top of the home page for each rerun.

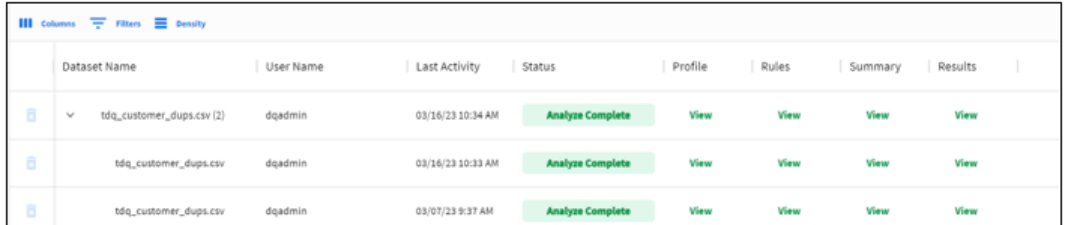
1. Click **View** under Rules, as shown in the following image.



2. Edit the rules by replacing existing rules or adding new ones, as shown in the following image.



- When you submit the new request for Rules execution, the most recent analysis will be shown at the top of the home page. The results from different analysis runs will be grouped together with the most recent results shown first. Click on the drop-down button to expand and view results from previous analysis.



The screenshot shows a table with the following columns: Dataset Name, User Name, Last Activity, Status, Profile, Rules, Summary, and Results. The first row is expanded, showing a dropdown arrow next to the Dataset Name 'tdq_customer_dups.csv (2)'. The Status column for all rows contains a green 'Analyze Complete' button. The Profile, Rules, Summary, and Results columns contain 'View' links.

	Dataset Name	User Name	Last Activity	Status	Profile	Rules	Summary	Results
▼	tdq_customer_dups.csv (2)	dqadmin	03/16/23 10:34 AM	Analyze Complete	View	View	View	View
	tdq_customer_dups.csv	dqadmin	03/16/23 10:33 AM	Analyze Complete	View	View	View	View
	tdq_customer_dups.csv	dqadmin	03/07/23 9:37 AM	Analyze Complete	View	View	View	View



Chapter 5

Management and Monitoring

This chapter describes how to manage ibi Data Quality services, data classes, rules, users, and monitor Data Quality (DQ) metrics.

In this chapter:

- [Managing Services](#)
 - [Managing Data Classes](#)
 - [Managing Rules](#)
 - [Managing Users](#)
 - [Monitoring Data Quality Metrics](#)
 - [Managing Artifacts](#)
-

Managing Services

In ibi Data Quality, a service represents a workflow that implements the logic to verify and cleanse data. Services can either be generic or domain specific.

Packaged (Built-in) Services

ibi Data Quality provides a set of built-in services that can be exposed as Rules. These services are registered in the DQCore workspace and cannot be edited or replaced by end users. However, service developers can define and register new services in the Custom workspace.

For more information on the default services that are included in the product, see [Packaged DQ Services](#) on page 187.

Adding New Services

This section describes how to add new services in ibi Data Quality.

Service Requirements

To add a new service in ibi Data Quality, the new service must comply with the following ibi Data Quality service specifications.

Request Method

The service must support HTTP POST.

Authentication Method

The service may require basic authentication (user ID and password), but no other security mechanism is supported.

Service Inputs

1. The input to the service must be a single block of CSV text.
2. The input data must not include a column header.
3. Input values must be provided in the order specified in the service definition provided to ibi Data Quality. For more information, see [Service Definition JSON](#) on page 91.
4. When creating the service definition, input column names should be prefaced with *in_*.

For example, an email cleanse service should have an input variable called *in_email*.

Service Outputs

1. Output column names that represent cleansed input values should be prefaced by *out_*.

Example: An email cleanse service should have an output variable called *out_email*.

2. In addition, the output should contain one column named *tag_value* and another called *tag_category*.
 - a. Tag value can be empty. Tag values should represent a list of issues or reportable facts identified in the input data. If a tag value is generated it should be a string consisting of letters, underscores or digits.

Example: An email cleanse service can generate the following types of tag values:

- EMPTY_OR_WHITESPACES.** Input value is an empty string or contains white spaces.
- COULD_NOT_VERIFY.** Value does not represent a valid email address.
- DISPOSABLE_TEMPORARY_COMPLAINER.** The email address provided is a disposable mailbox.

- b. Tag category can be empty. Tag category should be used to categorize the final outcome of the data quality analysis that must be either of these four values:

- MISSING.** Indicates the input data is empty or missing.
- VALID.** Indicates the input data is valid and has the same value as the output data.

- ❑ **CLEANSED.** Indicates that the input value was cleansed and a standardized, augmented, or enriched output value was generated.
 - ❑ **INVALID.** Indicates the input value is invalid and the issues identified cannot be fixed.
3. All input values should be included in the output and records should be emitted in the same order in which they were provided in the input data set.
 4. The output should be a CSV block with the first line being a header.

Service Parameters

Developers may optionally specify parameters for Services. These parameters represent name-value pairs that are passed in the Service request and are immutable values in the scope of a request executed by the Service.

Example: The *cleanse_email* service specifies its URL as http://server.com/cleanse_email and a possible parameter as *default_email*, which allows a rule author to set up a default email that replaces invalid email addresses in the output. Execution of the rule would result in the following URL:

http://server.com/cleanse_email?default_email=service@myco.com

Service Registration

This section describes service registration requirements.

Service Definition JSON

Create a definition for the service in JSON format (an example is provided below). All fields except credentials and parameters are required.

Attribute	Description
name	Name of the service, has to be unique in the “custom” workspace
description	A brief description of the service
location	Service location URL
sendFullDataSet	Optional. Set this to <i>true</i> if the service expects the entire data set as input as opposed to one row at a time.

Attribute	Description
supportsJson	Optional. Set this to <i>true</i> if the service can interpret the request body in JSON format and generate a response in JSON format.
inputColumns	An array of paired values that have: <ul style="list-style-type: none"> <input type="checkbox"/> name. Name of the input column <input type="checkbox"/> description. A brief description of the input column.
outputColumns	An array of paired values that have: <ul style="list-style-type: none"> <input type="checkbox"/> name. Name of the output column <input type="checkbox"/> description. A brief description of the output column.
parameters	Optional. An array of values that have: <ul style="list-style-type: none"> <input type="checkbox"/> name. Name of the parameter. <input type="checkbox"/> description. A brief description of the parameter.
credentials	Optional. A pair of values that have: <ul style="list-style-type: none"> <input type="checkbox"/> user. User name for the service credentials. <input type="checkbox"/> password. Password for the service credentials.

The following is an example JSON document for a new service called *cleanse_vin*:

```

{
  "name": "cleanse_vin",
  "description": "Cleanse vehicle identification number",
  "location": "http://service.mysite.com/cleanse_vin",
  "supportsJson": true
  "inputColumns": [
    {
      "name": "in_vin",
      "description": "value to be cleansed or verified"
    }
  ],
  "outputColumns": [
    {
      "name": "out_vin",
      "description": "input value when tag_category is VALID,
cleansed value when tag_category is CLEANSED, default value when
tag_category is MISSING or INVALID"
    },
    {
      "name": "tag_value",
      "description": "Tag value that provides explanation for
malformed, unexpected or missing data"
    },
    {
      "name": "tag_category",
      "description": "Tag category that categorizes tags as Missing
Data, Cleansed Data or Invalid Data"
    }
  ],
  "tags": [
    {
      "name": "EMPTY_OR_WHITESPACES",
      "description": "Input value is an empty string or contains all
whitespaces"
    },
    {
      "name": "INVALID_VIN",
      "description": "Value does not represent a valid email address"
    },
    {
      "name": "VALID_VIN",
      "description": "Input value is Valid"
    },
    {
      "name": "NORMALIZED_VIN",
      "description": "Input value was reformatted or cleansed"
    }
  ],
  "parameters": [
    {
      "name": "default_vin",
      "description": "Default value when tag_category is MISSING or
INVALID"
    }
  ]
}

```

Service Registration Endpoint

To register a new service, POST the service definition JSON (described above) to the following endpoint:

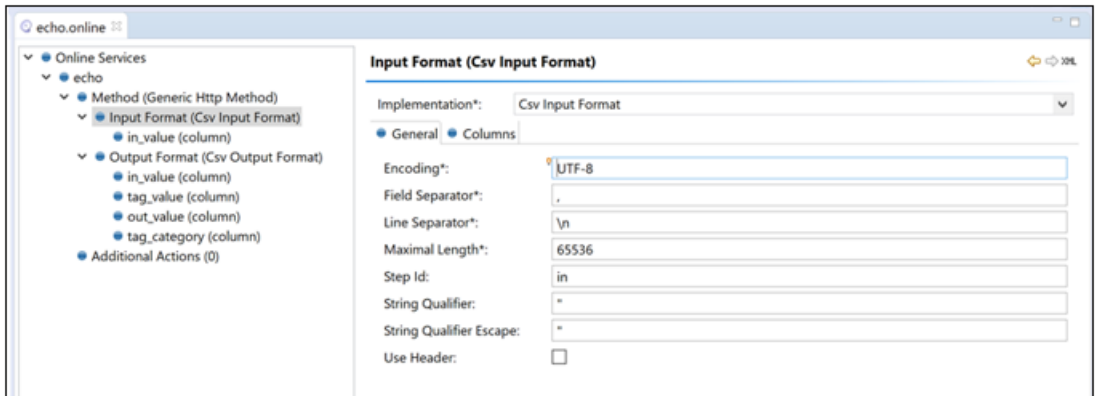
`https://{host}:9803/api/v1/service`

The response will contain the ID of the new service.

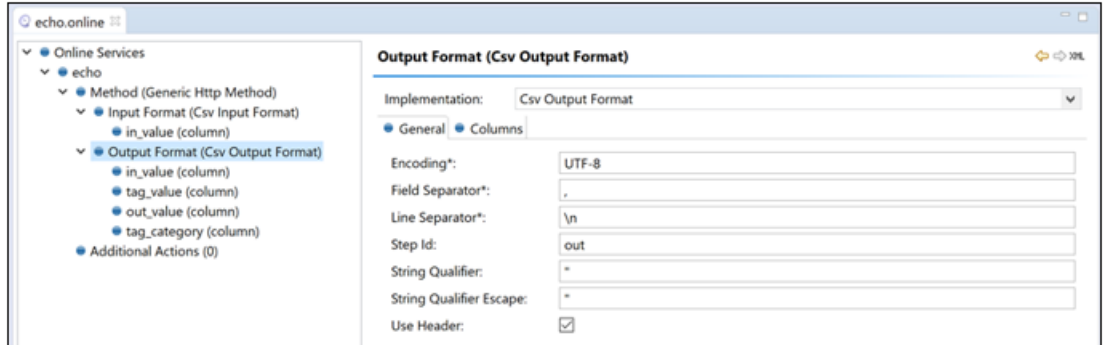
If a new service is successfully registered, it will be available for rule authors to create new ibi Data Quality rules using the new service. For more information, see [Managing Rules](#) on page 104.

Authoring Services Using Omni-Gen Data Quality Server

Developers familiar with Omni-Gen Data Quality Server (DQS) can create new projects and add them to their Data Quality server. Set the online file for the project to be of type *Generic Http* with a CSV input format and include parameters as shown below. Set all input columns to be of type *string* and prefix them with *in_*.



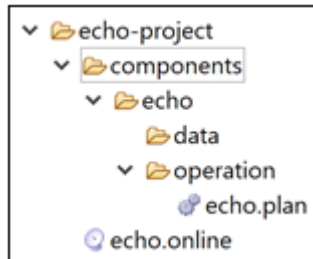
Set the output format to be of type CSV with parameters, as shown in the following image.



The service must meet the service requirements as mentioned in [Service Requirements](#) on page 89.

Project Structure and Deployment

Create the service with a directory structure, as shown below:



Copy the service to the running `dq-dqs` container by executing the following command from the root of the project folder:

```
C:\echo-project>docker cp . dq-dqs:/dqs/server/services
```

Follow the instructions in [Service Registration](#) on page 91 to register the new DQS service. Inspect the DQS log for more information on any service-related errors.

Managing Data Classes

A data class represents a real-world entity (for example Credit Card, Email, Phone Number, etc.). When a user uploads a data set and submits a request for data profiling, the Profiler uses the definitions of known data classes from the Knowledge Hub to classify and tag those data variables with their corresponding data classes.

The screenshot shows a 'View Profile' interface with tabs for Overview, Variables, Duplicates, Correlations, and Advanced Insights. The 'Variables' tab is active, displaying a table of data variables. Two variables are highlighted: 'first_name' and 'last_name'. For 'first_name', the suggested data class is 'person_first_name', which is highlighted with a red box. For 'last_name', the suggested data class is 'person_last_name'. The table also shows profile scores, data types, and various statistics for each variable.

Variable Name	Profile Score	Data Type	Sensitive Data	Suggested Data Class	Complete %	Unique %	Profile Highlights	Frequency
first_name	100	string		person_first_name	95%	93%	Length: Min: 2 Max: 10 Avg: 5.97 Median: 6 Most Common Mask: LLLLLL (20%) Most Common Pattern: W (100%)	Quentin: <1% Pamela: <1% Mittie: <1% Lesha: <1% Lesha: <1% Other values: 93%
last_name	100	string		person_last_name	94%	94%	Length: Min: 3 Max: 13 Avg: 6.81 Median: 7 Most Common Mask: LLLLLL (25%) Most Common Pattern: W (100%)	Zurcher: <1% Zepp: <1% Zane: <1% Zager: <1% Yam: <1% Other values: 93%

Packaged (Built-in) Data Classes

The following table lists and describes the built-in data classes that are recognized by the Profiler.

Name	Description	Sensitive Flag
address_city	Address City	FALSE
address_country	Address Country	FALSE
address_line	Address Line	FALSE
address_postal_code	Address Postal Code	FALSE
address_state	Address State	FALSE
airport_code	Airport Code	FALSE
date	Date	FALSE
email	Email Address	TRUE
gender	Person Gender	FALSE

Name	Description	Sensitive Flag
iban	International Bank Account Number	TRUE
person_first_name	Person First Name	FALSE
person_full_name	Person Full Name	FALSE
person_last_name	Person Last Name	FALSE
person_name_prefix	Person Name Prefix	FALSE
person_name_suffix	Person Name Suffix	FALSE
phone_number	Phone Number	TRUE
time	Time	FALSE
us_company_name	United States Company Name	FALSE
us_dea	United States Drug Enforcement Agency assigned Prescriber Identifier	TRUE
us_npi	United States National Provider Identifier	TRUE
us_ssn	United States Social Security Number	TRUE
vin	Vehicle Identification Number	TRUE

The following image shows the built-in data classes exposed through the web UI.

Data Class	Description	Type	Created By	Sensitive	Created Date	Data Class	Description	Type	Created By	Sensitive	Created Date
address_city	Address City	Machine Learning	ibioddy		09/15/22 9:28 AM	person_full_name	Person Full Name	Machine Learning	ibioddy		09/15/22 9:28 AM
address_county	Address County	Machine Learning	ibioddy		09/15/22 9:28 AM	person_last_name	Person Last Name	Machine Learning	ibioddy		09/15/22 9:28 AM
address_line	Address Line	Machine Learning	ibioddy		09/15/22 9:28 AM	person_name_parts	Person Name Parts	Machine Learning	ibioddy		09/15/22 9:28 AM
address_postal_code	Address Postal Code	Machine Learning	ibioddy		09/15/22 9:28 AM	person_name_suffix	Person Name Suffix	Machine Learning	ibioddy		09/15/22 9:28 AM
address_state	Address State	Machine Learning	ibioddy		09/15/22 9:28 AM	phone_number	Phone Number	Regular Expression	ibioddy	🔒	09/15/22 9:28 AM
airport_code	Airport Code	Machine Learning	ibioddy		09/15/22 9:28 AM	time	Time	Machine Learning	ibioddy		09/15/22 9:28 AM
city	City	Machine Learning	ibioddy		09/15/22 9:28 AM	us_company_name	United States Company Name	Machine Learning	ibioddy		09/15/22 9:28 AM
email	Email Address	Regular Expression	ibioddy	🔒	09/15/22 9:28 AM	us_fbi	United States FBI Enforcement Agency assigned Research Identifier	Regular Expression	ibioddy	🔒	09/15/22 9:28 AM
gender	Person Gender	Machine Learning	ibioddy		09/15/22 9:28 AM	us_ipr	United States National Precision Identifier	Regular Expression	ibioddy	🔒	09/15/22 9:28 AM
iban	International Bank Account Number	Regular Expression	ibioddy	🔒	09/15/22 9:28 AM	us_ssn	United States Social Security Number	Regular Expression	ibioddy	🔒	09/15/22 9:28 AM
person_first_name	Person First Name	Machine Learning	ibioddy		09/15/22 9:28 AM	vin	Vehicle Identification Number	Regular Expression	ibioddy	🔒	09/15/22 9:28 AM
person_full_name	Person Full Name	Machine Learning	ibioddy		09/15/22 9:28 AM						

Tip: User-defined data classes are shown in blue, as shown in the following image.

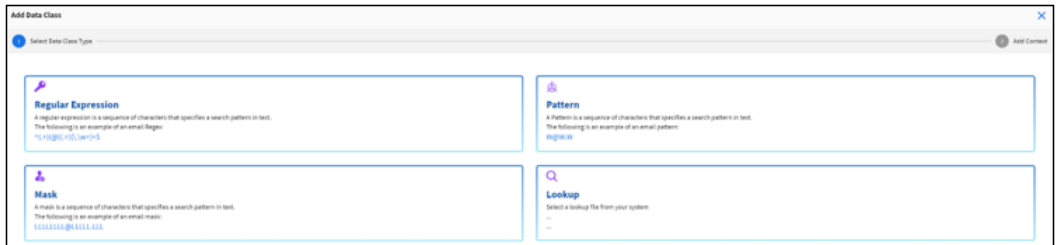
vin	Vehicle Identification Number	Regular Expression	tibcdq	🔒	09/15/22 9:26 AM
web_url	Identify web url using pattern match	Pattern	tibcdq		09/19/22 9:57 AM

Defining New Data Classes

Users can use the Data Class Editor to add definitions for new data classes.

1. Go to the *Data Classes* tab and click *New Data Class*.

The Add Data Class dialog opens, as shown in the following image.



2. Select one of the following options:

- Regular Expression.** If you have a regular expression that can be used to match input data values.
- Lookup.** If you have a CSV data set reference data values.
- Mask.** If you have run the profile for the data set, you can choose to select one or more masks from the profile.

- ❑ **Pattern.** If you have run the profile for the data set, you can choose to select one or more patterns from the profile.

Regular Expression

Use this option when you have a valid regular expression to identify a new class of data.

1. Enter a unique name for the data class.
2. Set the Sensitive data flag to *True* or *False*.
3. Provide a description for the new data class.
4. Enter a valid regular expression.

The screenshot shows a 'Data Class (html_tag)' configuration window. The 'Name' field contains 'html_tag' and the 'Sensitive' flag is set to 'False'. The 'Description' field contains 'Data class that represents a html tag'. Below the description, there is a blue icon representing a document with a magnifying glass, followed by the text 'Regular Expression'. At the bottom, there is a 'Regular Expression' label and a text input field containing a complex regular expression.

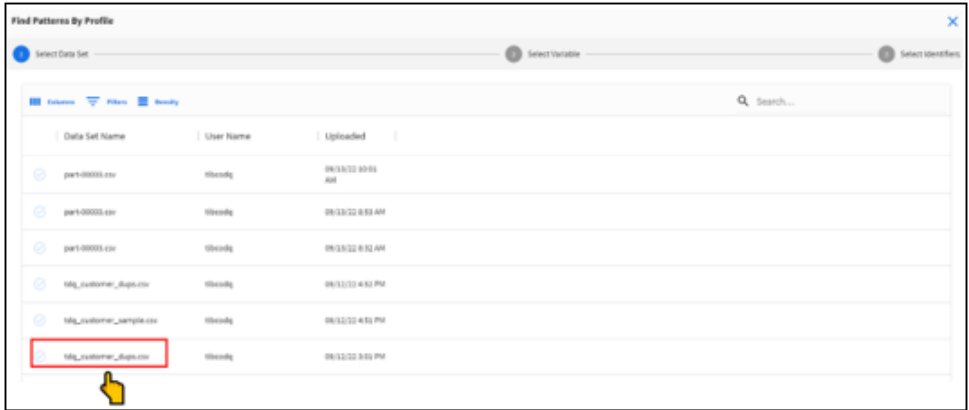
Pattern or Mask

Use this option when you have generated a data profile and want to use the patterns and masks discovered by the profiler to define a new data class. The following steps apply to patterns and masks:

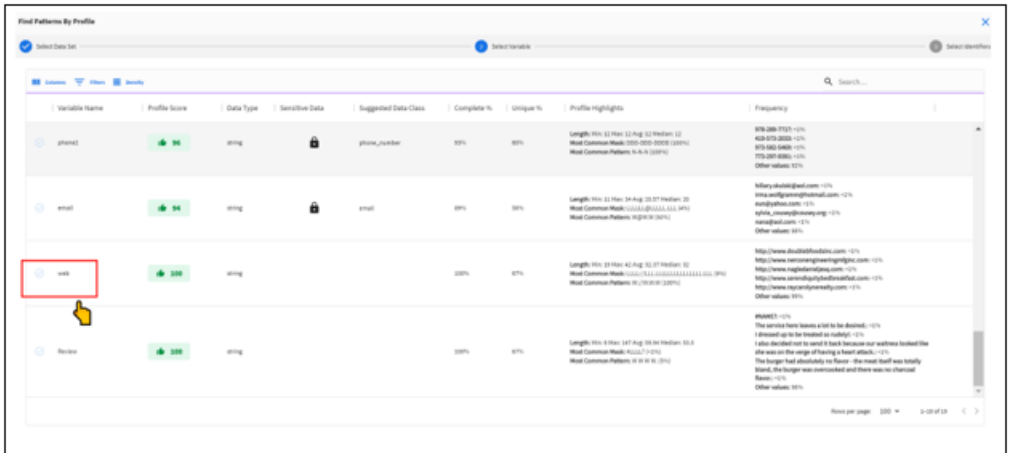
1. Enter a unique name for the data class.
2. Set the Sensitive data flag to *True* or *False*.
3. Provide a description for the new data class.
4. Click *Find Profile Patterns* or *Find Profile Masks* to select pattern(s) or mask(s) from a previously generated data profile, as shown in the following image.

The screenshot shows a 'Data Class (web_url)' configuration window. The 'Name' field contains 'web_url' and the 'Sensitive' flag is set to 'False'. The 'Description' field contains 'Pattern based match to identify web url'. Below the description, there is a blue icon representing a document with a magnifying glass, followed by the text 'Patterns (Max of 5)'. At the bottom, there is a 'Patterns' label and a text input field containing 'W:/WWW'. A red box highlights a blue button labeled 'Find Profile Patterns' with a magnifying glass icon, and a yellow hand cursor is pointing at it.

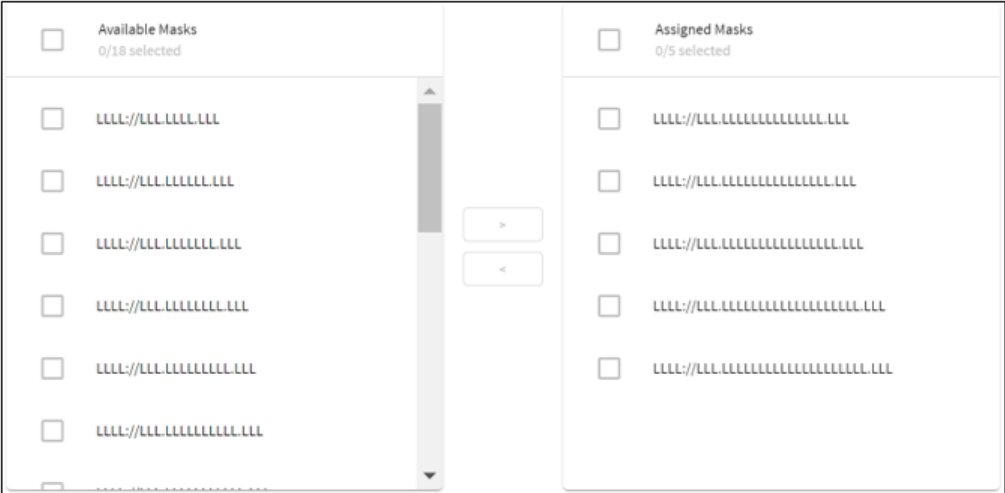
5. Search and select the data set name of the profile you are going to use, as shown in the following image.



6. Select the appropriate variable from the data profile, as shown in the following image.

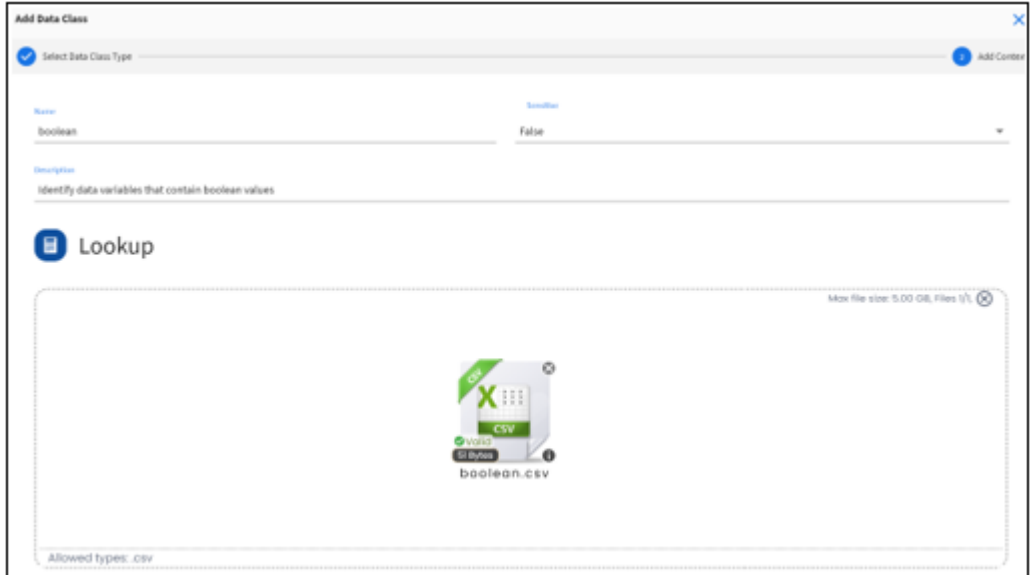


7. Select up to a max of five different Patterns or Masks, and then click *Finish*.



Lookup

Use this option if you have an enumerated list of values that represent the new class of data.

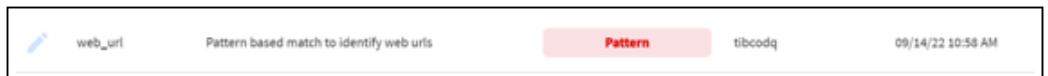


1. Create a CSV lookup file that meets the following specifications:
 - The CSV file requires a header row.
 - The CSV file contains only one column of data values.
 - The CSV file contains a max of up to 1000 values.
2. Enter a unique name for the data class.
3. Set the Sensitive data flag to *True* or *False*.
4. Provide a description for the new data class.

Verifying New Data Classes

To verify new data classes:

1. Go to the *Data Classes* tab and verify that the newly created data class definition is successfully published into the Knowledge Hub, as shown in the following image.



- Upload data and re-profile that data to verify the newly created data class is identified by the profiler, as shown in the following image.

email	100	string		email	89%	56%	Length: Min: 11 Max: 34 Avg: 20.57 Median: 20 Most Common Mask: LLLLLL@LLLLLLLL (4%) Most Common Pattern: W@W.W (50%)
web	100	string		web_url	100%	67%	Length: Min: 19 Max: 42 Avg: 32.37 Median: 32 Most Common Mask: LLLL/LLLLLLLLLLLLLLLLLLL (9%) Most Common Pattern: W./W.W (100%)
Review	100	string			100%	67%	Length: Min: 6 Max: 147 Avg: 59.94 Median: 53.5 Most Common Mask: #LLLL? (*1%) Most Common Pattern: W W W W. (5%)

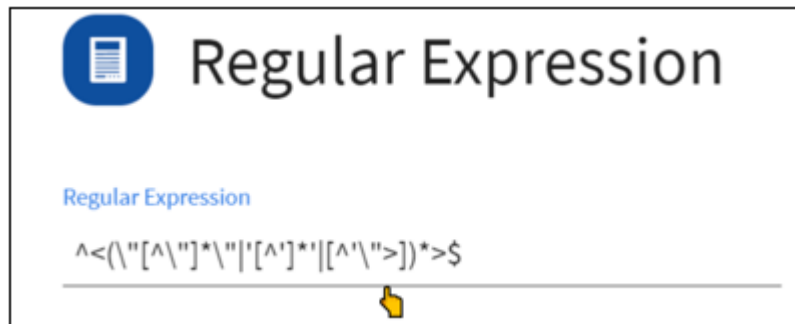
Editing Data Classes

Users can edit the data classes they have created.

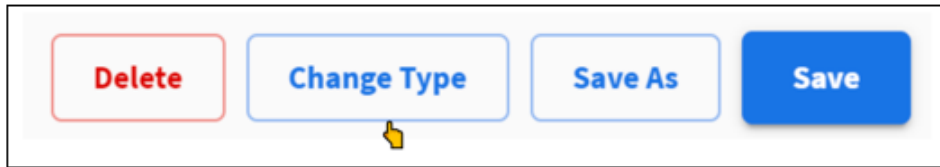
- Go to the *Data Classes* tab and click the *edit* icon next to the data class name, as shown in the following image.

	html_tag	Data class that represents a html tag	Regular Expression	tibcodq
	web_url	Pattern based match to identify web urls	Pattern	tibcodq

- You can either change the definition, such as replacing it with a new Regular Expression, add/remove masks and patterns, or replace it with a new lookup file.



3. You can change the data class definition type from the existing type to a new type, such as changing a Pattern-based definition to a Regular Expression, etc.



Managing Rules

ibi Data Quality provides users with a set of built-in rules that can be used for Data Quality (DQ) analysis. These rules provide sample implementations illustrating the use of service parameters for the different services in ibi Data Quality. ibi Data Quality users can add, edit, or delete rules in the Rules Catalog.

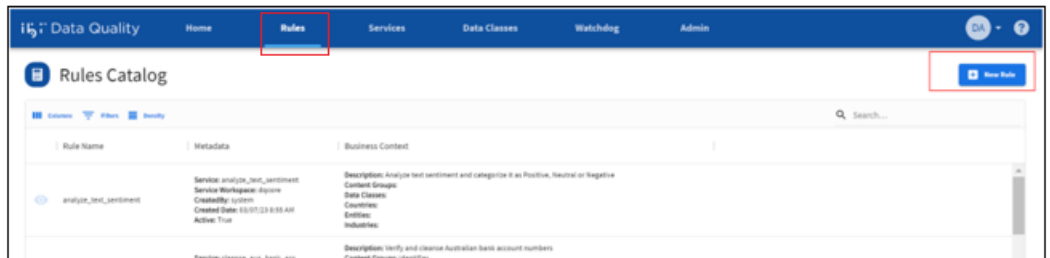
Adding New Rules

There are two ways to add new rules.

Defining a New Rule

To define a new rule:

1. Go to the *Rules* tab (Rules Catalog) and click on *New Rule*, as shown in the following image.



2. Enter the rule information.

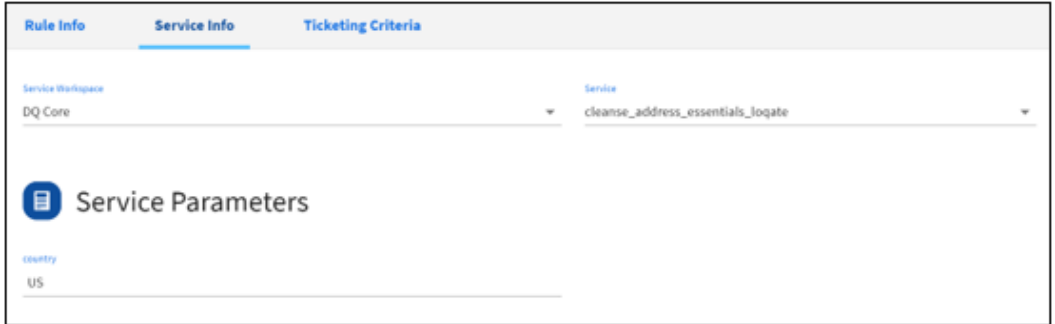
The screenshot shows a configuration form for a rule. The 'Rule Info' tab is active. The form contains the following fields and values:

- Name:** my_us_address
- Active:** True
- Description:** my rule for cleansing US addresses and creating tickets when more than 50% addresses are invalid
- Content Groups:** contact
- Data Classes:** address_city, address_country, address_line +1
- Entities:** UNITED STATES
- Industries:** (empty)

- A unique name for the rule.
- Set Active status to *True* if you want to publish the rule for active use.

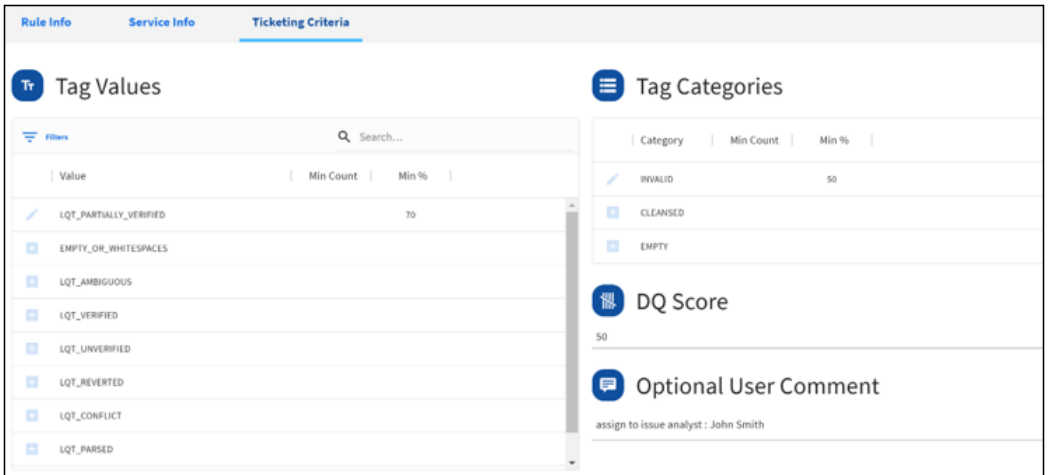
Note: If you accept the default status as *False*, then the rule cannot be used for DQ analysis.
- A brief description of the rule.
- Add rule metadata (optional):
 - Content Group.** Select from the list of available values or add a new value. Content groups are a higher level classification of data classes. For example, data classes *email* and *phone_number* belong to content group **contact**, *us_ssn* and *us_dea* belong to content group **identifier**.
 - Data Class.** Select a data class from the list of available data classes.
 - Entity.** Select one or more values from the list of available entities or add a new value.
 - Country.** Select one or more countries from the list of available countries.
 - Industry.** Select one or more industries from the list of available industries.

3. Enter service information.



- Select workspace.** *DQ Core* represents prebuilt services delivered with the product. *Custom* represents services registered by your developers in this ibi Data Quality instance.
- Select service.** Search and select the appropriate service. Depending on the service you select, you will be presented with service parameters to set.
- Service parameters.** These are service-specific parameters. Refer to service parameter definitions on how to configure them.

4. Enter ticketing criteria to define scenarios where a ticket can be generated for data quality issues that require manual intervention. If multiple criteria are met, only one ticket is created that includes a description of each issue identified during DQ analysis.



Note: Tickets generated during runtime execution of a Rule are stored in the analysis results folder in a file called *tickets.json*. The calling application is responsible for processing the ticket through a ticket management workflow.

- a. Tag based criteria - You can select one or more tag values and specify a minimum threshold (number or percentage of records). When records in a data meet this minimum threshold for this tag value, a data quality ticket will be generated with the description of the issue identified through this criteria.



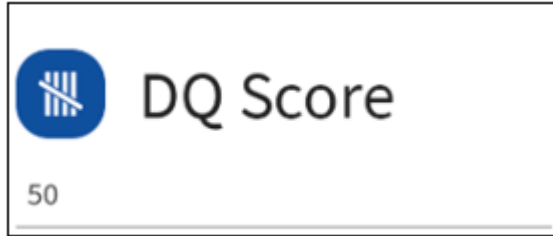
The screenshot shows a dialog box titled "Edit Tag Value" with a close button (X) in the top right corner. It contains two input fields: "Tag Value" with the text "LQT_AMBIGUOUS" and "Min Percent *" with the value "5". To the right of the "Min Percent *" field is a radio button selector with "# %" options. At the bottom right, there are "Cancel" and "Save" buttons.

- b. Tag Category based criteria - You can select one or more tag categories and specify a minimum threshold (number or percentage of records). When the records in a data set meet this minimum threshold for this tag category, a data quality ticket will be generated with the description of the issue identified through this criteria.



The screenshot shows a dialog box titled "Edit Tag Category" with a close button (X) in the top right corner. It contains two input fields: "Tag Category" with the text "INVALID" and "Min Percent *" with the value "50". To the right of the "Min Percent *" field is a radio button selector with "# %" options. At the bottom right, there are "Cancel" and "Save" buttons.

- c. Score based criteria - You can specify a target DQ score between 1 and 100. When records in a data set do not meet this minimum DQ score, a data quality ticket will be generated with the description of the issue identified through this criteria.



- d. Optional User Comment - You can provide an optional comment for the data governance application or the end user who is going to resolve this data quality issue.
5. Click *Finish* and verify the rule is available for use in the catalog.

Verifying a New Rule

To verify a new rule:

1. Go to the *Rules* tab (Rules Catalog) and verify that the rule is available for use in the catalog. Ensure the active flag is set to *True*, as shown in the following image.

Rule Name	Metadata	Business Context
 my_us_address	Service: cleanse_address_essentials_loqate Service Workspace: dqcore CreatedBy: dqadmin Created Date: 03/14/23 10:54 AM Active: True	Description: my rule for cleansing US addresses and creating tickets when more than 50% addresses are invalid Content Groups: contact Data Classes: address_city, address_country, address_line, address_postal_code Countries: usa Entities: Industries:

2. Upload data and attach the rule to a data variable to execute the rules-based DQ analysis, as shown in the following image.

Input Variables

Variable Name	Mapped Rules
zip	Group: primary_address Mapped to: in_address_postal
address	Group: primary_address Mapped to: in_address_line1
spending_score	
city	Group: primary_address Mapped to: in_address_city

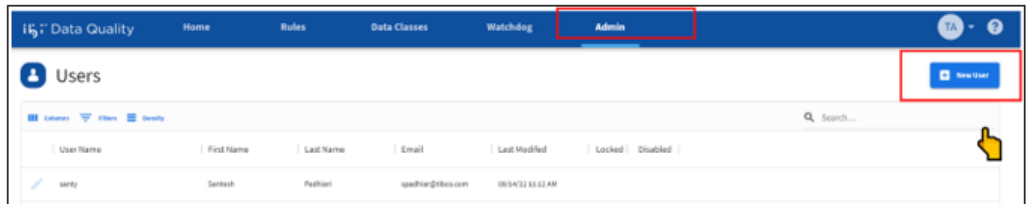
Output Mapping

Mapped Variables	Rule Name
in_address_line1: address in_address_line2: in_address_city: city in_address_state: state in_address_postal: zip	my_us_address Group: primary_address

Adding New Users

To add new users:

1. Go to the *Admin* tab and click *New User*, as shown in the following image.



The New User dialog opens, as shown in the following image.

A screenshot of the 'New User' dialog box. The dialog has a title bar with a close button (X) and three tabs: 'Basic Details' (selected), 'User Groups', and 'Summary'. The 'Basic Details' tab contains several input fields: 'User Name *', 'Email *', 'First Name *', 'Last Name *', 'Password', and 'Confirm Password'. Each field is accompanied by a horizontal line for text entry.

2. Enter the following information about the new user account:

- a. Unique username for the account.
- b. Email address of the user.
- c. First Name and Last Name of the user.
- d. Default password for the account.

The password must meet the following requirements:

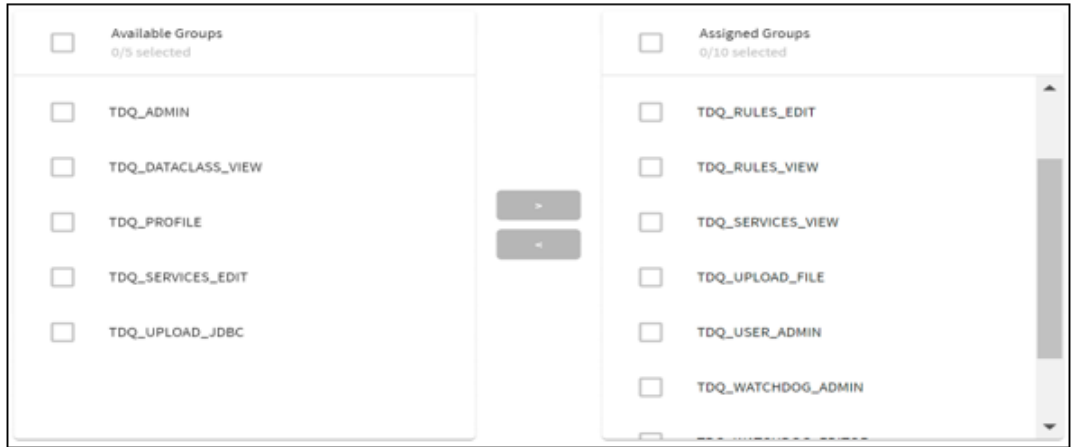
- Must be minimum 8 characters and maximum 20 characters long.
- Must contain one uppercase letter.
- Must contain one lowercase letter.
- Must contain one numeric character.

- ❑ Must contain one special character.
3. Assign one or more groups for the user account. The available groups are listed in the following table:

Group	Privilege
TDQ_ADMIN	Super admin, has access to every ibi Data Quality feature.
TDQ_USER_ADMIN	Can access the Admin tab to manage user accounts.
TDQ_DATACLASS_EDIT	Can access the Data Classes tab and can add definitions for new data classes. Can add/edit data classes through the Valet API.
TDQ_DATACLASS_VIEW	Can access the Data Classes tab, but cannot add new data classes. Can read available data classes through the Valet API.
TDQ_SERVICES_EDIT	Can register new services through the Valet API.
TDQ_SERVICES_VIEW	Can read list of registered services through the Valet API.
TDQ_RULES_EDIT	Can access the Rules tab and can add new rules. Can add/edit data rules through the Valet API.
TDQ_RULES_VIEW	Can access the Rules tab, but cannot add new rules. Can read available rules through the Valet API.

Group	Privilege
TDQ_UPLOAD_FILE	<p>Can access the Home tab, Explore New Data, and only upload data through File Upload.</p> <p>Can upload data files through the Valet API.</p>
TDQ_UPLOAD_JDBC	<p>Can access the Home tab, Explore New Data, and only upload data from JDBC data source.</p> <p>Can upload data from JDBC data sources through the Valet API.</p>
TDQ_PROFILE	<p>Can run the Add profile job to generate data profiles.</p> <p>Can execute data profile job through the Valet API.</p>
TDQ_ANALYZE	<p>Can run the Add rules and run analysis job to generate data quality reports.</p> <p>Can execute rules-based DQ analysis through the Valet API.</p>
TDQ_WATCHDOG_EDIT	<p>Can access the Watchdog tab and manage reports, dashboard, alerts, and notifications on the Watchdog App.</p>
TDQ_WATCHDOG_VIEW	<p>Can access the Watchdog tab and view available dashboards.</p>

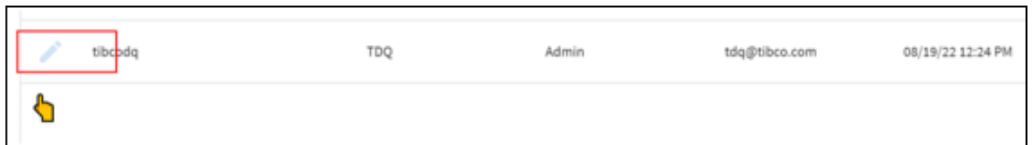
For example:



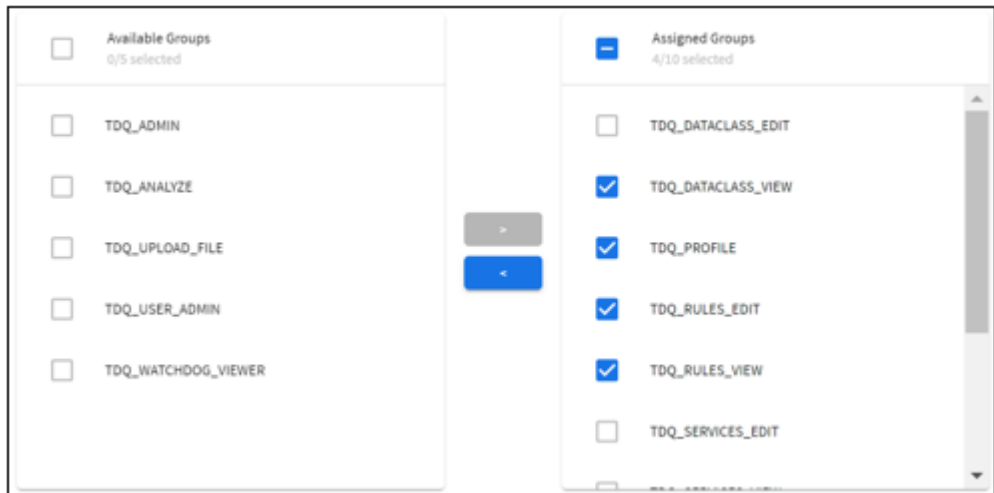
Managing Existing Users

To manage existing users:

1. Go to the *Admin* tab and click the *edit* icon next to the user name, as shown in the following image.



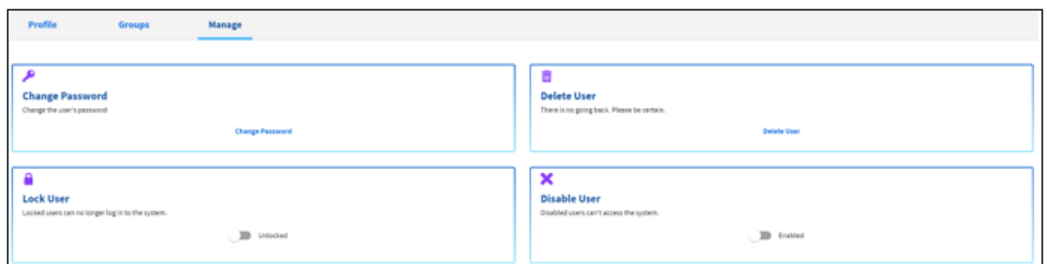
2. Click *Groups* to reassign user roles.



Note: Group changes are not effective until the user logs out and logs back in.

3. Click *Manage* to manage the user account and perform one or more of the following operations:

- Change password of the user account.
- Permanently delete the user account.
- Temporarily lock the user account so the user is forced to reset the password for that account.
- Disables the user account preventing the user from logging in to the system.



Recommended Groups by User Roles

The following table lists recommended groups by user roles.

Role	Interaction	Suggested Groups
Analyst	These users will upload data, profile the data, find rules in the catalog, assign rules to data variables, run DQ analysis, review analysis reports, and monitor metrics on Watchdog.	TDQ_UPLOAD_FILE, TDQ_UPLOAD_JDBC or both TDQ_PROFILE TDQ_DATACLASS_VIEW TDQ_RULES_VIEW TDQ_ANALYZE TDQ_WATCHDOG_VIEWER
Data Experts	These users can do all of the above plus add new data class definitions and add new rules.	TDQ_UPLOAD_FILE, TDQ_UPLOAD_JDBC or both TDQ_PROFILE TDQ_RULES_VIEW TDQ_ANALYZE TDQ_WATCHDOG_VIEWER + TDQ_DATACLASS_EDIT TDQ_RULES_EDIT

Role	Interaction	Suggested Groups
Data Stewards	These users can do all of the above plus create new panels and dashboards on Watchdog.	TDQ_UPLOAD_FILE, TDQ_UPLOAD_JDBC or both TDQ_PROFILE TDQ_RULES_VIEW TDQ_ANALYZE TDQ_WATCHDOG_VIEWER TDQ_DATACLASS_EDIT TDQ_RULES_EDIT + TDQ_WATCHDOG_EDITOR
Watchdog Admin	These users will administer the Watchdog App (add data sources, configure notification channels).	TDQ_WATCHDOG_ADMIN

Monitoring Data Quality Metrics

The ibi Data Quality WatchDog application provides users with meaningful insights on their Data Quality (DQ) metrics. Watchdog users can create new panels and dashboards with pre-configured metrics and KPIs, setup notification channels, and create alerts for monitoring metrics against user defined thresholds.



To learn more about creating reports, dashboards, and notification channels on Grafana, see the following website:

<https://grafana.com/docs/grafana/latest/whatsnew/whats-new-in-v7-5/>

DQ Metrics Views

This section describes all of the DQ metric views that are accessible through the Watchdog app.

Profile Summary

View Name

watchdog_pstats_summ

Description

This view contains the summary stats and metrics generated by the Profiler for an input data set. This view will have one row for each data set profiled in ibi Data Quality.

Column	Description
id	Unique identifier for the profile summary record.
dataset_id	Unique identifier for the input data set.
created_by	User account that executed the data profile.
created_date	Date when the data profile report is generated.
source_name	User assigned source name for the data set.
source_type	User assigned source type for the data set.
app_name	User assigned app name for the data set.
industry	User assigned industry associated with the data set.
entity	User assigned entity associated with the data set.
count_variables	Total number of data attributes in the data set.
count_observations	Total number of observations in the data set (row times column).
count_rows	Total number of rows in the data set.
count_dup_rows	Total number of duplicate rows in the data set.
pct_dup_rows	Percentage of duplicate rows in the data set.

Column	Description
count_dedup_rows	Total number of rows that were removed during data dedup.
pct_dedup_rows	Percentage of of rows that were removed during data dedup.
count_missing	Total number of observations that are empty, blank or nulls.
pct_missing	Percentage of observations that are empty, blank or nulls.
profile_score	Profile score of the data set calculated using the profiling stats and user's expectations of the data.

Profile Details

View Name

watchdog_pstats_dtl

Description

This view contains the detailed stats and metrics generated by the Profiler for an input data set. This view will have one row for each data attribute in the data set that is profiled in ibi Data Quality.

Column	Description
id	Unique identifier for the profile detail record.
pstats_summ_id	Unique identifier for the profile summary record.
dataset_id	Unique identifier for the input data set.
created_by	User account that executed the data profile.

Column	Description
created_date	Date when the data profile report is generated.
source_name	User assigned source name for the data set.
source_type	User assigned source type for the data set.
app_name	User assigned app name for the data set.
industry	User assigned industry associated with the data set.
entity	User assigned entity associated with the data set.
variable_name	Name of the data attribute.
type	Data type of the attribute.
content_type	Data class of the attribute.
sensitive_flag	True if the data attribute is classified a Sensitive data, otherwise False.
expect_nulls	User assigned value set to True if values in the data attribute can be NULL.
expect_unique	User assigned value set to True if values in the data attribute is expected to be unique.
business_impact	User assigned value set to HIGH, MEDIUM, or LOW indicating business impact of the data attribute.
profile_score	Profile score of the data attribute calculated using the profiling stats and user's expectations of the data.
count_total	Total number of observations for the data attribute.

Column	Description
count_unique	Total number of unique observations for the data attribute.
count_non_unique	Total number of non-unique observations for the data attribute.
count_distinct	Total number of distinct observations for the data attribute.
count_blanks	Total number of blank values or empty strings in the data attribute.
count_nulls	Total number of NULL values in the data attribute.
count_not_nulls	Total number of non-NULL values in the data attribute.
length_min	Minimum character length for the values in the data attribute.
length_max	Maximum character length for the values in the data attribute.
length_avg	Average character length for the values in the data attribute.
length_median	Median character length for the values in the data attribute.
most_common_pattern	Most common pattern for the values in the data attribute.
count_pattern	Total count of the values with the most common pattern in the data attribute.
pct_pattern	Percentage of the values with the most common pattern in the data attribute.
most_common_mask	Most common mask for the values in the data attribute.

Column	Description
count_mask	Total count of the values with the most common mask in the data attribute.
pct_mask	Percentage of the values with the most common mask in the data attribute.
num_avg	Average value calculated from all the numeric values in the data attribute.
num_min	Minimum value calculated from all the numeric values in the data attribute.
num_max	Maximum value calculated from all the numeric values in the data attribute.
num_std_dev	Standard deviation value calculated from all the numeric values in the data attribute.
num_pctl_25	25th percentile value calculated from all the numeric values in the data attribute.
num_pctl_50	50th percentile value calculated from all the numeric values in the data attribute.
num_pctl_75	75th percentile value calculated from all the numeric values in the data attribute.

DQ Summary

View Name

watchdog_dqstats_summ

Description

This view contains the summary stats and metrics generated by executing Rules against an input data set. This view will have one row for each set of Rules analyzed against an input data set in ibi Data Quality.

Column	Description
dq_summ_id	Unique identifier for the data quality summary record.
input_name	Unique identifier for the input data set.
created_by	User account that executed the Rules against the input data set.
created_date	Date when the data quality report is generated.
source_name	User assigned source name for the data set.
source_type	User assigned source type for the data set.
app_name	User assigned app name for the data set.
industry	User assigned industry associated with the data set.
entity	User assigned entity associated with the data set.
count_processed	Total number of observations processed.
count_invalid	Total number of observations that failed validation and could not be fixed.
count_valid	Total number of observations that contain valid output values.
count_cleansed	Total number of observations that were fixed - standardized, normalized, or enriched, number of cleansed observations are also counted under number of valid observations.

Column	Description
count_missing	Total number of observations that are empty or missing.
pct_valid	Percentage of total observations that failed validation and could not be fixed.
pct_invalid	Percentage of total observations that contain valid output values.
pct_cleansed	Percentage of total observations that were fixed - standardized, normalized, or enriched, number of cleansed observations are also counted under number of valid observations.
pct_missing	Percentage of total observations that are empty or missing.
dq_score	Data quality score of the input data set calculated using the data quality stats and user's expectations of the data.

DQ Details

View Name

watchdog_dqstats_dtl

Description

This view contains the detailed stats and metrics generated by executing Rules against an input data set. This view will have one row for each Rule executed against an input data set in ibi Data Quality.

Column	Description
dq_dtl_id	Unique identifier for the data quality analysis detail record.

Column	Description
dq_summ_id	Unique identifier for the data quality summary record.
variable_group_name	Name of the data attribute or group of data attributes analyzed by a Rule.
input_name	Name of the data set.
created_by	User account that executed the Rules against the input data set.
created_date	Date when the data quality report is generated.
source_name	User assigned source name for the data set.
source_type	User assigned source type for the data set.
app_name	User assigned app name for the data set.
industry	User assigned industry associated with the data set.
entity	User assigned entity associated with the data set.
rule_id	Unique identifier of the Rule.
count_processed	Total number of observations processed.
count_invalid	Total number of observations that failed validation and could not be fixed.
count_valid	Total number of observations that contain valid output values.
count_cleansed	Total number of observations that were fixed - standardized, normalized, or enriched, number of cleansed observations are also counted under number of valid observations.

Column	Description
count_missing	Total number of observations that are empty or missing.
dq_score	Data quality score of the data attribute or group of data attributes calculated using the data quality stats and user's expectations of the data.

DQ Transaction Summary

View Name

watchdog_dqstats_summ

Description

This view contains the summary stats and metrics generated by executing Rules against an input data set. This view will have one row for each set of Rules analyzed against an input data set in ibi Data Quality.

Column	Description
dq_summ_id	Unique identifier for the data quality summary record.
input_name	Unique identifier for the input data set.
created_by	User account that executed the Rules against the input data set.
created_date	Date when the data quality report is generated.
source_name	User assigned source name for the data set.
source_type	User assigned source type for the data set.
app_name	User assigned app name for the data set.

Column	Description
industry	User assigned industry associated with the data set.
entity	User assigned entity associated with the data set.
count_processed	Total number of observations processed.
count_invalid	Total number of observations that failed validation and could not be fixed.
count_valid	Total number of observations that contain valid output values.
count_cleansed	Total number of observations that were fixed - standardized, normalized, or enriched, number of cleansed observations are also counted under number of valid observations.
count_missing	Total number of observations that are empty or missing.
pct_valid	Percentage of total observations that failed validation and could not be fixed.
pct_invalid	Percentage of total observations that contain valid output values.
pct_cleansed	Percentage of total observations that were fixed - standardized, normalized, or enriched, number of cleansed observations are also counted under number of valid observations.
pct_missing	Percentage of total observations that are empty or missing.

Column	Description
dq_score	Data quality score of the input data set calculated using the data quality stats and user's expectations of the data.

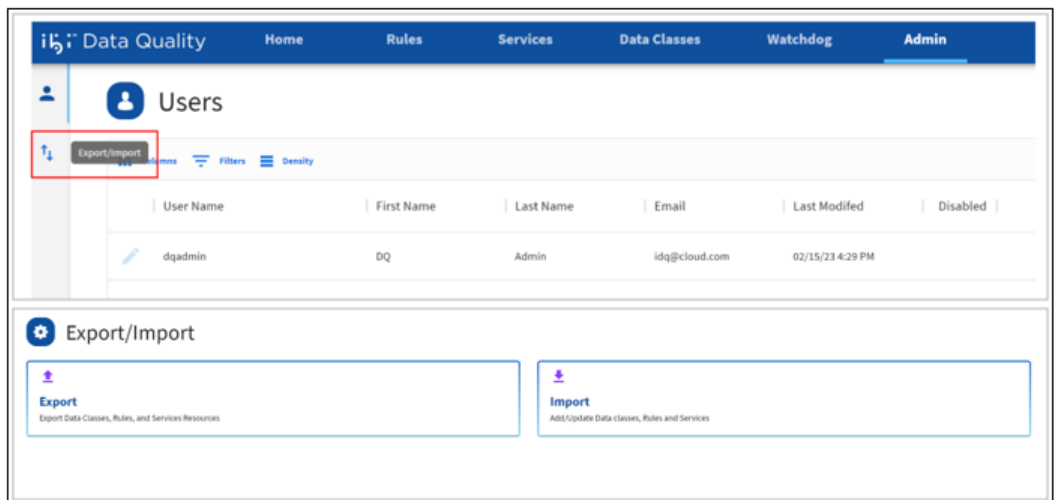
Managing Artifacts

Users who are assigned with the TDQ_ADMIN_ROLE privilege can migrate artifacts (data class definitions, rule definitions, and service definitions) between different environments.

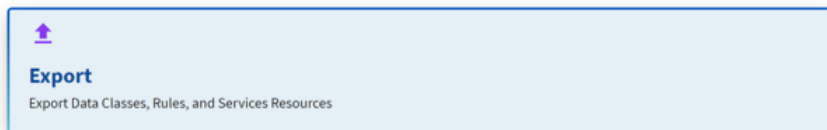
Exporting Artifacts

To export artifacts:

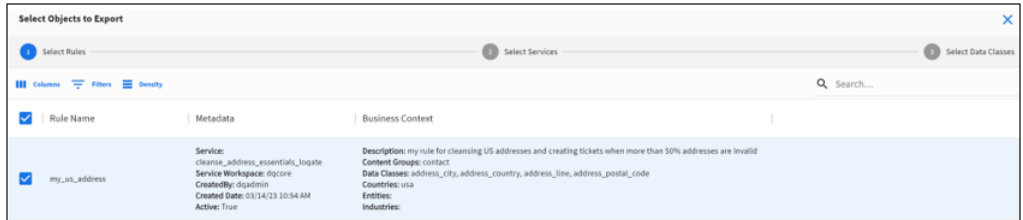
1. Go to the *Admin* tab and click *Export/Import*, as shown in the following image.



2. Click *Export*.



The Select Objects to Export dialog opens, as shown in the following image.



3. Select the Rules, Services, and Data Classes that you want to export.

Note: Users can only select custom artifacts created in their environment.

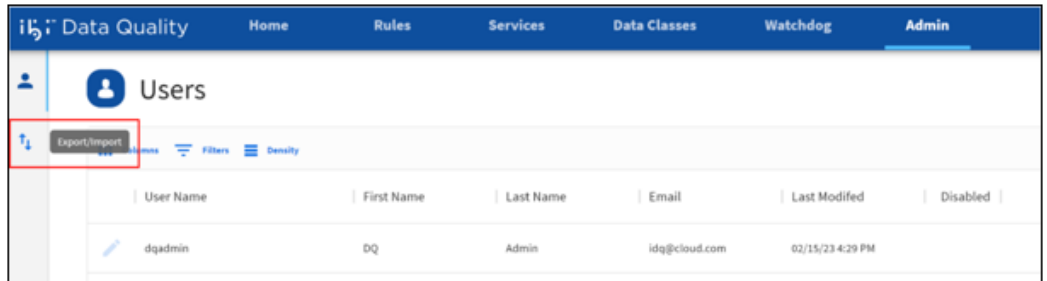
4. When you click *Finish*, the selected artifacts are downloaded to your local machine as a .zip file.



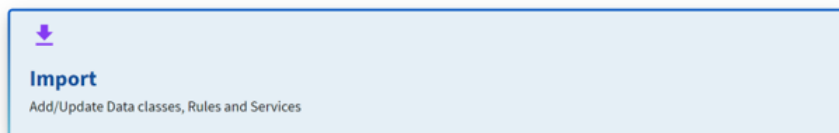
Importing Artifacts

To import artifacts:

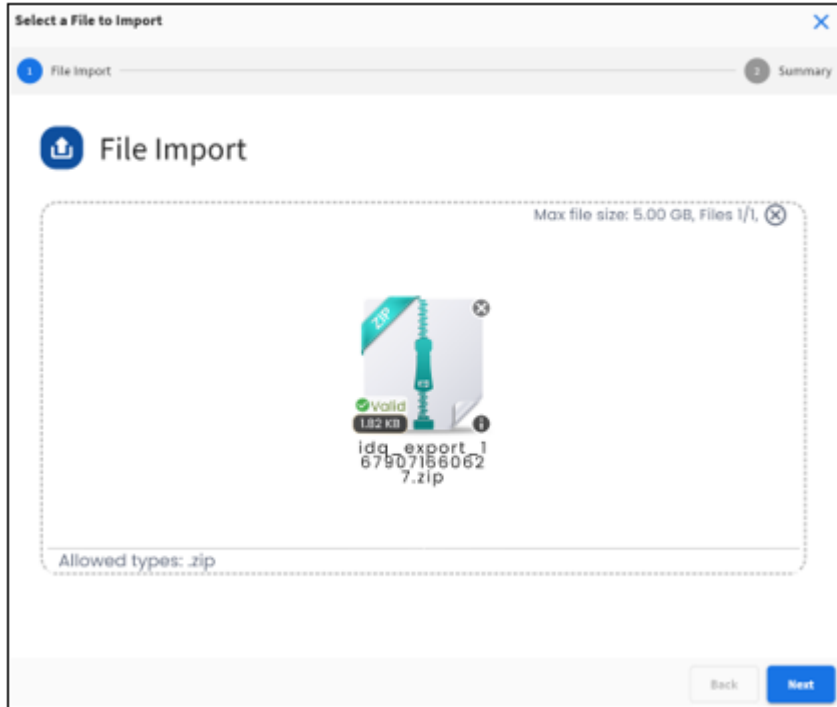
1. Go to the *Admin* tab and click *Export/Import*, as shown in the following image.



2. Click *Import*.



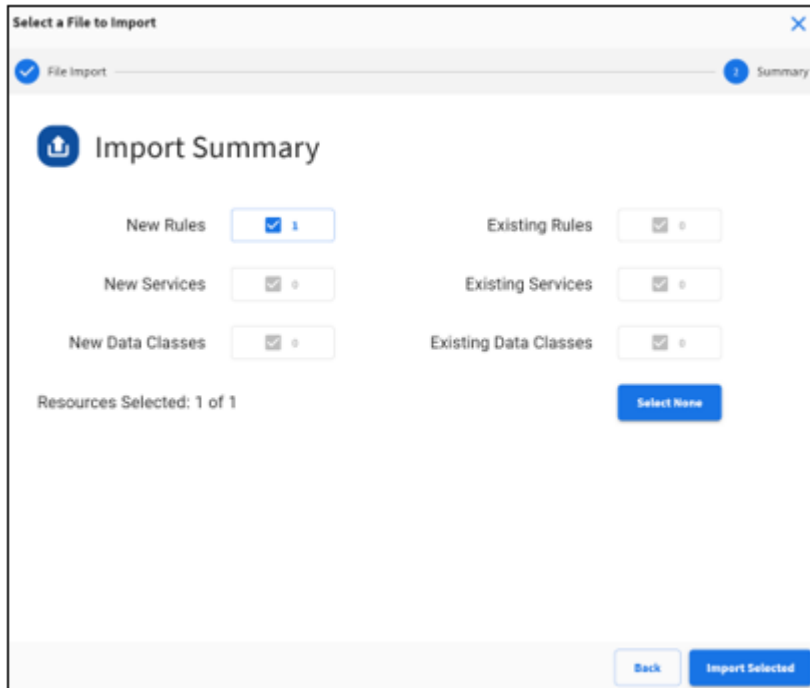
The Select Objects to Import dialog opens, as shown in the following image.



3. Select and upload the .zip file that contains the artifacts you want to import.

Note: You can only upload .zip files that contain valid JSON definitions for ibi Data Quality artifacts.

4. Select options on how you want to import each artifact (add all new artifacts, add new and replace existing artifacts, or just replace existing artifacts).



The screenshot shows a dialog box titled "Select a File to Import" with a close button (X) in the top right corner. The dialog is divided into two tabs: "File Import" (active) and "Summary". Under the "File Import" tab, there is an "Import Summary" section with a blue upload icon. Below this, there are six input fields arranged in two columns. The left column contains "New Rules" (with a checked checkbox and the number "1"), "New Services" (with an unchecked checkbox), and "New Data Classes" (with a checked checkbox). The right column contains "Existing Rules" (with a checked checkbox), "Existing Services" (with an unchecked checkbox), and "Existing Data Classes" (with a checked checkbox). Below these fields, it says "Resources Selected: 1 of 1". There is a blue "Select None" button to the right of this text. At the bottom of the dialog, there are two buttons: "Back" and "Import Selected".

5. Click *Import Selected* to finish importing the artifacts into your environment.

API Interactions

This appendix provides a reference for the ibi Data Quality API.

In this appendix:

- [High Level Flow](#)

High Level Flow

There are two ways of running data quality analysis via the REST API services:

1. Upload and analyze an entire data set.

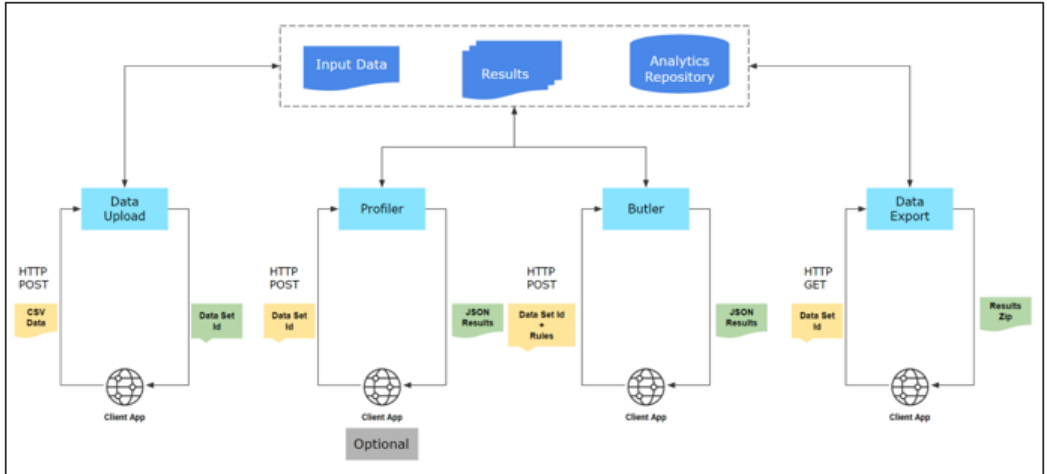
This is the recommended method for processing data in batch mode.

2. Send one or more transaction records at a time.

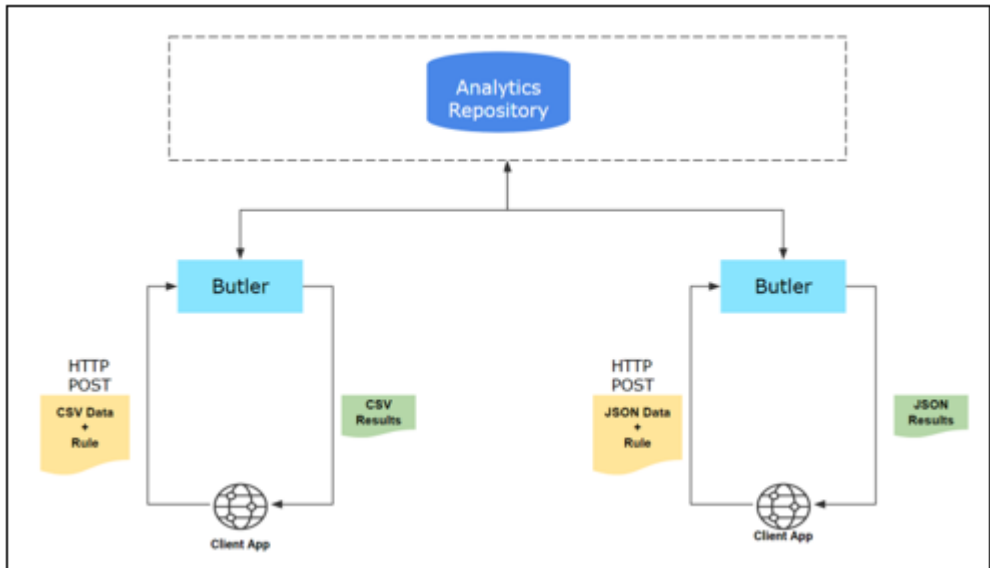
This is the recommended method for streaming data.

Batch Mode	Transaction Mode
Upload and analyze the entire data set.	Send one or more transaction records in a request.
Profile data.	Profile analysis is not applicable for individual records.
Apply multiple rules to different attributes of a data set.	Execute one rule per request.
Calculate Profile and DQ Scores.	Scoring is not applicable for individual transactions.
Save and retrieve input data, detailed results, and summarized reports.	Input requests and responses are not stored in the file system. Summarized results are stored in the transaction details table in the database.

Analyze an entire data set



Analyze one or more records one Rule at a time



Authorize

Description

Use this endpoint to send a request with the user account credentials and receive a response with an access token.

Endpoint

`https://{{host}}:{{port}}/api/v1/auth`

HTTP Method

POST

Request

username	Name of the user account.
password	Password of the user account.

Response

access_token	Access tokens are used in token-based authentication to allow an application to access an API. The application receives an access token after a user successfully authenticates and authorizes access, then passes the access token as a credential when it calls the target API.
refresh_token	Typically, a user needs a new access token after the previous access token granted to them expires. A Refresh Token is a credential artifact that OAuth can use to get a new access token without user interaction. This allows the Authorization Server to shorten the access token lifetime for security purposes without involving the user when the access token expires.
scope	Default: openid The application uses OIDC to verify the user's identity

id_token	ID tokens are used in token-based authentication to cache user profile information and provide it to a client application, thereby providing better performance and experience.
token_type	Default: Bearer A bearer token means that the bearer can access authorized resources without further identification.
expires_in	Default: 3600 seconds

Upload Data Set

Description

Use this endpoint to upload an input data set.

Endpoint

<https://{{host}}:{{port}}/api/v1/valet/upload>

HTTP Method

POST

Authorization

Bearer Token	The <i>access_token</i> received in the authorization response.
--------------	---

Parameters

dataset	Name of the input data set.
charset	Character encoding. Supported values are: UTF-8, UTF-16, or ISO-8859-1

hasHeader	<p>Flag to indicate if the input data set has a header column.</p> <p>Supported values are: true or false</p>
delimiter	<p>Field delimiter.</p> <p>Supported values are:</p> <ul style="list-style-type: none"> <input type="checkbox"/> %2C (for comma) <input type="checkbox"/> %7C (for pipe) <input type="checkbox"/> %20 (for space) <input type="checkbox"/> %09 (for tab) <input type="checkbox"/> %3B (for semicolon)
quoteCharacter	<p>Enclosing character if text within a field also includes the delimiter character.</p> <p>Supported values are:</p> <ul style="list-style-type: none"> <input type="checkbox"/> %22 (for double quote “) <input type="checkbox"/> %60 (for grave accent `)
sourceType	<p>Indicate whether the data source is considered internal or external to the organization.</p> <p>Supported values are: Internal or External</p>
sourceName	Name of the data source.
appName	Name of the application that generated the data.
industry	<p>Select an industry represented by the data.</p> <p>Supported values are: NAICS industry descriptions (see NAICS Industry Classification below)</p>

entity	Name of the business entity the data represents (e.g., customer, partner, supplier, office).
pct	For large data sets, it is recommended to upload data in smaller chunks. Use this parameter to indicate the percentage of data loaded into ibi Data Quality. For example, if there are 10 chunks and you are uploading the second chunk, then set this value to 20.
lastChunk	Use this value to indicate the final chunk of the data. Set this value to false if the request body is not the last chunk of the data set. Set this value to true if the request body is the last chunk of the data set. Supported values are: false or true

NAICS Industry Classification

Agriculture, Forestry, Fishing and Hunting
Mining, Quarrying, and Oil and Gas Extraction
Utilities
Construction
Manufacturing
Wholesale Trade
Retail Trade
Transportation and Warehousing
Information
Finance and Insurance

Real Estate and Rental and Leasing
Professional, Scientific, and Technical Services
Management of Companies and Enterprises
Administrative and Support and Waste Management and Remediation Services
Educational Services
Health Care and Social Assistance
Arts, Entertainment, and Recreation
Accommodation and Food Services
Other Services (except Public Administration)
Public Administration

Request Body

Rows of input data with columns separated by a delimiter.

Response

status	CREATED when the data set is uploaded successfully (or OK when the request is not the last chunk).
code	201 when the data set is uploaded successfully (or 200 when the request is not the last chunk).
message	The ID of the data set to be used in subsequent requests.
developerMessage	UPLOAD_COMPLETE (or UPLOAD_STARTED when the request is not the last chunk).
responsetype	NA

response	<ul style="list-style-type: none"> <input type="checkbox"/> <code>dataSetId</code>: The ID of the data set to be used in subsequent requests. <input type="checkbox"/> <code>status</code>: <code>UPLOAD_COMPLETE</code> (or <code>UPLOAD_STARTED</code> when the request is not the last chunk).
exception	NA

Profile Data

Description

Use this endpoint to request a data profile analysis on a previously uploaded data set.

Endpoint

Synchronous:

<https://{{host}}:{{port}}/api/v1/{{dataset-id}}/profileWithOptions>

Asynchronous:

<https://{{host}}:{{port}}/api/v1/{{dataset-id}}/profileWithOptions/asynch>

HTTP Method

POST

Authorization

Bearer Token	The <code>access_token</code> received in the authorization response.
--------------	---

Parameters

<code>dataset-id</code> (path)	A unique identifier for the data set, this ID should have the same value as the message value received in the response JSON in the Upload Data Set on page 136 step.
--------------------------------	---

Request Body

For each column that should be included in the data profile analysis:

id	Name of the column
businessImpact	<p>A number that represents HIGH, MEDIUM, or LOW.</p> <p>Default values are: HIGH: 10, MEDIUM: 5, LOW: 1</p> <p>Check with your administrator to find the values setup for your implementation.</p>
allowNulls	<p>Indicate whether you expect Null values in this column.</p> <p>Supported values are: true (nulls are expected) or false (nulls are not expected)</p>
shouldBeUnique	<p>Indicate whether you expect column values to be unique.</p> <p>Supported values are: true (values should be unique) or false (values can be non-unique)</p>

Example:

```
[
  {
    "id": "first_name",
    "businessImpact": 10,
    "allowNull": false,
    "shouldBeUnique": true
  },
  {
    "id": "last_name",
    "businessImpact": 10,
    "allowNull": false,
    "shouldBeUnique": true
  },
]
```

Response

status	OK when the data is profiled successfully.
--------	--

code	200 when the data is profiled successfully.
message	This value is the same as the unique identifier for the data set, message value received in the response JSON should match the ID of the data set sent in the request parameter.
developerMessage	NA
responsetype	<code>com.tibco.tdq.common.model.profile.Profile</code>
response	The output data profile in JSON format. For more information, see Profiling Results JSON Schema on page 170.
exception	NA

Check Profile Status

Description

Use this endpoint to check the status of a profile request submitted via the asynchronous endpoint.

Endpoint

Asynchronous:

`https://{host}:{port}/api/v1/valet/{dataset-id}/status/{ref-operation}`

HTTP Method

GET

Authorization

Bearer Token	The <code>access_token</code> received in the authorization response.
--------------	---

Parameters

dataset-id (path)	A unique identifier for the data set, this ID should have the same value as the message value received in the response JSON in the Upload Data Set on page 136 step.
ref-operation (path)	Supported values are: PROFILE

Response

status	OK when the current status is available.
code	200 when the current status is available.
message	NA
developerMessage	NA
responsetype	com.tibco.tdq.valet.services.activity.ActivityStatusDto
response	<ul style="list-style-type: none"> <input type="checkbox"/> id. The data set ID. <input type="checkbox"/> activityName. PROFILE <input type="checkbox"/> progress. A numeric value that represents the percentage of job completion. <input type="checkbox"/> status. Current status of the job. <input type="checkbox"/> startDate. Date time when the job started. <input type="checkbox"/> endDate. Date time when the job completed. <input type="checkbox"/> complete. Flag set to <i>true</i> if the job is complete.

exception	NA
-----------	----

Deduplicate

Description

Use this endpoint to deduplicate rows on a previously uploaded data set.

Authorization

Bearer Token	The <code>access_token</code> received in the authorization response.
--------------	---

Endpoint

<https://{{host}}:{{port}}/api/v1/{{dataset-id}}/deduplicate>

HTTP Method

POST

Parameters

dataset-id (path)	A unique identifier for the data set, this ID should have the same value as the message value received in the response JSON in the Upload Data Set on page 136 step.
-------------------	---

Request Body

NA

Response

status	OK when data deduplication is successful.
code	200 when data deduplication is successful.

message	This value is the same as the unique identifier for the data set, message value received in the response JSON should match the ID of the data set sent in the request parameter.
developerMessage	NA
responsetype	com.tibco.tdq.common.model.profile.Deduplicate
response	<ul style="list-style-type: none"> <input type="checkbox"/> countRowsAfterDeduplication. Number of unique rows in the data set after duplicate rows are removed. <input type="checkbox"/> countDuplicateRowsRemoved. Number of duplicate rows removed. <input type="checkbox"/> pctDuplicateRowsRemoved. Percentage of duplicate rows removed.
exception	NA

Numeric Analysis

Description

Use this endpoint to run numeric analysis on columns that have numeric data.

Authorization

Bearer Token	The <code>access_token</code> received in the authorization response.
--------------	---

Endpoint

<https://{host}:{port}/api/v1/{dataset-id}/profile/numerics>

HTTP Method

POST

Parameters

dataset-id (path)	A unique identifier for the data set, this ID should have the same value as the message value received in the response JSON in the Upload Data Set on page 136 step.
-------------------	---

Request Body

An array of strings with column names enclosed in double-quotes. For example:

```
[
  "age",
  "billed",
  "paid"
]
```

Response

status	OK when the numeric data is profiled successfully.
code	200 when the numeric data is profiled successfully.
message	This value is the same as the unique identifier for the data set, message value received in the response JSON should match the ID of the data set sent in the request parameter.
developerMessage	NA
responsetype	<code>com.tibco.tdq.common.model.profile.Deduplicate</code>
response	The new data profile in JSON format. For more information, see Profiling Results JSON Schema on page 170.
exception	NA

Correlation Analysis

Description

Use this endpoint to run correlation analysis on columns that have numeric, date, or categorical data.

Authorization

Bearer Token	The <code>access_token</code> received in the authorization response.
--------------	---

Endpoint

<https://{{host}}:{{port}}/api/v1/{{dataset-id}}/correlations>

HTTP Method

POST

Parameters

dataset-id (path)	A unique identifier for the data set, this ID should have the same value as the message value received in the response JSON in the Upload Data Set on page 136 step.
count	Specify the number of rows for correlation analysis. Note: Max number of observations that can be uploaded for correlations is 500,000. Adjust the number of data attributes and the row count to reduce the total number of observations if it exceeds the max limit.

correlation	<p>Specify the correlation method. You can specify more than one method by adding multiple correlation parameters (up to 3 max).</p> <p>Supported values are: kendall, pearson, spearman</p>
-------------	--

Request Body

An array of strings with column names enclosed in double-quotes. For example:

```
[
  "age",
  "spending_score",
  "billed"
]
```

Response

status	OK when the correlation analysis is successful.
code	200 when the correlation analysis is successful.
message	This value is the same as the unique identifier for the data set, message value received in the response JSON should match the ID of the data set sent in the request parameter.
developerMessage	NA
responsetype	NA

<p>response</p> <p>(An array of response values in JSON format - one for each correlation method you chose.)</p>	<ul style="list-style-type: none"> <input type="checkbox"/> method. Name of the correlation method. <input type="checkbox"/> count. Number of rows analyzed. <input type="checkbox"/> variables. List of columns analyzed. <input type="checkbox"/> coefficients. An array of correlation coefficients in the order of columns specified in the request. <input type="checkbox"/> svg. XML representing the correlation chart.
exception	NA

K-Means Clustering Analysis

Description

Use this endpoint to run K-Means clustering analysis on a pair of columns.

Authorization

Bearer Token	The <code>access_token</code> received in the authorization response.
--------------	---

Endpoint

<https://{{host}}:{{port}}/api/v1/{{dataset-id}}/clustering/kmeans>

HTTP Method

POST

Parameters

dataset-id (path)	A unique identifier for the data set, this ID should have the same value as the message value received in the response JSON in the Upload Data Set on page 136 step.
clusters	Specify the number of clusters you expect in the data set (default is set to 3, min allowed is 1, and max allowed is 25).
count	Specify the number of rows for clustering analysis (max allowed is 4000).
sampling	Specify the sampling order (Top or Bottom). If your data set has more than 4000 rows, then this order determines the top or bottom 4000 rows sampled for analysis. Supported values are: head or tail

Request Body

An array of two column names enclosed in double-quotes, the first column represents x-axis and the second column represents y-axis.

```
[
  "age",
  "spending_score"
]
```

Response

status	OK when K-Means clustering analysis is successful.
code	200 when K-Means clustering analysis is successful.

message	This value is the same as the unique identifier for the data set, message value received in the response JSON should match the ID of the data set sent in the request parameter.
developerMessage	NA
responsetype	NA
response (Response values in JSON format.)	<input type="checkbox"/> method. Name of the method used for clustering analysis. <input type="checkbox"/> count. Number of rows analyzed. <input type="checkbox"/> variables. List of columns analyzed. <input type="checkbox"/> svg. XML representing the clustering chart.
exception	NA

Download Profile

Description

Use this endpoint to download the profile for a previously analyzed data set.

Authorization

Bearer Token	The <i>access_token</i> received in the authorization response.
--------------	---

Endpoint

<https://{{host}}:{{port}}/api/v1/valet/{{dataset-id}}/profile/export/>

HTTP Method

GET

Parameters

dataset-id (path)	A unique identifier for the data set, this ID should have the same value as the message value received in the response JSON in the Upload Data Set on page 136 step.
-------------------	---

Request Body

NA

Response

The requested data set profile in JSON format. For more information, see [Profiling Results JSON Schema](#) on page 170.

Download Correlation Analysis Results**Description**

Use this endpoint to download correlation results for a previously analyzed data set.

Authorization

Bearer Token	The <code>access_token</code> received in the authorization response.
--------------	---

Endpoint

<https://{{host}}:{{port}}/api/v1/{{dataset-id}}/{{artifact-correlations}}/export>

HTTP Method

GET

Parameters

dataset-id (path)	A unique identifier for the data set, this ID should have the same value as the message value received in the response JSON in the Upload Data Set on page 136 step.
artifact-correlations	Supported values are: correlations

Request Body

NA

Response

<p>response</p> <p>(An array of response values in JSON format, one for each correlation method you chose. For more information, see Correlation Analysis Results JSON Schema on page 177.)</p>	<ul style="list-style-type: none"> <input type="checkbox"/> method. Name of the correlation method. <input type="checkbox"/> count. Number of rows analyzed. <input type="checkbox"/> variables. List of columns analyzed. <input type="checkbox"/> coefficients. An array of correlation coefficients in the order of columns specified in the request. <input type="checkbox"/> svg. XML representing the correlation chart.
---	--

Download K-Means Clustering Results**Description**

Use this endpoint to download K-Means clustering analysis results for a previously analyzed data set.

Authorization

Bearer Token	The <code>access_token</code> received in the authorization response.
--------------	---

Endpoint

`https://{{host}}:{{port}}/api/v1/{{dataset-id}}/{{artifact-kmeans}}/export`

HTTP Method

GET

Parameters

dataset-id (path)	A unique identifier for the data set, this ID should have the same value as the message value received in the response JSON in the Upload Data Set on page 136 step.
artifact-kmeans (path)	Supported values are: kmeans

Request Body

NA

Response

<p>response</p> <p>(Response values in JSON format. For more information, see K-Means Cluster Analysis Results JSON Schema on page 176.)</p>	<ul style="list-style-type: none"> <input type="checkbox"/> method. Name of the method used for clustering analysis. <input type="checkbox"/> count. Number of rows analyzed. <input type="checkbox"/> variables. List of columns analyzed. <input type="checkbox"/> svg. XML representing the clustering chart.
--	--

Get List of Rules

Description

Use this endpoint to download a list of Rules available in that instance of ibi Data Quality.

Authorization

Bearer Token	The <code>access_token</code> received in the authorization response.
--------------	---

Endpoint

`https://{{host}}:{{port}}/api/v1/rules`

HTTP Method

GET

Parameters

status	Supported values are: ACTIVE or INACTIVE
--------	--

Request Body

NA

Response

status	OK when retrieving the list of rules is successful.
code	200 when retrieving the list of rules is successful.
message	NA
developerMessage	NA
responsetype	<code>com.tibco.tdq.common.model.rules.Rule</code>
response (An array of values in JSON format, one per rule.)	Refer to the <i>Rule JSON</i> schema.
exception	NA

Match Rules

Description

Use this endpoint to find Rules that match the input data attributes in a previously generated data profile.

Authorization

Bearer Token	The <code>access_token</code> received in the authorization response.
--------------	---

Endpoint

<https://{{host}}:{{port}}/api/v1/{{dataset-id}}/profilematch/all>

HTTP Method

GET

Parameters

dataset-id (path)	A unique identifier for the data set, this ID should have the same value as the message value received in the response JSON in the Upload Data Set on page 136 step.
-------------------	---

Request Body

NA

Response

status	OK when retrieving the list of matching rules is successful.
code	200 when retrieving the list of matching rules is successful.

message	This value is the same as the unique identifier for the data set, message value received in the response JSON should match the id of the data set sent in the request parameter.
developerMessage	NA
responsetype	<code>java.util.LinkedHashMap</code> <code>\$LinkedValues</code>
response (An array of values in JSON format, one per input data attribute in the data set.)	<ul style="list-style-type: none"> <input type="checkbox"/> name. Name of the input data attribute. <input type="checkbox"/> type. Data type of the input data attribute. <input type="checkbox"/> suggestedRule. Null if no Rules matched the data attribute, or: <ul style="list-style-type: none"> <input type="checkbox"/> ruleName. Name of the rule matched with the data attribute. <input type="checkbox"/> inputMap. Mapping of input data attribute with Rule input. <input type="checkbox"/> ruleId. Unique identifier for the matched Rule.
exception	NA

DQ Analysis With Rules

Description

Use this endpoint to submit a request with Rules to run a data quality analysis against a previously uploaded data set.

Note: API clients do not have to execute a data profile request as a prerequisite for running Rules-based DQ analysis.

Authorization

Bearer Token	The <code>access_token</code> received in the authorization response.
--------------	---

Endpoint

Synchronous:

```
https://{host}:{port}/api/v1/{dataset-id}/butler
```

Asynchronous:

```
https://{host}:{port}/api/v1/{dataset-id}/butler/async
```

HTTP Method

POST

Parameters

dataset-id (path)	A unique identifier for the data set, this ID should have the same value as the message value received in the response JSON in the Upload Data Set on page 136 step.
-------------------	---

Request Body

For each Rule that should be executed in the data quality analysis:

ruleName	Name of the Rule.
groupName	This is only applicable to Rules that require multiple input values. Specify a unique name for a group of data attributes that are mapped to Rule inputs (see the example below).
inputMap	Specify input data attributes that map with the Rule inputs.

ruleId	Unique identifier for the Rule.
--------	---------------------------------

Example:

```
{
  "matches" : [ {
    "ruleName" : "compare_pair_int_values_eq",
    "groupName" : "bill_vs_paid",
    "inputMap" : {
      "in_value_a" : "billed",
      "in_value_b" : "paid"
    },
    "ruleId" : "compare_pair_int_values_eq"
  }, {
    "ruleName" : "compare_pair_int_values_eq",
    "groupName" : "bill_vs_quote",
    "inputMap" : {
      "in_value_a" : "billed",
      "in_value_b" : "quote"
    },
    "ruleId" : "compare_pair_int_values_eq"
  }, {
    "ruleName" : "cleanse_usa_phone",
    "inputMap" : {
      "in_phone" : "phone1"
    },
    "ruleId" : "cleanse_usa_phone"
  }, {
    "ruleName" : "cleanse_email",
    "inputMap" : {
      "in_email" : "email"
    },
    "ruleId" : "cleanse_email"
  } ]
}
```

Response

status	OK when Rules execution is successful.
code	200 when Rules execution is successful.
message	This value is a unique identifier for the Analyze job that was executed. Users can submit multiple combinations of Rules for a given data set, each job will be associated with its unique Analyze Job ID.

developerMessage	This value is the same as the unique identifier for the data set, message value received in the response JSON should match the ID of the data set sent in the request parameter.
responsetype	<i>com.tibco.tdq.common.model.butler.CleansingResults</i>
response	For more information, see DQ Analysis Summary Results JSON Schema on page 178.
exception	NA

Check DQ Analysis Status

Description

Use this endpoint to check the status of a DQ analysis request submitted via the asynchronous endpoint.

Endpoint

Asynchronous:

<https://{{host}}:{{port}}/api/v1/{{dataset-id}}/butler/async>

HTTP Method

GET

Authorization

Bearer Token	The <i>access_token</i> received in the authorization response.
--------------	---

Parameters

dataset-id (path)	A unique identifier for the data set, this ID should have the same value as the message value received in the response JSON in the Upload Data Set on page 136 step.
-------------------	---

Response

status	OK when the current status is available.
code	200 when the current status is available.
message	NA
developerMessage	NA
responsetype	<code>com.tibco.tdq.valet.services.activity.ActivityStatusDto</code>
response	<ul style="list-style-type: none"> <input type="checkbox"/> id. The analyze ID. <input type="checkbox"/> activityName. ANALYZE <input type="checkbox"/> progress. A numeric value that represents the percentage of job completion. <input type="checkbox"/> status. Current status of the job. <input type="checkbox"/> startDate. Date time when the job started. <input type="checkbox"/> endDate. Date time when the job completed. <input type="checkbox"/> complete. Flag set to <i>true</i> if the job is complete.
exception	NA

Download All Results

Description

Use this endpoint to download analysis results for a data set.

Authorization

Bearer Token	The <code>access_token</code> received in the authorization response.
--------------	---

Endpoint

`https://{{host}}:{{port}}/api/v1/valet/analyze/{{analysis-id}}/export`

HTTP Method

GET

Parameters

analysis-id (path)	This is the unique identifier for the previous analysis job. This ID should have the same value as the message value received in the response JSON from the DQ Analysis With Rules on page 157step.
--------------------	---

Request Body

NA

Response

A .zip file that contains the following folders:

1. **input**. Contains the input data set.
2. **profile**. Contains data profiling results.
3. **results**. Contains DQ analysis results.

For more information, see [Viewing Results](#) on page 83.

Download Results Summary

Description

Use this endpoint to download analysis results for a data set.

Authorization

Bearer Token	The <code>access_token</code> received in the authorization response.
--------------	---

Endpoint

`https://{{host}}:{{port}}/api/v1/valet/analyze/{{analysis-id}}/summary/export`

HTTP Method

GET

Parameters

analysis-id (path)	This is the unique identifier for the previous analysis job. This ID should have the same value as the message value received in the response JSON from the DQ Analysis With Rules on page 157step.
--------------------	---

Request Body

NA

Response

The requested data quality analysis results in JSON format. For more information, see [DQ Analysis Summary Results JSON Schema](#) on page 178.

Delete Analysis Results**Description**

Use this endpoint to delete the results of a previous DQ analysis job.

Authorization

Bearer Token	The <code>access_token</code> received in the authorization response.
--------------	---

Endpoint

`https://{{host}}:{{port}}/api/v1/valet/analyze/{{analysis-id}}`

HTTP Method

DELETE

Parameters

analysis-id (path)	This is the unique identifier for the previous analysis job. This ID should have the same value as the message value received in the response JSON from the DQ Analysis With Rules on page 157step.
--------------------	---

Request Body

NA

Response

status	OK when deletion is successful.
code	200 when deletion is successful.
message	NA
developerMessage	NA
responsetype	<code>java.lang.Boolean</code>
response	"true" when deletion is successful.
exception	NA

Delete Entire Dataset

Description

Use this endpoint to delete a previously uploaded data set along with all the analysis results.

Authorization

Bearer Token	The <code>access_token</code> received in the authorization response.
--------------	---

Endpoint

`https://{{host}}:{{port}}/api/v1/valet/{{dataset-id}}`

HTTP Method

DELETE

Parameters

dataset-id (path)	A unique identifier for the data set, this ID should have the same value as the message value received in the response JSON in the Upload Data Set on page 136 step.
-------------------	---

Request Body

NA

Response

status	OK when deletion is successful.
code	200 when deletion is successful.
message	NA
developerMessage	NA
responsetype	<code>java.lang.Boolean</code>
response	"true" when deletion is successful.
exception	NA

Transactional Requests

Description

Use this endpoint to submit CSV or JSON data in the request body and run an analysis with a single Rule, without uploading an entire data set. Results from these transactional requests are not scored. Results are summarized and stored in the *watchdog_dqstats_trans_dtl* view.

Authorization

Bearer Token	The <i>access_token</i> received in the authorization response.
--------------	---

Endpoint

<https://{{host}}:{{port}}/api/v1/cleanseRecords>

HTTP Method

POST

Parameters

ruleId	Unique identifier for the rule that you want to execute.
--------	--

Request Body

- For CSV input, rows of comma-separated values that correspond with the Rule input columns.
- For JSON input, array of name-value pairs that correspond with the Rule input columns.

Response

status	OK when execution is successful.
code	200 when execution is successful.
message	NA
developerMessage	NA

responsetype	<code>java.lang.String</code>
response	<ul style="list-style-type: none"><input type="checkbox"/> For CSV input, response body contains rows of comma-separated values that correspond with Rule output values.<input type="checkbox"/> For JSON input, response contains an array of name-value pairs that correspond to the Rule output columns.
exception	NA

JSON Schemas

This appendix provides a reference for the JSON schema documents used by ibi Data Quality for data analysis results.

In this appendix:

- [Profiling Results JSON Schema](#)
 - [K-Means Cluster Analysis Results JSON Schema](#)
 - [Correlation Analysis Results JSON Schema](#)
 - [DQ Analysis Summary Results JSON Schema](#)
-

Profiling Results JSON Schema

```

{
  "$schema": "https://json-schema.org/draft/2020-12/schema",
  "$id": "https://ibi.com/dq/schemas/profile",
  "title": "Profile",
  "description": "Profile of the dataset",
  "inputDataSetName": {
    "type": "string",
    "description": "Name of the input data set"
  },
  "createdBy": {
    "type": "string",
    "description": "User account that executed profiling analysis and
created the data profile"
  },
  "createdDate": {
    "type": "string",
    "description": "Date and time when the data profile is created"
  },
  "countRows": {
    "type": "integer",
    "description": "Number of rows analyzed"
  },
  "countVariables": {
    "type": "integer",
    "description": "Number of columns analyzed"
  },
  "countObservations": {
    "type": "integer",
    "description": "Number of observations (rows times columns)
analyzed"
  },
  "countDuplicateRows": {
    "type": "integer",
    "description": "Number of duplicate rows in the data set"
  },
  "pctDuplicateRows": {
    "type": "number",
    "description": "Percentage of total rows that are duplicate rows in
the data set"
  },
  "countDuplicateRowsRemoved": {
    "type": "integer",
    "description": "Number of duplicate rows removed in the
deduplication process"
  },
  "pctDuplicateRowsRemoved": {
    "type": "number",
    "description": "Percentage of total rows that were duplicate and
were removed in the deduplication process"
  },
  "countRowsAfterDeduplication": {
    "type": "number",
    "description": "Number of total rows that remain in the data set
after the deduplication process"
  },
}

```

```

"countMissing": {
  "type": "integer",
  "description": "Number of observations that are missing (blank or
null) in the data set"
},
"pctMissing": {
  "type": "number",
  "description": "Percentage of total observations that are missing
(blank or null) in the data set"
},
"profileScore": {
  "type": "number",
  "description": "Data profile score for the entire data set"
},
"variables": {
  "type": "array",
  "items": {
    "type": "object",
    "properties": {
      "name": {
        "type": "string",
        "description": "Name of the variable"
      },
      "type": {
        "enum": [
          "string",
          "number",
          "date"
        ],
        "description": "Data type of the variable"
      },
      "count": {
        "type": "integer",
        "description": "Total number of observations"
      },
      "countDistinct": {
        "type": "integer",
        "description": "Total number of distinct observations,
distinct includes null and blank values"
      },
      "countUnique": {
        "type": "integer",
        "description": "Total number of unique observations,
unique does not include null or blank values"
      },
      "countNonUnique": {
        "type": "integer",
        "description": "Total number of non-unique observations"
      },
      "countNulls": {
        "type": "integer",
        "description": "Total number of null observations"
      },
      "countNotNulls": {
        "type": "integer",
        "description": "Total number of non-null observations"
      }
    }
  }
}

```

```

"countBlanks": {
    "type": "integer",
    "description": "Total number of blank observations"
  },
  "pctUnique": {
    "type": "number",
    "description": "Percentage of total observations that
are unique"
  },
  "pctComplete": {
    "type": "number",
    "description": "Percentage of total observations that
are not null or blank"
  },
  "sensitive": {
    "type": "boolean",
    "description": "true if the variable contains sensitive
data"
  },
  "frequency": {
    "type": "array",
    "description": "Frequency of values",
    "items": {
      "type": "object",
      "properties": {
        "value": {
          "type": "string",
          "description": "Observation value"
        },
        "count": {
          "type": "integer",
          "description": "Number of observations that
has this value"
        },
        "pct": {
          "type": "number",
          "description": "Percentage of total
observations that has this value"
        }
      }
    }
  },
  "lengths": {
    "type": "object",
    "description": "Character length of the observation
values",
    "properties": {
      "min": {
        "type": "integer",
        "description": "Minimum length of all the
observation values"
      },
      "max": {
        "type": "integer",
        "description": "Maximum length of all the
observation values"
      }
    }
  },

```

```

"median": {
    "type": "number",
    "description": "Median length of all the
observation values"
},
"avg": {
    "type": "number",
    "description": "Average length of all the
observation values"
}
},
"patterns": {
    "type": "array",
    "items": {
        "type": "object",
        "description": "Patterns discovered from the
observation values",
        "properties": {
            "value": {
                "type": "string",
                "description": "Pattern value"
            },
            "count": {
                "type": "integer",
                "description": "Total number of observation
values that have this pattern"
            },
            "pct": {
                "type": "number",
                "description": "Percentage of total
observation values that have this pattern"
            }
        }
    }
},
"masks": {
    "type": "array",
    "items": {
        "type": "object",
        "description": "Masks discovered from the
observation values",
        "properties": {
            "value": {
                "type": "string",
                "description": "Mask value"
            },
            "count": {
                "type": "integer",
                "description": "Total number of observation
values that have this mask"
            }
        }
    }
},

```

```

    "pct": {
      "type": "number",
      "description": "Percentage of total
observation values that have this mask"
    }
  },
  "contentTypes": {
    "type": "array",
    "description": "Data classes discovered from the
observation values",
    "items": {
      "type": "object",
      "properties": {
        "value": {
          "type": "string",
          "description": "Name of the data class"
        },
        "count": {
          "type": "integer",
          "description": "Total number of observation
values that belong to this data class"
        },
        "pct": {
          "type": "number",
          "description": "Percentage of total
observation values that belong to this data class"
        }
      }
    }
  },
  "numeric": {
    "type": "array",
    "description": "Numerical statistics of the observation
values",
    "items": {
      "type": "object",
      "properties": {
        "variable": {
          "type": "string",
          "description": "Name of the variable"
        },
        "avg": {
          "type": "number",
          "description": "Average value"
        },
        "min": {
          "type": "number",
          "description": "Minimum value"
        }
      }
    }
  }
}

```

```

    "max": {
        "type": "number",
        "description": "Maximum value"
    },
    "stdDev": {
        "type": "number",
        "description": "Standard deviation"
    },
    "pctl25": {
        "type": "number",
        "description": "25th percentile"
    },
    "pctl50": {
        "type": "number",
        "description": "50th percentile"
    },
    "pctl75": {
        "type": "number",
        "description": "75th percentile"
    }
  },
  "profileScore": {
    "type": "number",
    "description": "Profile score for the individual"
  },
  "notNullsSvg": {
    "type": "string",
    "description": "XML representing the completeness chart of the data"
  },
  "variableOptions": {
    "type": "array",
    "items": {
      "type": "object",
      "description": "Data expectations set by the user",
      "properties": {
        "id": {
          "type": "string",
          "description": "Name of the variable"
        },
        "businessImpact": {
          "type": "string",
          "description": "A number that represents the impact (HIGH, MEDIUM or LOW) of the variable on business outcomes"
        }
      }
    }
  }
}

```

K-Means Cluster Analysis Results JSON Schema

```
"allowsNulls": {
  "type": "string",
  "description": "True if null values are expected for
this variable"
},
  "shouldBeUnique": {
    "type": "string",
    "description": "True if only unique values are expected
for this variable"
  }
}
}
}
```

K-Means Cluster Analysis Results JSON Schema

```
{
  "$schema": "https://json-schema.org/draft/2020-12/schema",
  "$id": "https://ibi.com/dq/schemas/cluster_analysis/kmeans",
  "title": "KMeans Cluster Analysis",
  "description": "KMeans cluster analysis of two variables in a given
data set",
  "type": "object",
  "properties": {
    "method": {
      "type": "string",
      "const": "kmeans",
      "description": "Cluster analysis method"
    },
    "count": {
      "type": "integer",
      "description": "Number of rows analyzed"
    },
    "variables": {
      "type": "array",
      "description": "Name of the variables",
      "items": {
        "type": "string",
        "minItems": 2,
        "maxItems": 2,
        "uniqueItems": true
      }
    },
    "svg": {
      "type": "string",
      "description": "Cluster analysis SVG graph"
    }
  }
}
```


Correlation Analysis Results JSON Schema

```

{
  "$schema": "https://json-schema.org/draft/2020-12/schema",
  "$id": "https://ibi.com/dq/schemas/correlations",
  "title": "Correlation Analysis",
  "description": "Correlation analysis of variables in a dataset",
  "type": "array",
  "items": {
    "type": "object",
    "properties": {
      "method": {
        "type": "string",
        "enum": [
          "kendall",
          "pearson",
          "spearman"
        ],
        "description": "Cluster analysis method"
      },
      "count": {
        "type": "integer",
        "description": "Number of rows analyzed"
      },
      "variables": {
        "type": "array",
        "description": "Names of the variables",
        "items": {
          "type": "string",
          "uniqueItems": true
        }
      },
      "coefficients": {
        "type": "array",
        "description": "Correlation coefficients for a pair of
variables",
        "items": {
          "type": "array",
          "items": {
            "type": "number",
            "minimum": -1.0,
            "maximum": 1.0
          }
        }
      },
      "svg": {
        "type": "string",
        "description": "Correlation analysis SVG graph"
      }
    }
  }
}

```

DQ Analysis Summary Results JSON Schema

```

{
  "$schema": "https://json-schema.org/draft/2020-12/schema",
  "id": "urn:jonschema:com:ibi:dq:common:model:butler:CleansingResults",
  "type": "object",
  "description": "Summarized results of Rules driven data quality
analysis",
  "properties": {
    "inputFile": {
      "type": "string",
      "description": "File name and location of the input data set"
    },
    "outputFile": {
      "type": "string",
      "description": "File name and location of the output data set"
    },
    "dqScore": {
      "type": "integer",
      "description": "Data quality score for the data set"
    },
    "results": {
      "type": "array",
      "description": "Results for each DQ Rule executed against a
variable or group of variables",
      "items": {
        "type": "object",
        "id":
"urn:jonschema:com:ibi:dq:common:model:butler:Summary",
        "properties": {
          "fileName": {
            "type": "string",
            "description": "File name and location of the
cleansed output generated for this combination of Variable or Group of
Variables and DQ Rule"
          },
          "ruleId": {
            "type": "string",
            "description": "Unique identifier of the DQ Rule"
          },
          "groupName": {
            "type": "string",
            "description": "Name of the Variable or Group of
Variables"
          },
          "dqScore": {
            "type": "integer",
            "description": "Data quality score for this
combination of Variable or Group of Variables and DQ Rule"
          },
          "tags": {
            "type": "object",
            "additionalProperties": {
              "type": "integer",
              "description": "Tag values generated during the
DQ analysis"
            }
          }
        }
      }
    }
  },
}

```

```

    "countCleansed": {
      "type": "integer",
      "description": "Total number of cleansed values"
    },
    "countMissing": {
      "type": "integer",
      "description": "Total number of missing values"
    },
    "countProcessed": {
      "type": "integer",
      "description": "Total number of processed values"
    },
    "countInvalid": {
      "type": "integer",
      "description": "Total number of invalid values"
    },
    "countValid": {
      "type": "integer",
      "description": "Total number of valid values"
    }
  }
},
"businessContext": {
  "type": "object",
  "id":
"urn:jjsonschema:com:ibi:dq:common:model:dataset:DataSetContext",
  "properties": {
    "id": {
      "type": "string",
      "description": "Name of the variable"
    },
    "businessImpact": {
      "type": "string",
      "description": "A number that represents the impact
(HIGH, MEDIUM or LOW) of the variable on business outcomes"
    },
    "allowsNulls": {
      "type": "string",
      "description": "True if null values are expected for
this variable"
    },
    "shouldBeUnique": {
      "type": "string",
      "description": "True if only unique values are expected
for this variable"
    }
  }
}
}
}
}

```


Creating Python Services

This appendix describes how to create Python services for ibi Data Quality.

In this appendix:

- [Understanding the Service Requirements](#)
- [Creating Your Python Scripts](#)

Understanding the Service Requirements

Services created for ibi Data Quality must comply with the following requirements:

1. Input (a string of CSV data with each row separated by a newline (\n) character).
2. Output (CSV data).
3. Each output row should have a *tag_value* defined, where *tag_value*:
 - Represents a summary of the data defects or other meaningful facts about the data.
 - Should not contain spaces, or use underscores as word separators.
4. Each output row must have a *tag_category*, where *tag_category*:
 - Summarizes the final outcome of the data analysis.
 - Must contain one of the following supported values:

Value	Description
EMPTY	The input data is an empty string or only contains white spaces.
VALID	The input data is valid and has the same value as the output data.
INVALID	The input value is invalid. Either the defects cannot be fixed or the value does not represent the corresponding class of data.

Value	Description
CLEANSED	The input value is cleansed and a standardized, augmented, or enriched output value is generated.

Note: If a service generates an output *tag_category* with unsupported values, then the service will fail to execute in ibi Data Quality.

Creating Your Python Scripts

After you have reviewed the ibi Data Quality service requirements, you can begin to create your Python scripts, as described in the following sections.

Creating a New Cleanse Service

Create a python script with two function definitions:

1. **Parent** function, where we intend to do the following:
 - a. Parse and read the input CSV data rows.
 - b. Execute the child cleanse function for each row of data.
 - c. Consolidate results into an output result set.
 - d. Transform the output results into CSV format.
 - e. Return the CSV data as results.
2. **Child** function, where we intend to do the following:
 - a. Data prep (trimming white spaces, replacing unexpected characters, etc.).
 - b. Data validation.
 - c. Data standardization and/or normalization.
 - d. Data enrichment.
 - e. Returns output values that include *tag_value* and *tag_category* definitions.



Note: For your convenience, the *my_cleansse_app.zip* file is attached to this PDF, which provides a template for creating new Python cleanse functions. Every section of this code is commented as:

1. **Default.** Copy and reuse this portion of the code.
2. **Custom.** Make changes to this portion of the code as required.

Note: This template does not include code for error handling, logging, and authentication. Python developers are responsible for implementing the coding and engineering best practices applicable to their organization.

Exposing a Cleanse Service as an API

Create a Python script to expose your cleanse routine as a REST API.

1. Create the Flask App instance.
2. Define the route to handle incoming web requests and send responses to the user.

We are using Flask (a lightweight Python web framework) for this tutorial. For production, the Flask program should be invoked using a more robust server (for example, Gunicorn).

Note: This template does not include code for error handling, logging, and authentication. Python developers are responsible for implementing the coding and engineering best practices applicable to their organization.

Testing Your Web Application

Use an API client like Postman to connect and POST test data to your API. For example:

The screenshot shows a Postman interface for a POST request to `http://localhost:7030/cleanse_vin`. The request body is as follows:

```

1 ...
2 1MBGDM9AXKP042788
3 abc
4 | 2A4GM684X6R632476

```

The response status is 200 OK, with a time of 17 ms and a size of 694 B. The response body is a JSON object:

```

1 {"in_vin","out_vin","out_county","out_region","out_manufacturer","out_mmi","out_vds","out_vis","tag_value","tag_category"
2 "","","","","","","","","","EMPTY_OR_WHITESPACES","EMPTY"
3 "1MBGDM9AXKP042788","1MBGDM9AXKP042788","United States","United
4 States","Mercury","1M8","GDM9AX","KP042788","FOUND_VIN_INFO","VALID"
5 "abc","","","","","","","INVALID_VIN_VALIDATION_ERROR","INVALID"
6 " 2A4GM684X6R632476 ","2A4GM684X6R632476","Canada","Canada","Chrysler
7 Canada","2A4","GM684X","6R632476","FOUND_VIN_INFO","CLEANSED"

```

Service Registration

Service Definition JSON

Create a definition for the service in JSON format (an example is included below). Unless otherwise noted, the field is required.

Attribute	Description
name	Name of the service, must be unique in the <i>custom</i> workspace.
description	A brief description of the service.
location	Service location URL.
sendFullDataSet	Optional. Set this to <i>true</i> if the service expects the entire data set as input, as opposed to one row at a time.
supportsJson	Optional. Set this to <i>true</i> if the service can interpret the request body in JSON format and generate a response in JSON format.
inputColumns	An array of paired values that have: <ul style="list-style-type: none"> <input type="checkbox"/> name. Name of the input column. <input type="checkbox"/> description. A brief description of the input column.
outputColumns	An array of paired values that have: <ul style="list-style-type: none"> <input type="checkbox"/> name. Name of the output column. <input type="checkbox"/> description. A brief description of the output column.
parameters	Optional. An array of values that have: <ul style="list-style-type: none"> <input type="checkbox"/> name. Name of the parameter. <input type="checkbox"/> description. A brief description of the parameter.

Attribute	Description
credentials	Optional. A pair of values that have: <ul style="list-style-type: none"> <input type="checkbox"/> user. User name for the service credentials. <input type="checkbox"/> password. Password for the service credentials.

The following is an example of a JSON document for a new service called *cleanse_vin*:

```
{
  "name": "cleanse_vin",
  "description": "Cleanse vehicle identification number",
  "location": "http://service.mysite.com/cleanse_vin",
  "supportsJson": true
  "inputColumns": [
    {
      "name": "in_vin",
      "description": "value to be cleansed or verified"
    }
  ],
  "outputColumns": [
    {
      "name": "out_vin",
      "description": "input value when tag_category is VALID,
      cleansed value when tag_category is CLEANSED, default value when
      tag_category is MISSING or INVALID"
    },
    {
      "name": "tag_value",
      "description": "Tag value that provides explanation for
      malformed, unexpected or missing data"
    },
    {
      "name": "tag_category",
      "description": "Tag category that categorizes tags as Missing
      Data, Cleansed Data or Invalid Data"
    }
  ],
  "tags": [
    {
      "name": "EMPTY_OR_WHITESPACES",
      "description": "Input value is an empty string or contains all
      whitespaces"
    }
  ]
}
```

```

    },
    {
      "name": "INVALID_VIN",
      "description": "Value does not represent a valid email address"
    },
    {
      "name": "VALID_VIN",
      "description": "Input value is Valid"
    },
    {
      "name": "NORMALIZED_VIN",
      "description": "Input value was reformatted or cleansed"
    }
  ],
  "parameters": [
    {
      "name": "default_vin",
      "description": "Default value when tag_category is MISSING or
INVALID"
    }
  ]
}

```

Service Registration Endpoint

To register a new service, POST the service definition JSON (described above) to the following endpoint:

<https://{{host}}:9803/api/v1/service>

The response will contain the ID of the new service.

If a new service is successfully registered, it will be available for Rule authors to create new ibi Data Quality Rules using this new service. For more information, see [Managing Rules](#) on page 104

Packaged DQ Services

This appendix describes the default services packaged with ibi Data Quality in the *DQCore* workspace. Users can define Rules against these services but cannot edit or replace these services.

In this appendix:

- Analyze Text Sentiment
- Cleanse Address Using Loqate
- Cleanse Australian Bank Account Number
- Cleanse Australian Business Number
- Cleanse Australian Phone Number
- Cleanse Date
- Cleanse Email
- Cleanse Email With Loqate
- Cleanse Payment Card
- Cleanse Phone Using Loqate
- Cleanse USA DEA
- Cleanse USA NPI
- Cleanse USA Phone
- Cleanse USA SSN
- Compare Pair Values
- Detect Outliers Using Interquartile Range
- Detect Outliers Using Z-Score
- Impute Missing Numeric Values
- Verify Age
- Verify Value in Range
- Verify Value of Data Type
- Verify With Regular Expression

Analyze Text Sentiment

Name

analyze_text_sentiment

Description

Analyze text sentiment and categorize it as Positive, Neutral, or Negative.

Inputs

in_text	Input text to be analyzed.
---------	----------------------------

Outputs

out_score	Compound score is a metric computed by summing the valence scores of each word in the lexicon, adjusted according to the rules, and then normalized to be between -1 (most extreme negative) and +1 (most extreme positive).
positive_score	Ratios for proportions of text that fall in POSITIVE category.
neutral_score	Ratios for proportions of text that fall in NEUTRAL category.
negative_score	Ratios for proportions of text that fall in NEGATIVE category.
tag_value	POSITIVE if out_score \geq 0.05, NEGATIVE if out_score \leq -0.05, NEUTRAL if out_score is between -0.05 and 0.05
tag_category	Always set to CLEANSED if a score was successfully computed.

Rule Parameters

None

Tags

EMPTY_OR_WHITESPACES	Input value is an empty string or contains all whitespaces.
POSITIVE	Input text has Positive sentiment.
NEUTRAL	Input text has Neutral sentiment.
NEGATIVE	Input text has Negative sentiment.

Cleanse Address Using Loqate

Name

cleanse_address_essentials_loqate

Description

Standardizes, verifies, cleanses, and geocodes address data using Loqate.

Inputs

in_address_line1	Input flat, building, or house number and street name.
in_address_line2	Additional information for the address (e.g., suite number, unit, or other number).
in_address_city	Input name of the city, town, or locality.
in_address_state	Input name of the state / administrative area.
in_address_postal	Input value of the complete postal code for a particular delivery point.

Outputs

out_address_line1	The alphanumeric code identifies an individual location along with the most common street or block data element within a country.
tag_value	Secondary identifiers for a particular delivery point. For instance, FLAT 1 or SUITE 212.
out_address_city	The most common population center data element within a country such as USA city.
out_address_county	Smallest geographic data element within a country such as USA county.

out_address_state	Most common geographic data element within a country such as USA State.
out_address_country_name	ISO 3166 official country name.
out_address_country_code	ISO 3166 2-character country code.
out_address_postal	Complete postal code for a particular delivery point.
out_address_latitude	WGS 84 latitude in decimal degrees format.
out_address_longitude	WGS 84 longitude in decimal degrees format.
out_loqate_avc	Loqate Address Verification Code.
tag_value	Tag value that provides explanation for malformed, unexpected or missing data.
tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data or Invalid Data.

Rule Parameters

country	Country code
---------	--------------

Tags

EMPTY_OR_WHITESPACES	Input address contains empty string or contains all whitespaces.
LQT_PARTIALLY_VERIFIED	Input address could only be partially verified.
LQT_AMBIGUOUS	Input address had more than one close reference data match.
LQT_VERIFIED	Input address had an exact match in the reference data.

LQT_UNVERIFIED	Input address did not match the reference data.
----------------	---

Cleanse Australian Bank Account Number

Name

cleanse_australian_bank_acc

Description

Cleanse Australian bank account number.

Inputs

in_account	Input bank account number.
------------	----------------------------

Outputs

out_account	Input value when tag_category is VALID, cleansed value when tag_category is CLEANSED, default value when tag_category is MISSING or INVALID.
tag_value	Tag value that provides explanation for malformed, unexpected, or missing data.
tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data, or Invalid Data.

Rule Parameters

default_account	Default value when tag_category is MISSING or INVALID.
-----------------	--

Tags

EMPTY_OR_WHITESPACES	Input value is an empty string or contains all whitespaces.
INCORRECT_LENGTH_OR_NOT_AN_ABAN	Input value has incorrect length or does not represent an AUS Bank Account Number.
VALID_BANK_ACCOUNT	Input value is Valid.
CLEANSED_BANK_ACCOUNT	Input value was reformatted or cleansed.

Cleanse Australian Business Number

Name

cleanse_aus_business_number

Description

Cleanse Australian business number.

Inputs

in_abn	Input business number.
--------	------------------------

Outputs

out_abn	Input value when tag_category is VALID, cleansed value when tag_category is CLEANSED, default value when tag_category is MISSING or INVALID.
tag_value	Tag value that provides explanation for malformed, unexpected, or missing data.
tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data, or Invalid Data.

Rule Parameters

default_abn	Default value when tag_category is MISSING or INVALID.
-------------	--

Tags

EMPTY_OR_WHITESPACES	Input value is an empty string or contains all whitespaces.
INCORRECT_LENGTH_OR_NOT_AN_ABN	Input value has incorrect length or does not represent an AUS Business Number.
FAILED_CHECKSUM	Input value failed Modulus 10 checksum.
VALID_ABN	Input value is Valid.
STANDARDIZED_ABN	Input value was reformatted or cleansed.

Cleanse Australian Phone Number*Name*

cleanse_aus_phone

Description

Cleanse Australian phone number.

Inputs

in_phone	Value to be cleansed or verified.
----------	-----------------------------------

Outputs

out_phone_local	Input value when tag_category is VALID, cleansed value when tag_category is CLEANSED, default value when tag_category is MISSING or INVALID.
-----------------	--

Cleanse Date

out_phone_intl	Standardized phone in international format.
tag_value	Tag value that provides explanation for malformed, unexpected, or missing data.
tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data, or Invalid Data.

Rule Parameters

default_phone_local	Default value when tag_category is MISSING or INVALID.
default_phone_intl	Default value when tag_category is MISSING or INVALID.

Tags

EMPTY_OR_WHITESPACES	Input value is an empty string or contains all whitespaces.
FAILED_REGEX_CHECK	Input value failed regular expression check.
VALID_PHONE	Input value is Valid.
STANDARDIZED_PHONE	Input value was reformatted or cleansed.

Cleanse Date

Name

cleanse_date

Description

Cleanse date.

Inputs

in_date	Value to be cleansed or verified.
---------	-----------------------------------

Outputs

out_date	Input value when tag_category is VALID, cleansed value when tag_category is CLEANSED, default value when tag_category is MISSING or INVALID.
date_format	Expected output date format.
tag_value	Tag value that provides explanation for malformed, unexpected, or missing data.
tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data, or Invalid Data.

Rule Parameters

date_format	Expected output date format.
default_date	Default value when tag_category is MISSING or INVALID.

Tags

EMPTY_OR_WHITESPACES	Input value is an empty string or contains all whitespaces.
FAILED_TO_PARSE_DATE	Input value is either not a date value or is not a valid calendar date.
VALID_DATE_IN_CORRECT_FORMAT	Input value is Valid.
VALID_DATE_BUT_INVALID_OUTPUT_FORMAT	Input value is a Valid date, but the date format is Invalid.

REFORMATTED_DATE	Input value was reformatted or cleansed.
------------------	--

Cleanse Email

Name

cleanse_email

Description

Cleanse email addresses.

Inputs

in_email	Value to be cleansed or verified.
----------	-----------------------------------

Outputs

out_email	Input value when tag_category is VALID, cleansed value when tag_category is CLEANSED, default value when tag_category is MISSING or INVALID.
tag_value	Tag value that provides explanation for malformed, unexpected, or missing data.
tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data, or Invalid Data.

Rule Parameters

default_email	Default value when tag_category is MISSING or INVALID.
---------------	--

Tags

EMPTY_OR_WHITESPACES	Input value is an empty string or contains all whitespaces.
INVALID_EMAIL_ADDRESS	Value does not represent a valid email address.
VALID_EMAIL	Input value is Valid.
NORMALIZED_EMAIL	Input value was reformatted or cleansed.

Cleanse Email With Loqate*Name*

cleanse_email_loqate

Description

Cleanse email addresses with Loqate's online email validation service.

Inputs

in_email	Value to be cleansed or verified.
----------	-----------------------------------

Outputs

out_email	Input value when tag_category is VALID, cleansed value when tag_category is CLEANSED, default value when tag_category is MISSING or INVALID.
tag_value	Tag value that provides explanation for malformed, unexpected, or missing data.
tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data, or Invalid Data.

Cleanse Payment Card

Rule Parameters

default_email	Default value when tag_category is MISSING or INVALID.
---------------	--

Tags

EMPTY_OR_WHITESPACES	Input value is an empty string or contains all whitespaces.
COULD_NOT_VERIFY	Value does not represent a valid email address.
DISPOSABLE_TEMPORARY_COMPLAINER	The email address provided is a disposable mailbox.
VALID_EMAIL	Input value is Valid.
NORMALIZED_EMAIL	Input value was reformatted or cleansed.

Cleanse Payment Card

Name

cleanse_payment_card

Description

Cleanse payment card number (credit card, debit card, cash card, etc.).

Inputs

in_card	Value to be cleansed or verified.
---------	-----------------------------------

Outputs

out_card	Input value when tag_category is VALID, cleansed value when tag_category is CLEANSED, default value when tag_category is MISSING or INVALID.
out_card_issuer	Name of the card issuer.
tag_value	Tag value that provides explanation for malformed, unexpected, or missing data.
tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data, or Invalid Data.

Rule Parameters

default_card	Default value when tag_category is MISSING or INVALID.
--------------	--

Tags

EMPTY_OR_WHITESPACES	Input value is an empty string or contains all whitespaces.
COULD_NOT_EXECUTE_LUHN_CHECK	LUHN check could not execute successfully.
FAILED_LUHN_CHECK	Input value failed LUHN check.
VALID_CARD	Input value is Valid.
STANDARDIZED_CARD	Input value was reformatted or cleansed.

Cleanse Phone Using Loqate*Name*

cleanse_phone_loqate

Description

Cleanse phone number using Loqate's online validation service.

Inputs

in_phone	Value to be cleansed or verified.
----------	-----------------------------------

Outputs

out_phone_local	Input value when tag_category is VALID, cleansed value when tag_category is CLEANSED, default value when tag_category is MISSING or INVALID.
out_phone_intl	Standardized phone in international format.
out_phone_country	ISO 2-digit country code.
out_phone_country_code	International dial code for the country
out_phone_type	Type of phone number (MOBILE, LANDLINE, OR VOIP)
network_name	Name of the current operator serving the supplied number.
tag_value	Tag value that provides explanation for malformed, unexpected, or missing data.
tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data, or Invalid Data.

Rule Parameters

default_phone	Default value when tag_category is MISSING or INVALID.
country	ISO 2-digit country code.

Tags

EMPTY_OR_WHITESPACES	Input value is an empty string or contains all whitespaces.
COULD_NOT_VERIFY	Input value is not a valid phone number.
VALID_PHONE	Input value is Valid.
NORMALIZED_PHONE	Input value was reformatted or cleansed.

Cleanse USA DEA*Name*

cleanse_usa_dea

Description

Cleanse USA Drug Enforcement Agency assigned prescriber identifier.

Inputs

in_dea	Value to be cleansed or verified.
in_last_name	Last name of the prescriber.

Outputs

out_dea	Input value when tag_category is VALID, cleansed value when tag_category is CLEANSED, default value when tag_category is MISSING or INVALID.
tag_value	Tag value that provides explanation for malformed, unexpected, or missing data.
tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data, or Invalid Data.

Cleanse USA NPI

Rule Parameters

default_dea	Default value when tag_category is MISSING or INVALID.
-------------	--

Tags

EMPTY_OR_WHITESPACES	Input value is an empty string or contains all whitespaces.
INCORRECT_LENGTH_OR_NOT_A_DEA	Input value has incorrect length or does not represent a US DEA value.
INVALID_1ST_LETTER	First character does not belong to the list of allowed characters.
INVALID_2ND_LETTER	Second character does not match the first character of provider's last name.
INVALID_DEA	Input value does not represent a DEA number.
FAILED_DEA_CHECKSUM	Input value failed DEQ checksum.
VALID_DEA	Input value is Valid.
STANDARDIZED_DEA	Input value was reformatted or cleansed.

Cleanse USA NPI

Name

cleanse_usa_npi

Description

Cleanse USA National Provider Identifier.

Inputs

in_npi	Value to be cleansed or verified.
--------	-----------------------------------

Outputs

out_npi	Input value when tag_category is VALID, cleansed value when tag_category is CLEANSED, default value when tag_category is MISSING or INVALID.
tag_value	Tag value that provides explanation for malformed, unexpected, or missing data.
tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data, or Invalid Data.

Rule Parameters

default_npi	Default value when tag_category is MISSING or INVALID.
-------------	--

Tags

EMPTY_OR_WHITESPACES	Input value is an empty string or contains all whitespaces.
INCORRECT_LENGTH_OR_NOT_A_NPI	Input value has incorrect length or does not represent a US NPI value.
COULD_NOT_EXECUTE_LUHN_CHECK	The regular expression for US NPI could not be compiled.
FAILED_LUHN_CHECK	Input value failed LUHN check.
VALID_CARD	Input value is Valid.
STANDARDIZED_CARD	Input value was reformatted or cleansed.

Cleanse USA Phone*Name*

cleanse_usa_phone

Description

Cleanse USA phone number.

Inputs

in_phone	Value to be cleansed or verified.
----------	-----------------------------------

Outputs

out_phone_local	Input value when tag_category is VALID, cleansed value when tag_category is CLEANSED, default value when tag_category is MISSING or INVALID.
out_phone_intl	Standardized phone in international format.
tag_value	Tag value that provides explanation for malformed, unexpected, or missing data.
tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data, or Invalid Data.

Rule Parameters

default_phone_local	Default value when tag_category is MISSING or INVALID.
coundefault_phone_intl	Default value when tag_category is MISSING or INVALID.

Tags

EMPTY_OR_WHITESPACES	Input value is an empty string or contains all whitespaces.
FAILED_REGEX_CHECK	Input value failed regular expression check.

VALID_PHONE	Input value is Valid.
NORMALIZED_PHONE	Input value was reformatted or cleansed.

Cleanse USA SSN

Name

cleanse_usa_ssn

Description

Cleanse USA Social Security Number.

Inputs

in_ssn	Value to be cleansed or verified.
--------	-----------------------------------

Outputs

out_ssn	Input value when tag_category is VALID, cleansed value when tag_category is CLEANSED, default value when tag_category is MISSING or INVALID.
tag_value	Tag value that provides explanation for malformed, unexpected, or missing data.
tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data, or Invalid Data.

Rule Parameters

default_ssn	Default value when tag_category is MISSING or INVALID.
-------------	--

Tags

EMPTY_OR_WHITESPACES	Input value is an empty string or contains all whitespaces.
INCORRECT_LENGTH_OR_NOT_A_SSN	Input value has incorrect length or does not represent a US SSN value.
FAILED_SSN_REGEX_CHECK	Input value failed regular expression check.
INVALID_AREA_CODE	Input value contains invalid area code.
INVALID_GROUP_CODE	Input value contains invalid group code.
INVALID_SERIAL	Input value contains invalid serial number.
BLACKLISTED_SSN	Input value is blacklisted.
VALID_SSN	Input value is Valid.
STANDARDIZED_SSN	Input value was reformatted or cleansed.

Compare Pair Values

Name

compare_pair_values

Description

Compare two values using a comparison operator.

Inputs

input_value_a	Input value A.
input_value_b	Input value B.

Outputs

out_value_a	Input value when tag_category is VALID, cleansed value when tag_category is CLEANSED, default value when tag_category is MISSING or INVALID.
comp_operator	Comparison operator.
data_type	Expected data type, expected values are str, int, float, complex.
tag_value	Tag value that provides explanation for malformed, unexpected, or missing data.
tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data, or Invalid Data.

Rule Parameters

comp_operator	Comparison operator - expected operators are <, >, <=, >=, ==, !=
default_value	Default value when tag_category is MISSING or INVALID.
data_type	Expected data type, expected values are str, int, float, complex.

Tags

COMPARISON_PASSED	in_value_a passed comparison against in_value_b.
COMPARISON_FAILED	in_value_a failed comparison against in_value_b.
COMPARISON_FAILED_INCOMPATIBLE_TYPES	Input values could not be compared because they are of incompatible types.

Detect Outliers Using Interquartile Range

Name

detect_outlier_iqr

Description

Detects outliers using interquartile range.

Inputs

in_value	Input value.
----------	--------------

Outputs

out_value	Input value when tag_category is VALID, cleansed value when tag_category is CLEANSED, default value when tag_category is MISSING or INVALID
q1	first quartile
q3	third quartile
iqr	interquartile range
upper	upper limit
lower	lower limit
tag_value	Tag value that provides explanation for malformed, unexpected, or missing data.
tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data, or Invalid Data.

Rule Parameters

default_value	Default value when tag_category is MISSING or INVALID.
---------------	--

Tags

NORMAL	Input value is within normal range.
OUTLIER	Input value is abnormal or an anomaly.

Detect Outliers Using Z-Score*Name*

detect_outlier_zscore

Description

Detects outliers using Z-score.

Inputs

in_value	Input value.
----------	--------------

Outputs

out_value	Input value when tag_category is VALID, cleansed value when tag_category is CLEANSED, default value when tag_category is MISSING or INVALID
z_score	z score
z_threshold	z-score threshold
tag_value	Tag value that provides explanation for malformed, unexpected, or missing data.
tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data, or Invalid Data.

Rule Parameters

default_value	Default value when tag_category is MISSING or INVALID.
z_threshold	z-score threshold

Tags

NORMAL	Input value is within normal range.
OUTLIER	Input value is abnormal or an anomaly.

Impute Missing Numeric Values

Name

impute_missing_numeric_values

Description

Identify missing numeric values and replace them with default value using statistical methods.

Inputs

in_value	Input value.
----------	--------------

Outputs

out_value	Input value when tag_category is VALID, cleansed value when tag_category is CLEANSED, default value when tag_category is MISSING or INVALID
tag_value	Tag value that provides explanation for malformed, unexpected, or missing data.

tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data, or Invalid Data.
--------------	---

Rule Parameters

method	Statistical method to use - max, min, mean, median, pctl
pctl	Percentile value if selected method = pctl

Tags

IMPUTED	Input value is missing and output value has been imputed.
ORIGINAL	Output value is same as input value.

Verify Age*Name*

verify_age

Description

Calculate and verify age from birth date.

Inputs

in_birth_date	Input value to be cleansed or verified.
---------------	---

Outputs

out_age	Input value when tag_category is VALID, cleansed value when tag_category is CLEANSSED, default value when tag_category is MISSING or INVALID.
---------	---

Verify Value in Range

tag_value	Tag value that provides explanation for malformed, unexpected, or missing data.
tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data, or Invalid Data.

Rule Parameters

max_age	Maximum expected age.
---------	-----------------------

Tags

EMPTY_OR_WHITESPACES	Input value is an empty string or contains all whitespaces.
BIRTHDATE_IN_FUTURE	Birth date is in future.
GREATER_THAN_MAX_AGE	Calculated age is greater than max expected age.
CALCULATED_AGE_FROM_BIRTHDATE	Successfully parsed date and calculated age.
FAILED_TO_PARSE_BIRTHDATE	Invalid birth date, failed to parse date value.

Verify Value in Range

Name

verify_value_in_range

Description

Verify value is in a defined range.

Inputs

in_value	Input value to be cleansed or verified.
----------	---

Outputs

out_value	Input value when tag_category is VALID, cleansed value when tag_category is CLEANSED, default value when tag_category is MISSING or INVALID.
min_value	Lower value of the range.
max_value	Upper value of the range.
data_type	Expected data type, expected values are str, int, float, complex.
tag_value	Tag value that provides explanation for malformed, unexpected, or missing data.
tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data, or Invalid Data.

Rule Parameters

min_value	Lower value of the range.
max_value	Upper value of the range.
default_value	Default value when tag_category is MISSING or INVALID.
data_type	Expected data type, expected values are str, int, float, complex.

Tags

VALUE_IN_RANGE	Input value is within desired range.
VALUE_OUT_OF_RANGE	Input value is outside desired range.

COMPARISON_FAILED_INCOMPATIBLE_TYPES	Input value data type is incompatible with desired range.
--------------------------------------	---

Verify Value of Data Type

Name

verify_value_of_datatype

Description

Verify value against a known data type.

Inputs

in_value	Input value to be cleansed or verified.
----------	---

Outputs

data_type	Expected data type.
out_value	Input value when tag_category is VALID, cleansed value when tag_category is CLEANSED, default value when tag_category is MISSING or INVALID.
tag_value	Tag value that provides explanation for malformed, unexpected, or missing data.
tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data, or Invalid Data.

Rule Parameters

default_value	Default value when tag_category is MISSING or INVALID.
---------------	--

data_type	Expected data type, expected values are str, int, float, complex.
-----------	---

Tags

VALID_DATA_TYPE	Input value is of expected data type
INVALID_DATA_TYPE	Input value does not match expected data type.

Verify With Regular Expression*Name*

verify_with_regex

Description

Verify value by matching with a regular expression.

Inputs

in_value	Input value to be cleansed or verified.
----------	---

Outputs

regex	Regular expression.
out_value	Input value when tag_category is VALID, cleansed value when tag_category is CLEANSED, default value when tag_category is MISSING or INVALID.
tag_value	Tag value that provides explanation for malformed, unexpected, or missing data.
tag_category	Tag category that categorizes tags as Missing Data, Cleansed Data, or Invalid Data.

Rule Parameters

default_value	Default value when tag_category is MISSING or INVALID.
regex	Regular expression.

Tags

EMPTY_OR_WHITESPACES	Input value is an empty string or contains all whitespaces.
DID_NOT_MATCH_REGEX	Input value failed regular expression check.
MATCHED_REGEX	Input value is Valid because it matched regular expression.
TRIMMED_WHITESPACES	Input value was reformatted or cleansed.

Troubleshooting

This appendix describes how to troubleshoot ibi Data Quality by locating and reviewing system logs.

In this appendix:

- [Locating System Logs](#)
 - [Common Problems](#)
-

Locating System Logs

ibi Data Quality log files are written to a Docker volume, and the location of this volume can vary with the type of Docker installation.

Docker Desktop

In the Docker Desktop user interface, view Volumes using the menu on the left side of the screen, as shown in the following image.



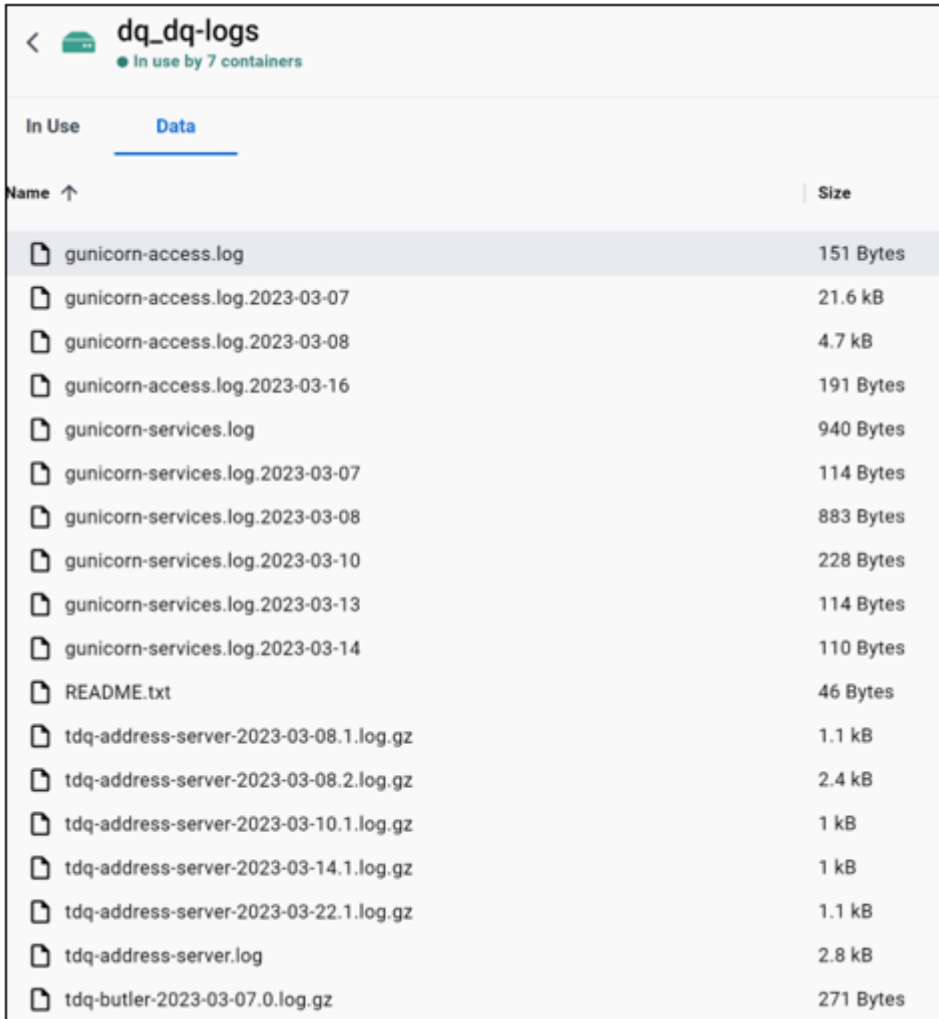
<input type="checkbox"/>	<code>dq_dq-db</code>	in use	less than a minut	82 MB	
<input type="checkbox"/>	<code>dq_dq-grafana-storage</code>	in use	about 4 hours ag-	528 kB	
<input type="checkbox"/>	<code>dq_dq-logs</code>	in use	about 4 hours ag-	220 kB	
<input type="checkbox"/>	<code>dq_dq-storage</code>	in use	15 days ago	7.8 MB	

Click the `dq_dq-logs` volume.

A list of containers in the dq_dq-logs is displayed, as shown in the following image.

CONTAINER NAME	IMAGE	PORT	TARGET
dq-valet-services	dq-dq-valet-services	--	/tdq-valet-services/logs /tdq-storage
dq-butler	dq-dq-butler	--	/tdq-butler/logs /tdq-storage
dq-ksource	dq-dq-ksource	--	/tdq-ksource/logs
dq-dq-profiler-1	dq-dq-profiler	--	/usr/src/tdq-profiler/logs /usr/src/tdq-profiler/target
dq-dqs	dq-dq-dqs	--	/loqate /dqs/server/logs /dqs/server/services
dq-dq-address-server-1	dq-dq-address-server	--	/tdq-address-server/logs /software/loqate
dq-python-services	dq-dq-python-services	--	/app/logs

Navigate to the DATA tab and select a log file. A menu will appear that will allow you to download the file, as shown in the following image.



Name ↑	Size
gunicorn-access.log	151 Bytes
gunicorn-access.log.2023-03-07	21.6 kB
gunicorn-access.log.2023-03-08	4.7 kB
gunicorn-access.log.2023-03-16	191 Bytes
gunicorn-services.log	940 Bytes
gunicorn-services.log.2023-03-07	114 Bytes
gunicorn-services.log.2023-03-08	883 Bytes
gunicorn-services.log.2023-03-10	228 Bytes
gunicorn-services.log.2023-03-13	114 Bytes
gunicorn-services.log.2023-03-14	110 Bytes
README.txt	46 Bytes
tdq-address-server-2023-03-08.1.log.gz	1.1 kB
tdq-address-server-2023-03-08.2.log.gz	2.4 kB
tdq-address-server-2023-03-10.1.log.gz	1 kB
tdq-address-server-2023-03-14.1.log.gz	1 kB
tdq-address-server-2023-03-22.1.log.gz	1.1 kB
tdq-address-server.log	2.8 kB
tdq-butler-2023-03-07.0.log.gz	271 Bytes

If you are using the WSL2 engine with Docker Desktop, then the dq-logs volume can be accessed through the WSL2 file system at:

```
/mnt/wsl/docker-desktop-data/version-pack-data/community/docker/volumes/dq\_dq-logs/\_data
```

You could `tail` the profiler log, for example, as follows:

```
sudo tail -f /mnt/wsl/docker-desktop-data/version-pack-data/community/
docker/volumes/dq_dq-logs/_data/tdq-profiler.log
```

Linux

On Linux, Docker volumes are usually found in `/var/lib/docker/volumes`, so you could `tail` the profiler log as follows:

```
sudo tail -f /var/lib/docker/volumes/dq_dq-logs/_data/tdq-profiler.log
```

Copying Logs From a Container

If the above methods do not work, you can use the Docker `cp` command to retrieve the log files from a running container. First, get the container ID for `dq-butler`:

```
$ docker ps
```

CONTAINER ID	IMAGE NAME	COMMAND	CREATED	STATUS	PORTS
ca67baeee fb6	dq-dq- nginx	"/docker- entrypoint ..."	25 hours ago	Up About a minute	0.0.0.0:9 800- >80/tcp
451417702 13a	dq-dq- valet-ui	"npm start"	25 hours ago	Up About a minute	0.0.0.0:9 801- >81/tcp
79e739ebe d6c	dq-dq- valet- services	"/tdq- valet- services..."	25 hours ago	Up About a minute	0.0.0.0:9 803- >83/tcp
b385822c7 2be	dq-dq- butler	"/tdq- butler/ docker-..."	25 hours ago	Up About a minute	0.0.0.0:9 807- >87/tcp

Then copy the `/logs` directory from the container to the host, using the ID of the `dq-butler` container:

```

$ docker cp b385822c72be:/tdq-butler/logs .

$ ls -l ./logs/
total 48
-rw-r--r-- 1 sdewitt sdewitt 46 Jan 18 2021 README.txt
-rw-r--r-- 1 sdewitt sdewitt 1820 Sep 27 10:46 gunicorn-access.log
-rw-r--r-- 1 sdewitt sdewitt 818 Sep 28 11:30 gunicorn-services.log
-rw-r--r-- 1 sdewitt sdewitt 1474 Sep 27 13:11 gunicorn-services.log.
2022-09-27
-rw-r--r-- 1 sdewitt sdewitt 281 Sep 28 11:31 tdq-
butler-2022-09-27.0.log.gz
-rw-r--r-- 1 sdewitt sdewitt 514 Sep 28 11:31 tdq-butler.log
-rw-r--r-- 1 sdewitt sdewitt 0 Sep 27 10:38 tdq-dqs-console.log
-rw-r--r-- 1 sdewitt sdewitt 0 Sep 28 11:30 tdq-dqs-request.log
-rw-r--r-- 1 sdewitt sdewitt 285 Sep 28 11:30 tdq-
ksource-2022-09-27.0.log.gz
-rw-r--r-- 1 sdewitt sdewitt 530 Sep 28 11:31 tdq-ksource.log
-rw-r--r-- 1 sdewitt sdewitt 90 Sep 28 11:31 tdq-profiler.90bb2ecc9968.log
-rw-r--r-- 1 sdewitt sdewitt 1113 Sep 27 10:43 tdq-profiler.
90bb2ecc9968.log.2022-09-27
-rw-r--r-- 1 sdewitt sdewitt 740 Sep 28 11:31 tdq-valet-
services-2022-09-27.0.log.gz
-rw-r--r-- 1 sdewitt sdewitt 511 Sep 28 11:31 tdq-valet-services.log

```

Common Problems

Problem: The *dq-wso2* container does not start when ibi DQ Docker Compose is initially brought up.

Solution:

A symptom of this is that the login screen does not appear when you attempt to load the Valet UI. The Docker `ps` command or the Docker Desktop Containers screen will show that the *dq-wso2* container is not running.

This usually happens when the WS02 container attempts to start before the PostgreSQL database has initialized. Either restart ibi DQ Docker Compose, without the `-build` switch, or restart the *dq-wso2* container individually.

Legal and Third-Party Notices

SOME TIBCO SOFTWARE EMBEDS OR BUNDLES OTHER TIBCO SOFTWARE. USE OF SUCH EMBEDDED OR BUNDLED TIBCO SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED TIBCO SOFTWARE. THE EMBEDDED OR BUNDLED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER TIBCO SOFTWARE OR FOR ANY OTHER PURPOSE.

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH SOFTWARE LICENSE AGREEMENT OR CLICKWRAP END USER LICENSE AGREEMENT, THE LICENSE(S) LOCATED IN THE "LICENSE" FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of Cloud Software Group, Inc.

TIBCO, the TIBCO logo, the TIBCO O logo, ibi, ibi logo, ActiveMatrix BusinessWorks, TIBCO Administrator, BusinessConnect, TIBCO Designer, Enterprise Message Service, Hawk, and Maporama are either registered trademarks or trademarks of Cloud Software Group, Inc. in the United States and/or other countries.

Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle Corporation and/or its affiliates.

This document includes fonts that are licensed under the SIL Open Font License, Version 1.1, which is available at: <https://scripts.sil.org/OFL>

Copyright (c) Paul D. Hunt, with Reserved Font Name Source Sans Pro and Source Code Pro.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only.

This software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. See the readme file for the availability of this software version on a specific operating system platform.

THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. CLOUD SOFTWARE GROUP, INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "READ ME" FILES.

This and other products of Cloud Software Group, Inc. may be covered by registered patents. Please refer to TIBCO's Virtual Patent Marking document (<https://www.tibco.com/patents>) for details.

Copyright © 2023. Cloud Software Group, Inc. All Rights Reserved.