Oracle® Enterprise Data Quality

Customer Data Services Pack Guide

Release 11g (11.1.1.9)

E56079-01

April 2015

Beta Draft

ORACLE CONFIDENTIAL.

For authorized use only.

Do not distribute to third parties.

Customer Data Services Pack Guide, Release 11g (11.1.1.9)

E56079-01

This documentation is in pre-production status and is intended for demonstration and preliminary use only. It may not be specific to the hardware on which you are using the software. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to this documentation and will not be responsible for any loss, costs, or damages incurred due to the use of this documentation.

The information contained in this document is for informational sharing purposes only and should be considered in your capacity as a customer advisory board member or pursuant to your beta trial agreement only. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle.

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle Software License and Service Agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced, or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

Contents

2 Using Business Services

3 Using Matching

4 Customizing Customer Data Services Pack

5 Installing and Using Data Quality Health Check

Preface

This document describes how to install, manage, and customize the Oracle Enterprise Data Quality Customer Data Services Pack.

Audience

This document is intended for system administrators or application developers who are installing the Oracle Enterprise Data Quality Customer Data Services Pack. It is assumed that you have a basic understanding of application server and web technology and have a general understanding of Linux, UNIX, and Windows platforms.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Related Documents

For more information, see the following documents in the Oracle Enterprise Data Quality documentation set.

EDQ Documentation Library

The following publications are provided to help you install and use EDQ:

- *Oracle Enterprise Data Quality Customer Data Services Pack Siebel Integration Guide*
- *Oracle Fusion Middleware Release Notes for Enterprise Data Quality*
- *Oracle Fusion Middleware Installing and Configuring Enterprise Data Quality*
- *Oracle Fusion Middleware Administering Enterprise Data Quality*
- *Oracle Fusion Middleware Understanding Enterprise Data Quality*
- *Oracle Fusion Middleware Integrating Enterprise Data Quality With External Systems*

- *Oracle Fusion Middleware Securing Oracle Enterprise Data Quality*
- *Oracle Enterprise Data Quality Address Verification Server Installation and Upgrade Guide*
- *Oracle Enterprise Data Quality Address Verification Server Release Notes*

Find the latest version of these guides and all of the Oracle product documentation at

`http://docs.oracle.com`

Online Help

Online help is provided for all Oracle Enterprise Data Quality user applications. It is accessed in each application by pressing the **F1** key or by clicking the Help icons. The main nodes in the Director project browser have integrated links to help pages. To access them, either select a node and then press **F1**, or right-click on an object in the Project Browser and then select **Help**. The EDQ processors in the Director Tool Palette have integrated help topics, as well. To access them, right-click on a processor on the canvas and then select **Processor Help**, or left-click on a processor on the canvas or tool palette and then press **F1**.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1 Installing Customer Data Services Pack

This chapter explains how to install EDQ-CDS.

This chapter includes the following sections:

- Section 1.1, "Planning Your Installation"
- Section 1.2, "Installing EDQ-CDS"
- Section 1.3, "Configuring with Run Profiles"
- Section 1.4, "Initializing Custom Reference Data"
- Section 1.5, "Starting and Stopping Real-Time Jobs and Processes"

1.1 Planning Your Installation

This section describes the prerequisites, integration, compatibility, and necessary installation components.

1.1.1 Prerequisites

EDQ-CDS Release 11g (11.1.1.9) requires the following:

- EDQ release 11g (11.1.1.9) or later.
- If you are integrating EDQ-CDS with Siebel, you must install:
  - Siebel CRM or UCM version 8.1 or later.
  - Siebel Connector Release 11.1.1.7.3 or later

The requirements for production systems are as follows:

- 64-bit Operating System.
- 64-bit Java Virtual Machine (JVM).
- Minimum system memory of 8GB, with 4GB allocated to the JVM.
- Recommended system memory of 16GB, with 8GB allocated to the JVM.

> **Note :** It may be possible to run Test or Development instances on 32-bit systems with less memory.

1.1.2 Integrating with Siebel

When integrating a Siebel instance with EDQ to use CDS services, Oracle recommends that the necessary components be installed and configured in the following order:

1. Install the EDQ-CDS pack on the EDQ server as detailed in this chapter.
2. Install the EDQ Siebel Connector on the Siebel server .
3. Integrate Siebel with EDQ-CDS, see *Oracle Enterprise Data Quality Customer Data Services Pack Siebel Integration Guide*.

1.1.3 Compatibility Matrix

The matrix below shows the compatibility of all released versions of EDQ-CDS with other EDQ components:

EDQ-CDS	EDQ	EDQ Siebel Connector	EDQ-AV
9.0.1	9.0.3 or later	9.0.3-9.0.5	Any
9.0.2	9.0.4 or later	9.0.4-9.0.5	Any
9.0.3	9.0.5 or later	9.0.4-9.0.5	Any
9.0.4	9.0.7 or later	9.0.6	12.4.0.0.0 or later
9.0.5	9.0.7 or later	9.0.6	12.4.0.0.0 or later
11.1.1.7.3	11.1.1.7.3 or later	11.1.1.7.3	12.4.0.0.0 or later
11.1.1.9.0	11.1.9.0 or later	11.1.1.7.3 or later	14.2.0.0.0 or later

1.1.4 Components

EDQ-CDS is delivered as a distribution containing the following components:

- `edq-cds-11.1.1.9.N.(N).dxi` - the packaged EDQ project containing the EDQ-CDS data quality services.
- `edq-cds-initialize-reference-data-11.1.1.9.N.(N).dxi` - the packaged EDQ project containing the processes to prepare the EDQ-CDS Reference Data.
- `edq-cds-data-quality-health-check-11.1.1.9.N.(N).dxi` - the packaged EDQ project containing the processes for the Data Quality Health Check extension, see Chapter 5, "Installing and Using Data Quality Health Check."
- `config.zip` - containing EDQ extensions, configuration files and pre-initialized reference data needed to support EDQ-CDS.
- The `sql` directory contains Siebel specific scripts for configuring the staging database and a default Structured Query Language (SQL) script for use in creating staging tables for use with generic batch jobs.
- The `properties` directory contains the `dnd.properties` file, which is used when EDQ-CDS is integrated with a Siebel server. For more information, see *Oracle Fusion Middleware Integrating and Managing Siebel Environments with Enterprise Data Quality.*

1.2 Installing EDQ-CDS

To install EDQ-CDS on the EDQ server:

> **Note :** If your EDQ server uses a different landing area path from that set during installation (for example, `oedq.local.home/landingarea`), then the `landingarea` directory you create when the `config.zip` is extracted must be copied over the existing `landingarea` directory.

1. Extract the `config.zip` file over the `oedq.local.home` directory of the EDQ installation.

   > **Note:** Check that the contents of the zip file have been correctly installed in the local home directory - in particular, check that the `localgadgets`, `localwidgets` and `landingarea` subfolders all contain the CDS extensions before continuing.

2. Restart the EDQ Server.
3. Start the **EDQ Director client**, and log on as a user with the permission to create projects (Administrator or Project Owner)
4. To open the `edq-cds-initialize-reference-data-11.1.1.9.N.(N).dxi` packaged project, do one of the following:
   - Select **Open Package File...** on the **File** menu and browse to the `.dxi` file.
   - Right-click on an empty part of the Project Browser, select **Open Package File...**, and browse to the `.dxi` files.
   - Drag and drop the files onto the Project Browser.
5. Expand the `edq-cds-initialize-reference-data-11.1.1.9.N.(N).dxi` file and drag the whole **EDQ-CDS - Initialize Reference Data** project onto the **Projects** node.
6. Repeat steps 4 and 5 for the `edq-cds-11.1.1.9.N.(N).dxi` and the EDQ-CDS project it contains.
7. Repeat steps 4 and 5 for the `edq-cds-data-quality-health-check-11.1.1.9.N.(N).dxi` and the EDQ-CDS Data Quality Health Check project it contains.
8. Once the projects have been imported, right-click on the `.dxi` files, and select **Close Package File**.

1.3 Configuring with Run Profiles

There are several configuration options for EDQ-CDS that are controlled by the properties in the EDQ-CDS run profiles that are installed with the product and are used as follows:

File Name	Use	Property Sets
`edq-cds.properties`	Default EDQ-CDS Run Profile.	Language Domains High Frequency Name Maps Cluster Level (Real-time and Batch) Match Threshold (Real-time and Batch) Real-time Match Results Address Cleaning Properties Staging Data for Batch Jobs Staged Data Visibility
`edq-cds-siebel.properties`	The Siebel run profile is for Siebel integrations, and sets properties specific to the Siebel EDQ-CDS integration.	Language Domains High Frequency Name Maps Cluster Level (Real-time and Batch) Match Threshold (Real-time and Batch) Real-time Match Results Address Cleaning Properties Siebel Staging Data for Batch Jobs (Staging Data for Batch Jobs) Staged Data Visibility
`edq-cds-data-quality-health-check.properties`	Sets properties for Health Check functions.	EDQ Dashboard Source Input File Encoding Export Check Results Address Verification Country Code Individual Results Book Functionality (Results Book Settings) Entity Results Book Functionality (Results Book Settings) Staged Data Visibility
`edq-cds-daas.properties`	Not used at this time.	
`edq-cds-fusion.properties`	Not used at this time.	

These files are in the `oedq.local.home/runprofiles` directory of your EDQ installation directory. You can copy properties from one file to another so that the Run Profile you want to use contains all of the properties necessary to your configuration.

To edit a Run Profile:

1. Go to the `oedq.local.home/runprofiles` directory of the EDQ installation.
2. Open the Run Profile with a text editor.
3. Edit the values of the properties as required.
4. Save the file.

The properties in each Run Profile fall into several categories, as described in the following sections.

Note : It is also possible to configure Address Cleaning on a per country basis, although this is not done using the Run Profile, see Section 1.3.4, "Address Cleaning Properties."

1.3.1 Pre-Initialized Reference Data

The initialized Latin reference data and the `cdslists-initialized-full.zip` file (supplied in the `config.zip` file and located within the `oedq.local.home/landingarea/cdslists/` directory) together contain initialized reference data for all supported languages.

The Latin reference data is copied in when `config.zip` is extracted during the installation process. No further configuration steps are necessary to use it.

To use initialized reference data for all other supported languages, extract the `cdslists-initialized-full.zip` file over the `cdslists` directory, overwriting pre-existing data.

To use a different set of languages (for example, only Japanese) or to customize the reference data (for example, to add additional name standardizations), prepare and initialize it as required. This overwrites the pre-prepared files.

Note : If this pre-initialized Reference Data is used, it is *not* necessary to use Section 1.3.2, "Initialize Reference Data Properties."

1.3.2 Initialize Reference Data Properties

The section explains how to configure the properties of the Initialize Reference Data project using run profiles.

1.3.2.1 Language Domains

By default, name data for all non-Latin script languages is excluded when using the Run Profile. This is controlled by the following property:

```
phase.Initialize.process.*.Language\ Domains = LAT
```

Note:

- This value is set to LAT by default, which means all Latin data is included. To exclude Latin data, delete this value.
- Multiple language domains can be specified as a comma-separated list.

To include data in one or more script languages, add the associated property value, as documented in the comments of the Run Profile.

For example, to include Arabic script data, add the ARA value to the property:

```
phase.Initialize.process.*.Language\ Domains = LAT, ARA
```

If you edit this property, you must run the Initialize Reference Data job.

1.3.2.2 High Frequency Name Maps

By default, all names are included when records are processed. It is possible to exclude those non-Latin names that do not occur with a high frequency (for example, are not commonly used).

This is controlled by the following property:

```
phase.Initialize.process.*.High\ Frequency\ Only = N
```

To exclude uncommon non-Latin names, change this property value to Y.

If you edit this property, you must run the Initialize Reference Data job.

1.3.3 Matching Properties

These values are used to control clustering and matching behavior.

1.3.3.1 Cluster Level (Real-time and Batch)

By default, the cluster levels in the EDQ-CDS project for Real-Time and Batch processing of all record types is set to 2 (Typical), on a scale of 1 (Limited) to 3 (Exhaustive).

To set a different level for one or more types of processing, edit the values of the following properties accordingly:

```
######### Cluster Level ###########
# 1 = limited, 2 = typical, 3 = exhaustive
# Default = 2 if this property is absent

# Real-time & Batch Clustering
phase.Individual\ Cluster.process.*.Individual\ Cluster\ Level = 2
phase.Entity\ Cluster.process.*.Entity\ Cluster\ Level         = 2
phase.Address\ Cluster.process.*.Address\ Cluster\ Level       = 2

# Batch Matching
phase.Individual\ Match.process.*.Individual\ Cluster\ Level   = 2
phase.Entity\ Match.process.*.Entity\ Cluster\ Level           = 2
phase.Address\ Match.process.*.Address\ Cluster\ Level         = 2
```

Note : While the cluster levels set in the Run Profile override the default project settings, values passed from the web service take priority over both.

1.3.3.2 Cluster Comparison Limits

The match processors contain default cluster comparison limits that are applied. When set, the cluster comparison limit is a default upper limit on the maximum number of comparisons to be performed on a single cluster. You calculate this figure by assessing the number of comparisons that you want performed in a cluster before processing it. If the number of comparisons that would be performed on the cluster is greater than the limit, the cluster is skipped.

You can set the limits for a given cluster by adding the cluster limits properties to your edq-cds.properties file and editing the limit values. For example:

```
# Change the cluster limits to have a maximum of 15,000 comparisons per cluster
group, and use the comparison limit in preference over the group limit.
phase.*.process.Match\ -\ Individual.*.individual_match_cluster_comparison_limit =
15000
phase.*.process.Match\ -\ Individual.*.individual_match_cluster_group_limit = 0
phase.*.process.Match\ -\ Entity.*.entity_match_cluster_comparison_limit = 15000
phase.*.process.Match\ -\ Entity.*.entity_match_cluster_group_limit = 0
```

1.3.3.3 Match Threshold (Real-time and Batch)

By default, the match threshold in the project for Real-Time and Batch processing of all record types is set to `70` (on a percentage scale). Matches with a rule score below this value will not be returned.

To set a different level for one or more types of processing, edit the values of the following properties accordingly:

```
######### Match Threshold ###########
# Rule score below which matches will not be returned
# Default = 70 if this property is absent

# Real-time and Batch Matching
phase.Individual\ Match.process.*.Individual\ Match\ Threshold = 70
phase.Entity\ Match.process.*.Entity\ Match\ Threshold         = 70
phase.Address\ Match.process.*.Address\ Match\ Threshold       = 70
```

> **Note :** While the match thresholds set in the Run Profile override the default project settings, values passed from the Web Service take priority over both.

1.3.3.4 Real-time Match Results

Siebel 8.1 and later requires that real-time matching responses include both the driving record and all matching candidate records, with their match scores. For all other use cases it is not necessary to return the driving record in the response. The following option controls whether or not to include the driving record in responses to real-time matching services:

```
phase.*.process.*.Return\ Real-time\ Driving\ Record=
```

The default settings for this property are as follows:

- `edq-cds.properties` - `N`
- `edq-cds-siebel.properties` - `Y`

If this option is set to `Y` the driving record (with only the ID populated) is returned as the first record in the response, where there was at least one match in the candidate set. Otherwise, the driving record is excluded.

1.3.4 Address Cleaning Properties

When using the Address Cleaning service with EDQ-AV, the properties described in this section can be configured as required. For more information about Address Cleaning, see *Oracle Enterprise Data Quality Address Verification Installation Guide*.

1.3.4.1 Default Country Code

```
phase.*.process.Clean\ -\ Address.Default\ Country\ Code = US
```

This property can be used to define a system-level default country code in installations where addresses will typically all be in the same country and will not be specified per request on the interface.

The default value is US. Any codes that are entered here are expected to comply with the ISO-3166-1-alpha-2 specification.

1.3.4.2 Whether Address Verification Should Enable Geocoding

```
phase.*.process.Clean\ -\ Address.Enable\ Geocoding = Y
```

This property controls whether the Address Verification processor should use Geocoding, and correspondingly return latitude and longitude information with the cleaned address.

1.3.4.3 Default Allowed Address Verification Result Codes

```
phase.*.process.Clean\ -\ Address.Default\ Allowed\ Verification\ Result\
Codes = PV
```

This property specifies which Verification codes are permitted, which by default are P(partially verified) and V(verified).

1.3.4.4 Default Minimum Address Verification Level

```
phase.*.process.Clean\ -\ Address.Default\ Minimum\ Verification\ Level =
2
```

This property specifies the minimum required (post-process) Verification Match level, on a scale of 1 to 5. The default value is 2.

1.3.4.5 Default Minimum Address Verification Match Score

```
phase.*.process.Clean\ -\ Address.Default\ Minimum\ Verification\ Match\
Score = 95
```

This property specifies the minimum Match score required, on a scale of 1-100. The default setting is 95.

> **Note :** The three properties above set system-level defaults that control whether the Address Verification processor should actually clean an address based on the strength of the verification it is able to perform. These properties can also be overridden on a per-request basis by specifying them on the Address Cleaning interface, or overridden on a per-country basis (see Section 1.3.7, "Address Cleaning Per Country.")

1.3.4.6 Number of Lines Returned by the Address Clean Process

```
phase.*.process.Clean\ -\ Address.Number\ Of\ Address\ Lines =
```

Applications commonly support two, three or four address lines for the house number/street part of the address.

This property indicates the number of cleaned address lines that should be returned by the cleaning service.

The default settings in the Run Profiles are as follows:

- edq-cds.properties - 4
- edq-cds-siebel.properties - 2

1.3.4.7 Post-Processing

Post-processing is run after address cleaning, to apply certain changes to the results which have been returned from AV. This functionality is intended for Siebel integrations. Therefore, the default settings in the Run Profiles are:

- edq-cds.properties - N
- edq-cds-siebel.properties - Y

Standardize a Verified Country Name to Specific Values

If this value is set to Y country names are standardized to those in the default Siebel pick list:

```
phase.*.process.Clean\ -\ Address\ Post\ Process.Standardize\ Verified\
Country\ to\ CRM\ Values =
```

Standardize a Verified adminarea to Specific Values

If this value is set to Y, only adminarea values in the default Siebel pick list are returned:

```
phase.*.process.Clean\ -\ Address\ Post\ Process.Standardize\ Verified\
Admin\ Area\ to\ CRM\ Values =
```

Standardize Blank Verified Address Fields to be Returned as a Space

When the Siebel Data Quality interface receives back an empty string from a standardization service, it interprets this as meaning 'the current value should be retained'. In the case of Address Cleaning, it is sometimes desirable deliberately to remove the current value for an attribute; for example, an address standardization service may change an input address such that sub-building details are moved from the second line of the address to the end of the first line. In this case, in order not to duplicate the sub-building details in both address lines, a single space is returned in a return attribute to indicate to Siebel that the input value should be removed. Siebel does not in fact insert a space into the value; it interprets the space as meaning the value should be removed.

If this value is set to Y, any blank fields are populated with a single space character before being returned to Siebel:

```
phase.*.process.Clean\ -\ Address\ Post\ Process.Standardize\ Verified\
Blank\ Address\ Fields\ to\ Space =
```

1.3.5 Staging Data for Batch Jobs

By default, the Staging Data configuration for Batch jobs is derived from the candidate snapshots and the properties are set using the defined data source and the table names are set to the EDQ-CDS defaults. These properties can be edited as necessary if you want to point the (generic) batch matching jobs at different staging tables. The SERVERID and JOBID columns are used to enable processing of multiple batch jobs in parallel so they need to be edited in the run profile accordingly prior to each job submission; if they are not needed then default values can be used.

```
######### Staging Data Configuration Parameters For Batch Jobs ###########
# The JNDI data source name and table names may be different dependent on the
installation

# Where clause for candidate snapshots, to obtain data for specific server and job
phase.*.snapshot.*.where   = serverid = 'SERVERID' AND jobid = 'JOBID'
```

```
# Export parameters for specific server and job
phase.*.process.*.serverid = SERVERID
phase.*.process.*.jobid    = JOBID

# JNDI data source name for staging schema in database
phase.*.snapshot.*.remotejndi = jdbc/edqcdsstaging
phase.*.export.*.remotejndi   = jdbc/edqcdsstaging

# Table names for candidate staging tables (snapshots)
phase.*.snapshot.Entity\ Candidates.table_name     = EDQCDS_CANDIDATES_ENT
phase.*.snapshot.Individual\ Candidates.table_name = EDQCDS_CANDIDATES_IND
phase.*.snapshot.Address\ Candidates.table_name    = EDQCDS_CANDIDATES_ADD

# Table names for result staging tables (exports)
phase.*.export.Batch\ Matches.table_name           = EDQCDS_MATCHES
phase.*.export.Batch\ Cluster\ Results.table_name  = EDQCDS_CLUSTER_KEYS
```

1.3.6 Staged Data Visibility

By default, most Staged Data sets are suppressed in the Results view of the Server Console. Only those Staged Data sets listed in this section of the Run Profile are visible in Server Console by default:

```
# Initialize Project
stageddata.\[QA\]\ Single\ chars.visible = yes
stageddata.\[QA\]\ Variant\ has\ Multiple\ Masters.visible = yes
stageddata.\[QA\]\ Variant\ is\ Master.visible = yes
stageddata.Conflict\ Res\ \-\ Removed\ Links\ ALL.visible = yes
```

To make other Staged Data sets visible, add a property in the format of those included in the Run Profile, as in the preceding example.

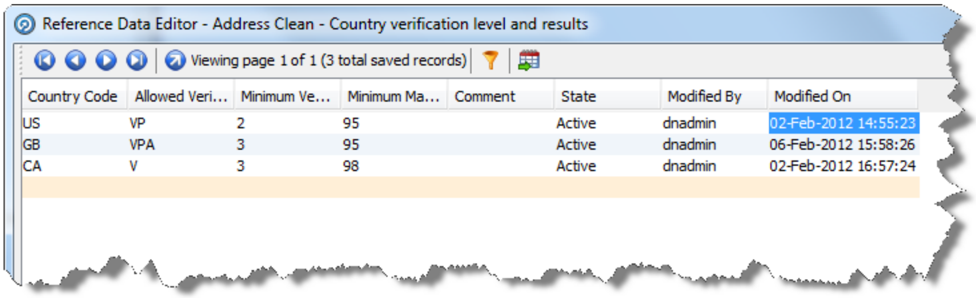
1.3.7 Address Cleaning Per Country

The extent to which EDQ-AV can verify addresses varies depending on the country. Additionally, address data from certain countries may be trusted more than data provided for others.

To allow for this, it is possible to set different parameters for address cleaning on a per-country basis.

To set the required parameters:

1. Open the Director client.
2. In the Project Browser, select **EDQ-CDS > Reference Data**.
3. Open the **Address Clean - Country verification level and results** Reference Data.



Reference Data Editor - Address Clean - Country verification level and results

Viewing page 1 of 1 (3 total saved records)

Country Code	Allowed Veri...	Minimum Ve...	Minimum Ma...	Comment	State	Modified By	Modified On
US	VP	2	95		Active	dnadmin	02-Feb-2012 14:55:23
GB	VPA	3	95		Active	dnadmin	06-Feb-2012 15:58:26
CA	V	3	98		Active	dnadmin	02-Feb-2012 16:57:24

4. In the Reference Data Editor, change the default settings for US, GB and CA, and add additional rows and settings for other countries as required.
5. Click **OK** to save changes, or **Cancel** to abandon.

> **Note :** For further details of the Verification settings, see Chapter 2, "Using Business Services."

1.4 Initializing Custom Reference Data

If the pre-initialized Reference Data shipped with EDQ-CDS is used, this procedure is not required. However, if any of the initialization options detailed in Section 1.3.2, "Initialize Reference Data Properties" have been changed from their default settings the Reference Data must be re-initialized by running the job in the Server Console.

To do this, use the following procedure:

1. Open the Server Console.
2. Expand the **EDQ-CDS - Initialize Reference Data** project.
3. Right-click the **MAIN Initialize Reference Data** job and select **Run...**
4. Select the EDQ-CDS run profile and specify a Run Label of **cds**.

> **Note:**
>
> - This job must be re-run if the Reference Data is customized, or if the Run Profile is modified in order to select different languages to initialize.
> - Oracle recommends that `cds` is used as the Run Label for all CDS jobs.

1.5 Starting and Stopping Real-Time Jobs and Processes

There are several jobs that *must* be running in order to use the Real-Time processes. These jobs are controlled by two other jobs: **Real-Time START ALL** and **Real-Time STOP ALL**, which *must* be started in the Server Console.

To start the Real-Time processes:

1. Open the Server Console.
2. Expand the **EDQ-CDS** project.
3. Run the **Real-Time START ALL** job.
4. Select the required Run Profile from the drop-down field.

> **Note :** If running the job in order to provide services to Siebel (either CRM or UCM), the edq-cds-siebel Run Profile must be selected, so that the correct configuration settings for Siebel are used.
>
> If running the job to provide services to other applications, the edq-cds Run Profile is recommended. For more information, see Section 1.3, "Configuring with Run Profiles."

5. Enter **cds** as the Run Label.
6. Click **OK**.

Under certain circumstances it may be necessary to stop and restart the Real-Time processes. For example, if new Reference Data has become available, it will be necessary to stop the Real-Time processes, re-run the **Initialize Reference Data** job, and start the Real-Time processes again.

To stop the Real-Time processes:

1. Open the Server Console.
2. Expand the **EDQ-CDS** project.
3. Run the **Real-Time STOP ALL** job.

1.5.1 Scheduling a Real-Time START ALL Job at Start Up

If the server restarts, it will be necessary to also restart the Real-Time jobs with the appropriate Run Profile and Run Label. To ensure this happens automatically, use the following procedure to configure the **Real-Time START ALL** job to run at start up:

1. Open the Server Console
2. Expand the EDQ-CDS project.
3. Open the **Real-Time START ALL** job.
4. Right click and select the **Schedule** option.
5. Select the **Startup** radio option.
6. Select the required Run Profile from the drop-down field.

> **Note :** If running the job in order to provide services to Siebel (either CRM or UCM), the **edq-cds-siebel** Run Profile must be selected, so that the correct configuration settings for Siebel are used.
>
> If running the job to provide services to other applications, the `edq-cds` Run Profile is recommended.

7. Specify a **Run Label** of **cds**.
8. Click **OK** to save the changes.

2 Using Business Services

This chapter describes how you can use the EDQ-CDS Business Services functionality.

This chapter includes the following sections:

The provided business services are ready for integration with Siebel Customer Relationship Management (CRM) or Universal Customer Master (UCM) and may also be called by other applications if they are configured to do so.

Ready-to-use, EDQ-CDS provides pre-configured data quality services which can be modified or enhanced by editing the underlying EDQ processes that implement their functionality. These three types of service can be used with Individual, Entity, and Address records:

- Cleaning
- Clustering (for use in Matching integrations)
- Matching

2.1 Cleaning Services

There are three cleaning services provided in EDQ-CDS: Address Clean, Individual Clean and Entity Clean. The Individual and Entity Clean services are provided as placeholders, which are pre-integrated with Siebel, and easily integrated with other applications, but which need to be modified towards specific requirements.

2.1.1 Address Clean

The EDQ-CDS Address Clean web service provides the following functionality:

- Verification of an input address (returning a verification code and description)
- Geocoding of an address (returning latitude and longitude co-ordinates, with additional metadata)
- Correction, standardization and completion of input addresses (provided the address was verified to a sufficient, configurable, level)
- Search (returning a list of possible addresses for partial input data)

Note: Siebel's Data Quality interface can only accept a single return address for each input address, which means that it cannot use Search mode (which returns many).

2.1.1.1 Address Search

The CDS Address Clean service now supports Search mode. If the input parameter mode is set to `S`, the AddressClean service will search for address matches for each input address, and may return multiple results.

Search mode means that the service will attempt to find the closest real address (in the installed Loqate data) for a partial input address. Search mode only supports searching for whole addresses. It is not suitable to return a subset of attributes based on partial input – for example it does not support the return of a list of postal codes (only) for a partial postal code input.

Search mode calls Loqate’s Address API (more information at www.loqate.com/oracle) in Search mode, meaning multiple addresses may be returned for each input address.

If you have purchased the Powersearch data from Loqate, the Powersearch method and data for fast address completion based on the beginning of a valid address will be used by the service if a) the service is run in search mode (mode `s`) and b) input data is only presented in the Address1 and Country fields. If information is input into other fields, such as Address2, Locality, AdminArea or PostalCode, the normal Loqate Search method is used. This method is more similar to Verify mode but allows multiple possible suggestions for a valid address to be returned from the service.

Search mode is most effective for the following use cases:

- Auto-completion of addresses where the user types the beginning of an address and multiple possibilities are returned. Note that calling applications may choose to call the service automatically after `N` key presses, allowing the results to be refined as more input data is provided. This requires the Loqate Powersearch data to be purchased and installed.
- Lookup searches in certain countries; for example in the UK, a Postcode Lookup can be performed by inputting the country and a complete postal code. The service will return all addresses at this postal code. Note that this is not suitable for all countries, where in order to return a reasonable number of potential address matches that can be displayed on a UI, more input information is needed.

The threshold parameters that control address correction are not used in Search mode, that is, search results below the thresholds are not suppressed.

The strongest `N` results will be returned from the Search service, where `N` is a parameter of the service that can be set in the run profile as follows. The default is 20:

```
# Maximum number of results to return in Search mode

phase.*.process.Clean\ -\ Address.Maximum\ Search\ Results = 20
```

For more information about the underlying Search mode used by this service, register at www.loqate.com and see the following page on Loqate’s support site: http://www.loqate.com/support/available-processes/search-process/

The search results will be returned in order of strength, according to the returned `AccuracyCode` for each search result. This order uses the Result (Verified, Partial, Ambiguous, Conflict, Reverted or Unverified), the Verification Level (Delivery Point, Premise, Thoroughfare, Locality, Admin Area) and Match Score (up to 100).

2.1.1.2 Address Verify

In Verify mode, the service is `N:N`; that is, single and multiple record input and output is possible, but only one record is returned for each record submitted. Each input address is verified and may be corrected, enhanced and geocoded, depending on the options that the job is run with, and the input parameters.

By default, Verify mode will correct input addresses to their best match in the Loqate reference data. If you want only to verify addresses and return a verification accuracy code, but not change the input addresses, set the `minimumverificationlevel` parameter to 6 so that correction will never occur even if the addresses were verified to delivery point level (level 5).

2.1.1.3 Using Address Clean

The Address Clean web service is normally used for real-time verification and cleansing of addresses as they are entered and updated in an application, such as Siebel.

In the case of Siebel, the web service is also used in batch. When a batch address cleansing job is run, the web service will be used on all of the in-scope records in the batch job.

For other applications, it is recommended to add configuration to EDQ-CDS to map data from and to the Address Clean data interface in order to run the service in batch mode.

2.1.1.4 Using Address Clean with Siebel

This section describes functionality for Siebel integration, and describes related reference data and post-processing options that are run after address cleaning, to apply certain changes to the results which have been returned from AV.

Standardizing Country Names

If a Siebel picklist field is mapped to the country attribute then the list of standardized country name values needs to correspond to this list.

See the reference data "Address Clean - Country Code to Standard CRM Country Name Map" provided in EDQ.

For more information on this post-processing option, see the Standardize a Verified Country Name to Specific Values post-processing option in the Post-Processing section in the Installing Customer Data Services Pack chapter.

Standardizing Admin Areas

This is a list of US state codes. If the service is being used on non-US addresses then this post-processing needs to be disabled or the list of admin areas needs to be extended to cover the required localities. For example, for Canadian addresses, if the province field is mapped to the admin area attribute then the standardization list should contain values corresponding to the possible picklist values.

See the reference data "Address Clean - Admin Area to Standard CRM Admin Area Map" provided in EDQ.

For more information on this post-processing option, see the Standardize a Verified adminarea to Specific Values post-processing option in the Post-Processing section in the Installing Customer Data Services Pack chapter.

Blanking Siebel Address Fields

When the Siebel Data Quality interface receives back an empty string from a standardization service, it interprets this as meaning 'the current value should be retained'. In the case of Address Cleaning, it is sometimes desirable deliberately to remove the current value for an attribute; for example, an address standardization service may change an input address such that sub-building details are moved from the second line of the address to the end of the first line. In this case, in order not to duplicate the sub-building details in both address lines, a single space is returned in a return attribute to indicate to Siebel that the input value should be removed. Siebel does not in fact insert a space into the value; it interprets the space as meaning the value should be removed.

For more information on this post-processing option, see the Standardize Blank Verified Address Fields to be Returned as a Space post-processing option in the Post-Processing section in the Installing Customer Data Services Pack chapter.

2.1.1.5 Interface

The following table provides a guide to the interface attributes of the Address Clean web service.

Attribute Name	Data Type	Use	Notes
`addressid`	String	In/Out	Unique identifier for the address.
`address1`	String	In/Out	Address line 1
`address2`	String	In/Out	Address line 2
`address3`	String	In/Out	Address line 3
`address4`	String	In/Out	Address line 4
`dependentlocality`	String	In/Out	A smaller population center data element, dependent on the contents of the `city` field. For example, a Neighborhood in Turkey. For many countries, this attribute is not used.
`doubledependentlocality`	String	In/Out	The smallest population center data element, dependent on both the contents of the `city` and `dependentlocality` fields. For example, a village in the UK. For many countries, this attribute is not used.
`city`	String	In/Out	The locality, town or city of the address.
`subadminarea`	String	In/Out	The smallest geographic data element within a country. For example, a county in the USA.
`adminarea`	String	In/Out	The most common geographic data element within a country. For example, a State in the USA or a Province in Canada.
`postalcode`	String	In/Out	Postal or zip code for the address, if relevant for the country.
`postalcodeprimary`	String	In/Out	The first part of a 2-part postal code, for example the ZIP code of a US address.

Attribute Name	Data Type	Use	Notes
`postalcodesecondary`	String	In/Out	The second part of a 2-part postal code, for example the +4 part of a US address ZIP+4 code.
`country`	String	In/Out	On input, an ISO two-character country code (preferred) or country name. On output, the full country name (even if the input is the country ISO code). **Note:** If the country field is blank, the service then attempts in sequence to derive the country from: ■ the default country code input, ■ the country code from the run profile, ■ to derive the country from the city.
`defaultcountrycode`	String	Input only	Default ISO two-character country code to use if country not populated. This overrides the default value used when running the job.
`case` (parameter)	String	Input only	Transforms output according to the setting: `U` (Upper)- Transforms all text to Upper case. `L` (Lower)- Transforms all text to Lower case. `M` (Mixed) - Transform all text to Mixed case, except `postalcode` and `adminarea`. These field values are left as returned from the AV processor if the address was verified. If the address was not verified, the `postalcode` is left as entered, and the `adminarea` is converted to Mixed case. `O` (Original) - Text is not transformed.
`mode` (parameter)	String	Input only	Mode in which the request is to be run. `v` (Verify)- Operates the service in verify mode and uses the thresholds to determine whether or not to 'change' the address depending on the confidence of the best match. `s` (Search)- Operates the service in search mode. Note that the thresholds described in this table do not apply, and the service simply returns the search results. **Note**: only Verify mode is supported for Siebel integrations.

Attribute Name	Data Type	Use	Notes
`minimumverificationmatchscore` (parameter)	Number	Input only	A numeric value between 0 and 100, representing the minimum score which a match must achieve to be used as a cleaned address. Input addresses will be left unchanged if the Match Score of the Address Verification processor for the address is lower than the input value. This parameter is ignored in Search mode.
`minimumverificationlevel` (parameter)	Number	Input only	A numeric value between 1 and 5, representing the minimum verification level which a match must achieve to be used as a cleaned address. Input addresses will be left unchanged if the Verification Level of the Address Verification processor for the address is lower than the input value. For a description of what each level means, see Section , "Notes on Verification Levels". This parameter is ignored in Search mode.
`allowedverificationresultcodes` (parameter)	String	Input only	A list of any of the following single-letter result codes with no separator (for example, 'VPA'): `V` (Verified), `P` (Partially Verified) `A` (Ambiguous) `R` (Reverted) `U` (Unverified) Input addresses will be left unchanged if the Verification Result Code of the best match is not in this list. For example, the first character of the `verificationcode`) for the address is not one of the listed input values. Applies only in Verify mode.
`fulladdress`	String	Output only	Full verified address returned from address verification. The address lines are pipe-separated.
`countrycode`	String	Output only	ISO 2 char country code of verified address country.
`verificationcode`	String	Output only	Verification code for the address.

Attribute Name	Data Type	Use	Notes
verified	String	Output only	This indicates whether or not the input address was changed [Cleaned] to the address returned from address verification. If the address does not verify to a sufficient level (according to the minimumverificationmatchscore, minimumverificationlevel and allowedverificationresultcodes parameters) then the input address is returned. Possible returned values are Y (cleaned), N (not cleaned) or X (EDQ Address Verification is not installed). Note that the verificationcode of the best match is always returned, even if the result, level or match score were under the required thresholds to consider the address as 'verified' and update the input address. Therefore it is possible for the verificationcode to indicate a 'V...' result but for 'verified' to be N.
verificationcodedescription	String	Output only	US English description of the verification code.
latitude	Number	Output only	WGS 84 latitude in decimal degrees format.
longitude	Number	Output only	WGS 84 longitude in decimal degrees format.
geoaccuracycode	String	Output only	A code indicating the level of accuracy of the returned geocodes (latitude and longitude) co-ordinates.
geoaccuracycodedescription	String	Output only	US English description of the geoaccuracycode.
geodistance	Number	Output only	Radius of accuracy (in meters) for the returned geocodes. The higher the value, the less accurate the geocoding result.

2.1.1.6 Parameters

The Address Clean web service uses a number of input parameters to control its behavior when processing addresses, as listed and described in the table above.

The minimumverificationmatchscore, minimumverificationlevel and allowedverificationresultcodes parameters are all used as message-level thresholds to override whether or not to change (clean) an input address based on the confidence level that the EDQ Address Verification processor reaches when processing it. Normally, and when using the Address Clean service with Siebel, these parameters are not used, and the underlying settings in the Address Clean process are used to drive whether or not to change the address. In this process it is possible to set these same parameters on a per-country basis if required. Where country-specific thresholds are not provided, global default settings are applied, and these may be set using the EDQ-CDS Run Profile. The priority in which the thresholds are applied is therefore:

1. Per-message threshold settings using the parameter attributes as above
2. Per-country threshold values expressed in the Reference Data set `Address Clean - Country verification level and results`
3. The global default settings expressed in the process and overridden on a per-run basis by the use of a run profile.

An additional configuration option is available to control the number of address lines that are returned from the service. This is not exposed as a parameter on the interface, but can be set using the `phase.*.process.Clean\ -\ Address.Number\ Of\ Address\ Lines` Run Profile setting. The default number of lines to return is `4`.

Notes on Verification Levels

The following verification levels are possible. The maximum verification level that it is possible to reach varies by country. For information on the maximum level in each country, see the Loqate Oracle EDQ Portal website at

`http://www.loqate.com/oracle`

The verification level is output as the second character of the Accuracy Code returned by the EDQ Address Verification processor. The 'post-processed' verification level is used (not the 'pre-processed' level); that is, the verification level achieved after EDQ Address Verification applies standardization and parsing to the input address.

Verification Level	Description
1	Verified to Administrative Area (State, Region or County) level
2	Verified to Locality (City or Town) level
3	Verified to Thoroughfare (Street) level
4	Verified to Premise (Building Number) level
5	Verified to Delivery Point (Sub-Building Number) level

Note: If EDQ Address Verification is not installed (or not installed correctly), the Address Clean service can still be installed, and the job that implements it can still be run. However, if a request is made to the service, all the output fields will be blank, except for the `verified` output field, which will have the value *`X`*, and the `verificationcode` output, which will have the value `-1.0`.

2.1.2 Individual Clean

The Individual Clean web service is designed to verify, correct, standardize or enhance records representing individuals, whether these be customers, prospective customers, contacts, or employees.

The **Clean - Individual** process that implements this service in EDQ-CDS is just a placeholder, and must be customized to requirements. A default process that converts the input name attributes to upper case is provided so that when connecting this service to Siebel or other applications, it is simple to test that the service is correctly connected.

The service is `N:N`; that is, single and multiple record input and output is possible, but only one record is returned for each record submitted.

The Siebel Data Quality interface always calls the service with a single record per request, whether running in real-time or batch.

2.1.2.1 Using Individual Clean

The Individual Clean web service may be extended for many purposes, including (but not limited to):

- Verification of input details related to individuals (for example, an email address)
- Standardization of input details related to individuals (for example, a job title)
- Enhancement of data related to individuals (for example, by matching reference data for individuals and returning additional attributes, such as social media handles)

Normally, the web service will be called in real-time, when individual records are added or updated in an application.

In the case of Siebel, the web service is also used in batch. When a batch contact cleansing job is run, the web service will be used on all of the in-scope records in the batch job.

For other applications, it is recommended to add configuration to EDQ-CDS to map data from and to the **Individual Clean** data interface in order to run the service in batch mode.

The interface used by the service is designed to map directly to the Contact business component in Siebel, but can be freely extended with new attributes on both input and output. Siebel's DQ vendor parameters may be extended to pass through different attributes to the service.

2.1.2.2 Interface

The following table provides a guide to the default Individual Clean web service interface. All attributes are both input and output by default. It is possible to input an empty value to the interface but to populate the attribute on output, so providing a data enhancement service.

Attribute Name	Data Type	Notes
`individualid`	String	A unique identifier for the individual record.
`languages`	String	3 character Siebel language code. This can be used to determine whether a name containing Kanji should be treated as Japanese or Chinese.
`nameid`	String	Unique identifier for the name.
`title`	String	Title
`firstname`	String	First name
`middlename`	String	Middle name
`lastname`	String	Last Name
`gender`	String	`M` or `F`
`dob`	String	Date of Birth
`jobtitle`	String	Job Title
`homephone`	String	Home Phone Number
`workphone`	String	Work Phone Number

Attribute Name	Data Type	Notes
`mobilephone`	String	Mobile Phone Number
`faxphone`	String	Fax Number
`alternatephone`	String	Alternate Phone Number
`email`	String	Email Address
`taxnumber`	String	Tax Number
`nationalidnumber`	String	Social Security Number (US) or equivalent.

2.1.3 Entity Clean

The Entity Clean web service is designed to verify, correct, standardize or enhance records representing entities, whether these be company customers, prospective company customers, suppliers, or other organizations.

The **Clean - Entity** process that implements this service in EDQ-CDS is just a placeholder, and must be customized to requirements. A default process that converts the input name and subname attributes to upper case is provided so that when connecting this service to Siebel or other applications, it is simple to test that the service is correctly connected.

The service is `N:N`; that is, single and multiple record input and output is possible, but only one record is returned for each record submitted.

The Siebel Data Quality interface always calls the service with a single record per request, whether running in real-time or batch.

2.1.3.1 Using Entity Clean

The Entity Clean web service may be extended for many purposes, including (but not limited to):

- Verification of input details related to entities (for example to check that a website is syntactically valid)
- Standardization of input details related to entities (for example company names and locations)
- Enhancement of data related to entities (for example by matching reference data for entities and returning additional attributes, such as DUNS numbers from Dun and Bradstreet)

Normally, the web service will be called in real-time, when entity records are added or updated in an application.

In the case of Siebel, the web service is also used in batch. When a batch account cleansing job is run, the web service will be used on all of the in-scope records in the batch job.

For other applications, it is recommended to add configuration to EDQ-CDS to map data from and to the **Entity Clean** data interface in order to run the service in batch mode.

The interface used by the service is designed to map directly to the Account business component in Siebel, but can be freely extended with new attributes on both input and output. Siebel's Data Quality vendor parameters may be extended to pass through different attributes to the service.

2.1.3.2 Interface

The following table provides a guide to the default Entity Clean web service interface. All attributes are both input and output by default. It is possible to input an empty value to the interface but to populate the attribute on output, so providing a data enhancement service.

Attribute Name	Data Type	Notes
entityid	String	A unique identifier for the entity record.
languages	String	3 character Siebel language code. This can be used to determine whether a name containing Kanji should be treated as Japanese or Chinese.
nameid	String	Unique identifier for the name.
name	String	Organization name for example, "Oracle Corporation UK".
subname	String	Department or site for example, "Reading", "Accounts Payable", etc.
phone	String	Phone Number
alternatephone	String	Alternate Phone Number
website	String	Website Address
taxnumber	String	Company Tax Number
vatnumber	String	Company VAT Number

2.2 Clustering Services

EDQ-CDS clustering services are designed to generate a number of key values for an individual, entity or address record. The returned key values for a record are then used by applications such as Siebel to select 'candidate' records for matching, where any existing record that shares any key value with the 'driving' record (the record submitted to the clustering service) should be considered a candidate. The driving and candidate records are then submitted in a single request to the relevant Matching service.

In addition to being called in real-time in order to prevent the insertion of duplicate records into an application, the clustering services are used in batch mode to populate the key values on all existing records in the application, so that real-time and incremental batch matching jobs, both of which use the key values for existing records for candidate selection, work correctly.

The clustering services are `N:N`; meaning that single and multiple record input and output is possible. In real-time, a single output record is returned containing an array of keys. In batch mode, each key is returned as a separate record in the staging table.

2.2.1 Using Clustering Services

The real-time clustering services are normally used as the first call to EDQ-CDS when a new or updated record needs to be checked for matches against records in an application. The returned key values are used to select candidate records to be submitted with the driving record to the matching service.

In order to ensure that keys are always up-to-date, the clustering services should be called whenever a record is updated. This includes the scenario when a master record in Siebel UCM, or other hub, is updated due to a confirmed match with an incoming driver record.

The clustering services are exposed using the following:

Web Services:

- `IndividualCluster`
- `EntityCluster`
- `AddressCluster`

Batch Jobs:

- `Batch Individual Cluster`
- `Batch Entity Cluster`
- `Batch Address Cluster`

2.2.2 Interface

The clustering web services present input interfaces with direct mappings to the shared 'Candidate' data interfaces as follows:

Web Service	Input Interface
`IndividualCluster`	See Section 2.4.1.2, "Individual Candidates."
`EntityCluster`	See Section 2.4.1.3, "Entity Candidates."
`AddressCluster`	See Section 2.4.1.4, "Address Candidates."

All the clustering web services return output attributes using the common Real-time Cluster Results Interface data interface, see Section 2.4.3, "Cluster Results Interfaces."

2.2.3 Parameters

The `IndividualCluster`, `EntityCluster` and `AddressCluster` web services all expose a `clusterlevel` parameter attribute, which is used to drive the sensitivity of the clustering service:

Parameter Attributes	Data Type	Accepted Values	Use
`clusterlevel`	String	A numeric value (1, 2 or 3)	1 = Limited, 2 = Typical, 3 = Exhaustive

The `clusterlevel` setting determines which methods of cluster key generation are used to generate keys for each driving record. If used, the per-message setting overrides the default setting in the process (that can be adjusted when running a job using the EDQ-CDS Run Profile).

The settings operate as follows:

1 (Limited)
Only cluster keys that do not normally return large numbers of candidate records are generated by the service. This is recommended if working with very large data sets with tight matching requirements.

2 (Typical)
Uses the default methods for generating cluster keys.

3 (Exhaustive)
Methods that may return a large number of candidate records are used to generate cluster keys. This includes methods that only use the name fields. This setting is recommended only when working with low volume data sets (for example, less than a million individuals, or less than 100,000 entities) with loose matching requirements.

2.3 Matching Services

The matching services - sometimes referred to as the Match Scoring services in Siebel Universal Data Quality documentation - compare input (driver and candidate) records and produce a list of possible matches, with scores to indicate how good the matches are, and additional information about how the records matched.

In the matching services, the record for comparison is called a driver, and the records it is compared with are known as candidates. Driver records are also compared with each other, but candidate records are never compared with other candidates. Only the highest scoring match for any given record pair is returned.

> **Note:** Siebel currently does not use the Address Matching service, in either batch or real-time, though this service may be integrated with other applications.

2.3.1 Using Matching Services

There are three forms of matching supported:

Real Time
A single driver record (possibly with multiple child entity records) is compared against many candidates.

Full Batch
All records are compared against one another (subject to clustering); for example, all are specified as drivers. This is an extensive operation that can take some time. It is normally used on a new installation, or perhaps as part of a regular maintenance operation.

Incremental Batch
A specific subset of record types are identified and specified as the driver records. Next, other records with matching cluster keys are identified, and specified as the candidates. The driver and candidate records are compared, and the driver records are compared with each other. An example of how this might be used is a regular check on all new records during a set time period, such as a week or month, against pre-existing records.

The real-time matching services are exposed via the following web services in EDQ-CDS:

- `IndividualMatch`
- `EntityMatch`
- `AddressMatch`

The batch and real-time processes that implement the matching services use the following Data Interfaces for input data (mapped to the above web services respectively for real-time matching):

- Individual Candidates
- Entity Candidates
- Address Candidates

The **Matches** data interface is used as a common output interface for all record types (Individual, Entity and Address), although the fields mapped for each record type vary.

2.3.2 Matching Using Multiple Identifier Values

Matching services within EDQ-CDS are designed to enable users to submit any number of alternative identifier values to use when matching a given individual or entity; for example, multiple email addresses, addresses or names.

EDQ-CDS can perform matching on multiple values of the following attributes if submitted as pipe-delimited lists:

- uid and eid attributes
- alternatephone
- email
- website
- taxnumber
- nationalidnumber
- vatnumber

However, in order for EDQ-CDS to match Individual or Entity records with multiple names or addresses, such records must first be split into multiple records. Each of these records must have the same `entityid` or `individualid`, but with different `nameid` and/or `addressid` attributes.

So, an Individual record with three names must be split into three records, as follows:

individualid	nameid	firstname	lastname	enail	address1
A1	1	John	Smith	jsmith@jsmith.com	56 High Street
A1	2	Jon	Smith	jsmith@jsmith.com	56 High Street
A1	3	J	Smith	jsmith@jsmith.com	56 High Street

An Individual record with three account names must be split into three records:

individualid	accountname	accountnameid	firstname	lastname	enail	address1
A1	entity1	1	John	Smith	jsmith@jsmith.com	56 High Street
A1	entity2	2	John	Smith	jsmith@jsmith.com	56 High Street
A1	entity3	3	John	Smith	jsmith@jsmith.com	56 High Street

Similarly, an Entity record with two names must be split into two records:

entityid	nameid	name	subname	website	address1
B1	1	OracleLtd	Accounts Payable	www.oracle.com	Oracle Parkway
B1	2	Oracle Corporation UK	Accounts Payable	www.oracle.com	Oracle Parkway

An Entity record with two names and two addresses must be split into four records:

entityid	nameid	name	subname	website	addressid	address1
C1	1	OracleLtd	Accounts Payable	www.oracle.com	A	Oracle Parkway
C1	1	OracleLtd	Accounts Payable	www.oracle.com	B	Thames Valley Park
C1	2	Oracle Corporation UK	Accounts Payable	www.oracle.com	A	Oracle Parkway
C1	2	Oracle Corporation UK	Accounts Payable	www.oracle.com	B	Thames Valley Park

The EDQ Siebel Connector automatically prepares the data to use the matching service appropriately, where EDQ-CDS is integrated with Siebel. If the use of multiple child entities has been configured in Siebel, then the data is prepared in the structure required by the EDQ-CDS matching services. For example, concatenating multiple phone numbers into a pipe-delimited list, and splitting out multiple records if the use of multiple names or addresses is configured.

> **Note:** For records with multiple child entities, only one match will ever be returned between a pair of records. This will always be the highest scoring match according to the match rules.

2.3.3 Interfaces

The matching web services present input interfaces with direct mappings to the shared 'Candidate' data interfaces as follows:

Web Service	Input Interface
IndividualMatch	See Section 2.4.1.2, "Individual Candidates."
EntityMatch	See Section 2.4.1.3, "Entity Candidates."
AddressMatch	See Section 2.4.1.4, "Address Candidates."

All the matching services return output attributes using the common **Matches Interface** data interface.

2.3.4 Parameters

The IndividualMatch, EntityMatch and AddressMatch web services all expose a matchthreshold attribute, which is used to suppress (not return) matches with a score below this threshold.

Parameter Attribute	Data Type	Accepted Values	Use
`matchthreshold`	Number	A numeric value between 0 and 100	Matches that are generated by the matching service with a score below the stated threshold will be suppressed (not returned in the web service response). If used, this overrides the default setting in the process, which can be adjusted when running a job using the EDQ-CDS Run Profile.
`matchoptions`	String	N/A	This parameter is not currently used. It is intended for use in future versions of EDQ-CDS.

2.4 Data Interfaces

This section describes the following EDQ-CDS data interfaces:

- Section 2.4.1, "Candidate Interfaces"
- Section 2.4.2, "Matches Interface"
- Section 2.4.3, "Cluster Results Interfaces"

2.4.1 Candidate Interfaces

The Candidate interfaces are used for data input to individual/entity/address matching and clustering in both batch and real-time.

> **Note:** For the interface fields which do not accept multiple values (identified in the tables), avoid using the pipe character, or double-pipe character.

2.4.1.1 Specifying ID fields for Multi-value Fields to Identify Value Matched

Some fields in the EDQ-CDS matching service, such as e-mail and phone number, accept a pipe-delimited string with multiple values to be matched up. The interface provides ID fields for each of these multi-value fields, that you can use to identify which of the values matched, and return these from the match service.

2.4.1.2 Individual Candidates

Attribute Name	Data Type	Supports Multiple Values? (Y/N)	Notes
`candidate`	String	N	0 = driving record, 1 = candidate. Used in matching only. All driving records are compared against each other and against each candidate, but candidates are not compared against each other.
`individualid`	String	N	Unique identifier of the individual (for example, customer, employee, or contact). Mandatory, for identifying which records matched in the return interface.

Attribute Name	Data Type	Supports Multiple Values? (Y/N)	Notes
`languages`	String	Y	3 character Siebel language code. Only used in name standardization to help determine whether a name containing Kanji is Japanese or Chinese.
`uid1`	String	Y	Unique ID 1 (single or pipe-delimited list of multiple values). **Note**: The Unique ID fields are used to match records based on custom unique identifiers, such as passport or tax numbers. For more information, see *Oracle Enterprise Data Quality Customer Data Services Pack Guide*.
`uid2`	String	Y	Unique ID 2 (single or pipe-delimited list of multiple values).
`uid3`	String	Y	Unique ID 3 (single or pipe-delimited list of multiple values).
`uid1id`	String	Y	A single or pipe-delimited list of multiple values corresponding to the ID values in `uid1`, which you can use to identify, in the case of a match, which of the values matched, and return this from the match service.
`uid2id`	String	Y	A single or pipe-delimited list of multiple values corresponding to the ID values in `uid2`, which you can use to identify, in the case of a match, which of the values matched, and return this from the match service.
`uid3id`	String	Y	A single or pipe-delimited list of multiple values corresponding to the ID values in `uid3`, which you can use to identify, in the case of a match, which of the values matched, and return this from the match service.
`eid1`	String	Y	Elimination ID 1 (single or pipe-delimited list of multiple values). **Note**: The Elimination ID fields are used to eliminate possible matches between records based on custom unique identifiers, such as passport or tax numbers. For more information, see *Oracle Enterprise Data Quality Customer Data Services Pack Guide*.
`eid2`	String	Y	Elimination ID 2 (single or pipe-delimited list of multiple values).
`eid3`	String	Y	Elimination ID 3 (single or pipe-delimited list of multiple values).
`nameid`	String	N	Unique identifier for the name, used to distinguish between different names for the same individual when multiple child entities are used. For more information, see Section 2.3.1, "Using Matching Services."

Attribute Name	Data Type	Supports Multiple Values? (Y/N)	Notes
`title`	String	N	Title
`firstname`	String	N	First Name
`middlename`	String	N	Middle Name
`lastname`	String	N	Last Name
`gender`	String	N	Gender (M or F)
`dob`	String	N	Date of Birth, in any of the formats recognized by the *Date Formats reference data set in EDQ.
`jobtitle`	String	N	Job Title
`homephone`	String	N	Home Phone Number
`workphone`	String	N	Work Phone Number
`mobilephone`	String	N	Mobile Phone Number
`faxphone`	String	N	Fax Number
`alternatephone`	String	Y	Alternative Phone Number - either a single value, or a pipe-delimited list of multiple values.
`alternatephoneid`	String	Y	A single or pipe-delimited list of multiple values corresponding to the ID values in `alternatephone`, which you can use to identify, in the case of a match, which of the values matched, and return this from the match service.
`email`	String	Y	A single value or a pipe-delimited list of multiple email addresses.
`emailid`	String	Y	A single or pipe-delimited list of multiple values corresponding to the ID values in `email`, which you can use to identify, in the case of a match, which of the values matched, and return this from the match service.
`taxnumber`	String	Y	A single value or a pipe-delimited list of multiple tax numbers.
`nationalidnumber`	String	Y	Social Security Number (US) or equivalent, single value or pipe-limited list.
`accountname`	String	N	The name of the account (for example, entity) to which this individual belongs, if relevant.
`accountnameid`	String	N	An ID field for the `accountname` field, which you can use to identify, in the case of a match, which account name was matched upon.

Attribute Name	Data Type	Supports Multiple Values? (Y/N)	Notes
`addressid`	String	N	Unique identifier for the address, used to distinguish between different addresses for the same individual when multiple child entities are used. For more information, see Section 2.3.1, "Using Matching Services."
`address1`	String	N	Address line 1
`address2`	String	N	Address line 2
`address3`	String	N	Address line 3
`address4`	String	N	Address line 4
`dependentlocality`	String	N	A smaller population center data element, dependent on the contents of the `city` field. For example, a Neighborhood in Turkey. For many countries, this attribute is not used.
`doubledependentlocality`	String	N	The smallest population center data element, dependent on both the contents of the `city` and `dependentlocality` fields. For example, a village in the UK. For many countries, this attribute is not used.
`city`	String	N	The locality, town or city of the address.
`subadminarea`	String	N	The smallest geographic data element within a country. For example, a county in the USA.
`adminarea`	String	N	The most common geographic data element within a country. For example, USA State or Canadian Province.
`postalcode`	String	N	Postal or zip code for the address, if relevant for the country. **Note:** With matching services, leading zeroes are stripped only on numeric `postalcodes` to avoid a numeric `postalcode` reinterpreted as a number by an external programs where leading zeroes are automatically stripped. For example, Excel may reformat numeric `postalcodes` as a number by removing the leading zeroes. This is enabled by default in the `edq-cds-daas.properties` Run Profile. If there are any alpha characters present, the leading zeroes are not stripped.
`country`	String	N	Country name or ISO 2 char code.
`clusterlevel`	String	N	1 = limited, 2 = typical, 3 = exhaustive. Used in clustering only. If a value is not supplied, the value defined by the override property is used if set; otherwise the default is 2.

Attribute Name	Data Type	Supports Multiple Values? (Y/N)	Notes
matchthreshold	Number	N	Minimum match rule score to return a result (0-100). Used in matching only. If a value is not supplied, the value defined by the override property in the Run Profile is used if set; otherwise the default is 70.
matchoptions	String	N	For future use.

2.4.1.3 Entity Candidates

Attribute Name	Data Type	Supports Multiple Values? (Y/N)	Notes
candidate	String	N	0 = driving record, 1 = candidate. Used in matching only. All driving records are compared against each other and against each candidate, but candidates are not compared against each other.
entityid	String	N	Unique record identifier. Mandatory, for identifying which records matched in the return interface.
languages	String	Y	3 character Siebel language code. Only used in name standardization to help determine whether a name containing Kanji is Japanese or Chinese.
uid1	String	Y	Unique ID 1 (single or pipe-delimited list of multiple values). **Note**: The Unique ID fields are used to match records based on custom unique identifiers, such as passport or tax numbers. For more information, see *Oracle Enterprise Data Quality Customer Data Services Pack Guide*.
uid2	String	Y	Unique ID 2 (single or pipe-delimited list of multiple values).
uid3	String	Y	Unique ID 3 (single or pipe-delimited list of multiple values).
uid1id	String	Y	A single or pipe-delimited list of multiple values corresponding to the ID values in uid1, which you can use to identify, in the case of a match, which of the values matched, and return this from the match service.
uid2id	String	Y	A single or pipe-delimited list of multiple values corresponding to the ID values in uid2, which you can use to identify, in the case of a match, which of the values matched, and return this from the match service.

Attribute Name	Data Type	Supports Multiple Values? (Y/N)	Notes
uid3id	String	Y	A single or pipe-delimited list of multiple values corresponding to the ID values in uid3, which you can use to identify, in the case of a match, which of the values matched, and return this from the match service.
eid1	String	Y	Elimination ID 1 (single or pipe-delimited list of multiple values). **Note**: The Elimination ID fields are used to eliminate possible matches between records based on custom unique identifiers, such as passport or tax numbers. For more information, see *Oracle Enterprise Data Quality Customer Data Services Pack Guide*.
eid2	String	Y	Elimination ID 2 (single or pipe-delimited list of multiple values).
eid3	String	Y	Elimination ID 3 (single or pipe-delimited list of multiple values).
nameid	String	N	Unique identifier for the name, used to distinguish between different names for the same entity when multiple child entities are used. For more information, see Section 2.3.1, "Using Matching Services."
name	String	N	Organization name, for example, "Oracle Corporation UK".
subname	String	N	Department or site, for example, "Reading" or "Accounts Payable".
phone	String	N	
alternatephone	String	Y	A single or pipe-delimited list of multiple alternative phone number values.
alternatephoneid	String	N	An ID field for the alternatephone multi-value field, which you can use to identify, in the case of a match, which of the values matched, and return this from the match service.
website	String	Y	A single or pipe-delimited list of multiple alternative web site addresses.
taxnumber	String	Y	A single or pipe-delimited list of multiple tax numbers.
vatnumber	String	Y	A single or pipe-delimited list of multiple VAT numbers.
addressid	String	N	Unique identifier for the address, used to distinguish between different addresses for the same entity when multiple child entities are used. For more information, see Section 2.3.1, "Using Matching Services."
address1	String	N	Address line 1

Attribute Name	Data Type	Supports Multiple Values? (Y/N)	Notes
address2	String	N	Address line 2
address3	String	N	Address line 3
address4	String	N	Address line 4
dependentlocality	String	N	A smaller population center data element, dependent on the contents of the city field. For example, Turkish Neighborhood.
doubledependentlocality	String	N	The smallest population center data element, dependent on both the contents of the city and dependentlocality fields. For example, UK Village.
city	String	N	
subadminarea	String	N	The smallest geographic data element within a country. For example, USA County.
adminarea	String	N	The most common geographic data element within a country. For example, USA State or Canadian Province.
postalcode	String	N	Postal or zip code for the address, if relevant for the country. **Note:** With matching services, leading zeroes are stripped only on numeric postalcodes to avoid a numeric postalcode reinterpreted as a number by an external programs where leading zeroes are automatically stripped. For example, Excel may reformat numeric postalcodes as a number by removing the leading zeroes. This is enabled by default in the edq-cds-daas.properties Run Profile. If there are any alpha characters present, the leading zeroes are not stripped.
country	String	N	Country name or ISO 2 char code.
clusterlevel	String	N	1 = limited, 2 = typical, 3 = exhaustive. Used in clustering only.
matchthreshold	Number	N	Minimum match rule score to return a result (0-100). Used in matching only.
matchoptions	String	N	For Future Use.

2.4.1.4 Address Candidates

Attribute Name	Data Type	Supports Multiple Values? (Y/N)	Notes
candidate	String	N	0 = driving record, 1 = candidate. Used in matching only.
addressid	String	N	Unique identifier for the address.

Attribute Name	Data Type	Supports Multiple Values? (Y/N)	Notes
address1	String	N	Address line 1
address2	String	N	Address line 2
address3	String	N	Address line 3
address4	String	N	Address line 4
dependentlocality	String	N	A smaller population center data element, dependent on the contents of the city field. For example, Turkish Neighborhood.
doubledependentlocality	String	N	The smallest population center data element, dependent on both the contents of the city and dependentlocality fields. For example, UK Village.
city	String	N	
subadminarea	String	N	The smallest geographic data element within a country. For example, USA County.
adminarea	String	N	The most common geographic data element within a country. For example, USA State or Canadian Province.
postalcode	String	N	Postal or zip code for the address, if relevant for the country.
country	String	N	Country name or ISO 2 char code.
clusterlevel	String	N	1 = limited, 2 = typical, 3 = exhaustive. Used in clustering only.
matchthreshold	Number	N	Minimum match rule score to return a result (0-100). Used in matching only.
matchoptions	String	N	For future use.

2.4.2 Matches Interface

The Matches interface is used for the output of the matching services in batch and real-time. It is used for individuals, entities and addresses because it contains no attributes specific to any business object.

With Individual and Entity matching, if there are multiple matches between records with the same masterid and matchids (for example, due to multiple matches with different names and addresses), only the strongest match (by match score) is returned for the record pair. Siebel does not currently use the returned mastername id, matchnameid, masteraddressid and matchaddressid attributes, though these may be used in other integrations to display the correct records in the application according to the best matches.

Attribute Name	Data Type	Notes
serverid	String	Server ID. Not applicable to Siebel.
jobid	String	Job ID. Not applicable to Siebel.
masterid	String	Driving record ID. Only used in Batch.

Attribute Name	Data Type	Notes
matchid	String	Matching record ID.
masternameid	String	Driving record name ID. Used to identify which name matched on the driving record, where multiple names were presented.
matchnameid	String	Matching record name ID. Used to identify which name matched on the candidate record, where multiple names were presented.
masteraccountnameid	String	Driving record account name ID. Used to identify which account name matched on the driving record, where multiple account names were presented.
matchaccountnameid	String	Matching record account name ID. Used to identify which account name matched on the candidate record, where multiple account names were presented.
masteraddressid	String	Driving record address ID. Used to identify which address matched on the driving record, where multiple addresses were presented.
matchaddressid	String	Matching record address ID. Used to identify which address matched on the candidate record, where multiple addresses were presented.
masteremailid	String	Driving record email ID. Used to identify which email address matched on the driving record, where multiple emails were presented.
matchemailid	String	Matching record email ID. Used to identify which email address matched on the candidate record, where multiple emails were presented.
masterphonenumberid	String	Driving record phone number ID. Used to identify which phone number matched on the driving record, where multiple phone numbers were presented.
matchphonenumberid	String	Matching record phone number ID. Used to identify which phone number matched on the candidate record, where multiple phone numbers were presented.
masterwebsiteid	String	Driving record website ID. Used to identify which website matched on the driving record, where multiple websites were presented.
matchwebsiteid	String	Matching record website ID. Used to identify which website matched on the candidate record, where multiple websites were presented.
masteruid1id	String	Driving record unique identifier 1 ID. Used to identify which unique identifier 1 (UID1) value matched on the driving record, where multiple UID1 values were presented.
matchuid1id	String	Matching record unique identifier 1 ID. Used to identify which unique identifier 1 (UID1) value matched on the candidate record, where multiple UID1 values were presented.
masteruid2id	String	Driving record unique identifier 2 ID. Used to identify which unique identifier 2 (UID2) value matched on the driving record, where multiple UID2 values were presented.
matchuid2id	String	Matching record unique identifier 2 ID. Used to identify which unique identifier 2 (UID2) value matched on the candidate record, where multiple UID2 values were presented.

Attribute Name	Data Type	Notes
masteruid3id	String	Driving record unique identifier 3 ID. Used to identify which unique identifier 3 (UID3) value matched on the driving record, where multiple UID3 values were presented.
matchuid3id	String	Matching record unique identifier 3 ID. Used to identify which unique identifier 3 (UID3) value matched on the candidate record, where multiple UID3 values were presented.
matchscore	Number	Match score.
rulename	String	Match rule name.
reversedriverflag	String	A flag indicating that an additional, reversed match record has been generated where there is a match between driving records in Batch matching. Valid values are Y and N.

Note:

- In Siebel integrations, the driving record(s) are also returned in the output from real-time matching requests, with a blank match score and rule name. This behavior is controlled by the phase.*.process.*.Return\ Real-time\ Driving\ Record Run Profile property, and therefore could be configured for other types of integration if required.
- So that external applications, such as Siebel, can simply consume the output from batch matching to update both records in a match, CDS batch matching provides two records for each match between driving records. Therefore, if A matches B, a record is returned with masterid A and matchid B, *and* an additional record is generated and returned with masterid B and matchid A. This additionally generated record will have reversedriverflag set to Y in case the external application does not need the additionally generated record.
- The Match Rule Name cannot be displayed in Siebel due to the limitations of the Siebel Data Quality interface that only accepts a returned score related to each matched record.

2.4.3 Cluster Results Interfaces

Two data interfaces are used for the output of the results of clustering; one for batch and one for real-time. They are used for entities, individuals, and addresses as they contain no attributes specific to a particular business object.The batch and real-time interfaces contain similar information, with the main difference being the way in which the results are processed. The Batch and Real-Time Results Interfaces contain similar information, the main difference is the way in which the results are processed.

2.4.3.1 Real-Time Cluster Results Interface

The Real-time Cluster Results interface is used for the output of Clustering Services in real-time. The output values are returned in arrays in no specific order; the clustervalues and clusterlevels array element always correspond.

Attribute Name	Data Type	Notes
`externalid`	String	ID of the individual, entity or address of the clustered record.
`clustervalues`	StringArray	Cluster key value array.
`clusterlevels`	StringArray	Cluster key level array.

2.4.3.2 Batch Cluster Results Interface

The Batch Cluster Results interface is only used in the Batch Clustering service. It differs from the Real-time Cluster Results interface in that it returns one row per cluster value per record, rather than arrays of cluster values for a record.

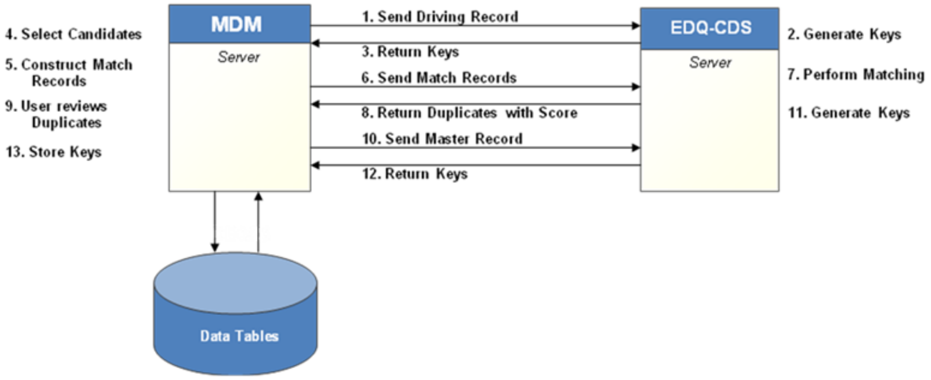
Attribute Name	Data Type	Notes
`serverid`	String	Server ID. Not applicable to Siebel.
`jobid`	String	Job ID. Not applicable to Siebel.
`externalid`	String	ID of the individual, entity, or address of the clustered record.
`clustervalue`	String	Cluster key value.
`clusterlevel`	String	Cluster key level.

2.5 Real-Time Integration

The EDQ-CDS real-time matching services can be called by an external application without any changes to the default configuration. It is the responsibility of the calling application to manage the storage of record cluster keys and to perform the selection of match candidates to be passed to the matching service.

A typical interaction between the calling application (for example, a CRM or Master Data Management [MDM] application) and EDQ-CDS during real-time matching (for example, Contact duplicate prevention) is illustrated as follows:

Figure 2–1 Overview of Expected Integration Architecture with Matching Services



In detail the matching services operate and are used as follows:

- **Send Driving Record** — The application sends the new (driving) record and the configured cluster level to EDQ-CDS.
- **Generate Keys** — EDQ-CDS generates the cluster key(s) for the driving record.

- **Return Keys** — EDQ-CDS returns the driving record's cluster keys to the MDM application.
- **Select Candidates** — The MDM application selects all (candidate) records that share any of the same cluster keys. If no candidates are identified then go to the Store Keys.
- **Construct Match Data** — The MDM application constructs the match data for the driving and candidate records
- **Send Match Records** — The MDM application sends the data for the new (driving) record and candidates to EDQ-CDS.
- **Perform Matching** — EDQ-CDS matches the driving record against the candidates to identify potential duplicates. Each match is assigned a score indicating the strength of match.
- **Return Duplicates with score** — EDQ-CDS returns the IDs of the matched candidates (and scores) to the MDM application. The driving record is also returned, but with a blank score. If no duplicates were identified by EDQ-CDS then go to the Store Keys.
- **User reviews Duplicates** — As indicated.
- **Send Master Record** — If duplicates were identified by EDQ-CDS and selected by the user, then the driving record is merged with the existing duplicate record. If a merge operation occurred then the MDM application sends the new merged (master) record details back to EDQ-CDS.
- **Generate Keys** — EDQ-CDS uses the details of the master record to generate cluster key values.
- **Return Keys** — EDQ-CDS returns the master record's cluster keys to the MDM application.
- **Store Keys** — The MDM application stores the cluster keys for new master record.

3 Using Matching

This chapter describes how you can use the EDQ-CDS matching functionality to match your data.

This chapter includes the following sections:

- Section 3.1, "Objectives of Matching"
- Section 3.2, "Using Clustering"
- Section 3.3, "Using Individual Matching"
- Section 3.4, "Using Entity Matching"
- Section 3.5, "Using ID Matching"
- Section 3.6, "Using Address Matching"

EDQ-CDS has been designed to match customer data that exhibits real-world variability. All relevant matches in the data set are presented back and appropriately scored according to the likelihood of a match between records. To do this, it uses a variety of different mechanisms, including the application of a wide range of matching algorithms on the data as it is presented, as well as matching techniques on derived forms of the data.

For example, names presented in one writing system are matched both using this writing system and also using a transformed version of the name, providing effective cross-script matching. Similarly, addresses are matched in near raw form (after standardization of international address words and phrases, and after removal of filler words), but also by extracting and matching key information from the address, such as the likely building number, sub-building number, and postal code.

3.1 Objectives of Matching

In general, the matching services provided by EDQ-CDS are designed for duplicate prevention, rather than searching. This means that the intention of the out-of-the-box services is to intervene when a record is added to a system if it appears that it may already exist. The implication of this is that the matching services are focused on much more than a single attribute (such as Name) and deliberately do not cast as wide a net as a typical search operation. There may be other records in the system that are not matched but which have similar details, perhaps even exactly the same name, but where the secondary identification information indicates that a match is unlikely. In these cases, EDQ-CDS aims to minimize the additional work for users or data stewards whose role it is to resolve possible matches. This makes the product ideally suited to operate as the data quality protection component of a Master Data Management system, such as Oracle Customer Hub, where the purpose of the services is to link as many records as possible together automatically with as little noise as possible. The same is true for a Customer Relationship Management system, such as Siebel.

Note : It is possible to change the configuration of EDQ-CDS in order to perform more exhaustive matching. This is mainly designed for use with low volume, high value data sets that do not necessarily offer sufficient secondary information (beyond name fields).

3.1.1 Multiple Locales and Languages

EDQ-CDS has been designed as a multi-locale system, and uses international and culture-sensitive name transcription, transliteration and variant recognition techniques, as well as using international dictionaries when standardizing and matching addresses.

The system is designed to work with international data, and provides international dictionaries of name and address standardizations for this purpose. The international 'Latin script' dictionaries provide coverage of the following 'base' locales, amongst others:

- United States and Canada
- United Kingdom
- France
- Germany
- Italy
- Spain
- Portugal
- Brazil
- Greece
- Ireland
- Austria
- Turkey
- South Africa
- Australia and New Zealand
- Scandinavia
- Argentina
- Mexico

In addition to these base locales, EDQ-CDS provides specific optional capabilities for advanced handling of data from the following locales:

- Arab World (Arabic and Mixed Arabic/Latin)
- Japan (Kanji, Katakana and Hiragana)
- China (Simplified and Traditional Chinese)
- Russia

- Korea (Hangul)

The set of enabled languages is determined by the configuration of the `EDQ-CDS - Initialize Reference Data` project, so that the same reference data may be used by any number of EDQ-CDS matching servers. By default, reference data sets for the base locales are pre-initialized in the EDQ server landing area, but these can be easily overwritten either by unzipping `cdslists-initialized-full.zip` over these files (to provide coverage for all supported locales and languages) or by configuring and running the Initialization job.

3.1.2 Uses of Matching

The Matching processes included in EDQ-CDS are designed primarily for the following use cases:

- Duplicate Prevention - uses the Cluster Generation and Matching web services to prevent duplicate records being entered into applications.
- Regular Batch Matching for Duplicate Removal - uses the Batch Matching job, run on all, or a subset of, data in an application, and links records together for potential merge.

It is also possible to use the Batch Matching processes as a template for the deduplication of records before they are loaded into a system. This is likely to require additional configuration, and use of EDQ. In such circumstances the best practice is to understand the data before matching using data profiling and audit techniques, such as those available in the EDQ-CDS Data Quality Health Check. In most cases, the set of enabled match rules will need some tuning towards the specifics of the in-scope data in order to provide the optimum balance between performance and effectiveness. It may also be necessary to use EDQ's Match Review application to review possible matches, and construct rules for merging records together.

> **Note :** EDQ-CDS does not provide any out-of-the-box merging (or survivorship) configuration, because in the two main use cases, merging is performed by the calling application after matches have been identified.

3.1.2.1 Duplicate Prevention

EDQ-CDS uses stateless web services for duplicate prevention to avoid complex replication and synchronization of large volume customer data. This places the following requirements on the application integrating with EDQ:

1. Storage of Cluster Key tables for each type of record (for example, Contacts or Accounts). These are normally thin tables with two columns - the Primary Key of the record and the Cluster Key. The table must allow for multiple key values per record.
2. Functionality to select and construct candidate records to submit to the Matching service. This involves:
   a. Querying the Cluster Key table for the relevant record, and finding all records that share a key value with the driving record.
   b. Constructing the data that is required for matching for each of these records.
   c. Submitting these Candidate records together with the driving record to the Matching service.

Optimum Duplicate Prevention Process Flow

In order to access the full capabilities of EDQ-CDS for duplicate prevention, the integration should work as follows:

1. To prepare the system for real-time duplicate prevention, key values are generated for each record in Batch using the Cluster Key Generation process. This can occur either when migrating the data into the application, or as a batch process to generate the key values into the application's Cluster Key tables.
2. When a record is added or updated in the application, the Cluster Key Generation service is called in real-time, and returns a number of cluster key values for the record.
3. The application then selects candidate records (those records which share a common key with the driving record) using the existing stored keys and submits them along with the driving record to the Matching service.
4. The Matching service decides which of the candidates are a likely match to the driving record and returns the ids of these records, and a score indicating the strength of match.
5. The application then decides how to consume the matching results; for example, whether to 'auto-match' or present possible matches to the user so that a decision can be made whether or not to continue with inserting a record, or merge it with an existing record.
6. If the record is merged with another record to create a changed master record, an additional call should be made to the Cluster Generation service in order to re-generate the correct cluster key values before committing the record.

In this model, complex multi-locale EDQ techniques are used to generate the keys and ensure that the right balance between performance and matching effectiveness is maintained, while ensuring that the calling application retains control of data integrity and transactional commits.

3.1.2.2 Batch Matching

When working with Siebel CRM, Siebel's Data Quality Manager is used to instigate batch jobs and a shared staging database is used to write records for matching and to consume match results. The EDQ-CDS batch matching processes automatically adjust to Siebel's 'Full Match' (match all records against each other) and 'Incremental Match' (match a subset of records against all of their selected candidates) modes.

3.1.3 Match Tuning

In EDQ-CDS matching, it is not necessary to be overly concerned with which identifiers will be populated in the data that is worked with. EDQ-CDS does not use a weighted algorithm that will place unnecessary emphasis on unpopulated data, and so does not require adjustment for this.

Matching works by examining all of the available data and attempting various ways to form a match. The matching design builds in the knowledge of how strong an identifier is likely to be based on real world principles. A significant advantage of this approach is that there is normally no need to apply algorithmic tuning adjustments, the results of which are hard to predict. Instead, match tuning is normally a matter of performing one of the following tasks, which will have a more predictable effect on match results:

- Adjusting the clustering configuration.
- Enabling or disabling a provided rule.

- Adjusting the score that a specific rule returns.
- Inserting a new rule (perhaps a stronger or weaker version of an existing rule).

> **Note:**
>
> - Even when inserting a new rule, it may well be possible to use existing comparisons and comparison results rather than adding new comparisons, though both are possible.
> - For batch matching of large data sets, it is recommended that redundant match rules [whose priority score is lower than the `matchthreshold` setting] are disabled as this will yield significant performance improvements.

3.2 Using Clustering

Clustering is used to minimize the work that is performed during the final stage of matching. It works by splitting the records into tranches (clusters), based on similarities in significant data fields. Only subsets of the data which share similar characteristics (and will therefore be placed in the same cluster) will be compared on a record-by-record basis during matching.

If loose clusters are used, there will be a large number of records in each cluster. This means that there is a reduced risk that true matches will be missed, but also that a greater amount of processing will be required to compare all the clustered records. It will also increase the number of false positives being returned, which will require extra time to assess. A tighter clustering strategy will result in smaller cluster groups and hence a reduced processing time, but will increase the likelihood that some true matches will not be detected.

EDQ-CDS is supplied with a number of different clustering algorithms for individual and entity data that use different combinations of key data fields in their construction. Each clustering algorithm has been assigned a unique prefix code for easy identification, and to ensure keys from different clusters are not identical. This prefix and all data elements within a cluster value are separated with the caret symbol (^).

3.2.1 Cluster Level Algorithms

All clustering algorithms are assigned a cluster level which relates to the tightness of the cluster groups that it generates with typical data. The following cluster level settings are available:

Level	Name	Usage
1	Limited	Useful for tight matching with large volumes of data.
2	Typical	The recommended setting for most applications providing a balance of performance with match tolerance.
3	Exhaustive	Required for the loosest possible matching where there is high risk if matches are missed.

3.2.2 Cluster Values

The format of the cluster values is:

```
[Prefix]^[Cluster Level]^[Cluster Component Value]
```

Additional components are further delimited with the ^ symbol:

```
[Prefix]^[Cluster Level]^[Cluster Component Value 1]^[Cluster Component
Value 2]
```

3.2.3 Individual Clustering Algorithms

The following clustering algorithms are provided for matching individual data:

Prefix	Cluster Name	Level	Description
`LMP`	Family Name Meta, Postal Code, Address1	1	4-character double-metaphone of the surname + First 5 characters of the postal code + First 3 characters of address1. **Note:** With matching services, leading zeroes are stripped only on numeric `postalcodes` to avoid a numeric `postalcode` reinterpreted as a number by an external programs where leading zeroes are automatically stripped. For example, Excel may reformat numeric `postalcodes` as a number by removing the leading zeroes. This is enabled by default in the `edq-cds-daas.properties` Run Profile. If there are any alpha characters present, the leading zeroes are not stripped.
`PLN`	Phone last N	1	Last N digits of the phone/fax/work/mobile number; set to 6.
`EF9`	Email first 9	1	First 9 characters of the email address.
`TAX`	Tax Number	1	First 10 characters of the tax number.

`EID1`
`EID2`
`EID3`
	Elimination Identifier	1	All non-alphanumeric characters are removed.

`UID1`
`UID2`
`UID3`
	Unique Identifier	1	All non-alphanumeric characters are removed.

`NID`	National Identifier	1	First 10 characters of the National ID number.
`FLP`	Given Names standardized, Family Name, Postal Code	2	First character of the standardized given name + First 3 characters of the family name + First 5 characters of the postal code.
`FLY`	Given Names standardized, Family Name, City	2	First 3 characters of the standardized given name + First 3 characters of the family name + First 10 characters of the city name.
`FA1`	Given Names standardized, Address1	2	First 3 characters of the standardized given name + First 10 characters of address line 1.
`LMC`	Family Name Meta, First Company word	2	First 4 characters of the family name + First word of the account name.
`A5F`	Address1, Address2, City	3	First 5 characters of address line 1 + First 5 characters of address line 2 + First 5 characters of the city name.
`OSP`	Original Script name, Postal Code	3	First 4 characters of the original script name + First 4 characters of the postal code.

Prefix	Cluster Name	Level	Description
FLM	Full Name Meta	3	The full name tokens are sorted and then the double-metaphone algorithm is applied to generate tokens of up to 3 characters in length. For each ordered pair of tokens, a cluster value is generated that is the concatenation of the two metaphone tokens.

Note : The clustering algorithms use data attributes that have been normalized (for example, converted to upper case and symbols stripped) and have had whitespace removed. This allows clustering and matching to be performed in a case-insensitive manner and to be tolerant of the spacing within attributes.

3.2.3.1 Examples

The following record data is used to provide examples of the cluster values that are generated by the individual clustering algorithms:

Attribute	Value
firstname	Jim
middlename	Frederick
lastname	Smith
mobilephone	077777 123456
email	j.smith@mymail.com
taxnumber	888666444
accountname	Acme Ltd
address1	14 high St
city	Cambridge
postalcode	CB1 2AB
uid1	00021-53563
eid1	gbr0008873323
nationalidnumber	AB 12 34 56 C

The cluster values that are generated using a `clusterlevel` setting of `3` (Exhaustive) are as follows:

Cluster Prefix	Cluster Values
LMP	LMP^1^SM0^CB12A^14H
PLN	PLN^1^123456
EF9	EF9^1^J.SMITH@M
TAX	TAX^1^888666444
EID1	EID1^1^GBR0008873323
UID1	UID1^1^0002153563

Cluster Prefix	Cluster Values
NID	NID^1^AB123456C
FLP	FLP^2^J^SMI^CB12A
FLY	FLY^2^JAM^SMI^CAMBRIDGE
FA1	FA1^2^JAM^14HIGH
LMC	LMC^2^SM0^ACME
A5F	A5F^3^14HIG^^CAMBR
FLM	FLM^3^FRTJMS FLM^3^FRTSM0 FLM^3^JMSSM0

3.2.4 Entity Clustering Algorithms

The following clustering algorithms are provided for matching entity data:

Prefix	Cluster Name	Level	Description
APC	Address 1 and Postal Code	1	First 3 characters of address line 1 + First 5 characters of the postal code. **Note:** With matching services, leading zeroes are stripped only on numeric postalcodes to avoid a numeric postalcode reinterpreted as a number by an external programs where leading zeroes are automatically stripped. For example, Excel may reformat numeric postalcodes as a number by removing the leading zeroes. This is enabled by default in the edq-cds-daas.properties Run Profile. If there are any alpha characters present, the leading zeroes are not stripped.
TAX	Tax Number	1	First 10 characters of the tax number.
VAT	VAT Number	1	First 10 characters of the VAT number.
PLN	Phone Last N Digits	1	Last N digits of the phone/fax/work/mobile number; set to 6.
NSD	Name and Sub-name	1	First 30 characters of the concatenation of the distilled name and sub-name.
EID1 EID2 EID3	Elimination Identifier	1	All non-alphanumeric characters are removed.
UID1 UID2 UID3	Unique Identifier	1	All non-alphanumeric characters are removed.
NPC	Name and Postal Code	2	First 4 characters of the name + First 3 characters of the postal code.
NMP	Name Metaphone, Address 1 and Postal Code	2	For each token in the distilled name: 4-character double metaphone of the token + First 4 characters of address line 1 + First 3 characters of the postal code.
WS	Website Stem	2	Website address without the top level domain name, common address prefix and any page portion of the url.

Prefix	Cluster Name	Level	Description
NMA	Full Name metaphone, Address No Numbers	2	Full name double-metaphone 4: address lines 1-4, concatenated, number words stripped, denoised including hyphens, first 10 characters.
NSM	Name metaphone and Sub-name metaphone	3	4-character double-metaphone of the name + 4-character double-metaphone of the sub-name.
OS	Original Script	3	For each token in the original script name: First 5 characters of the name token. For Chinese, Japanese and Korean script each token will generate a cluster value.
NST	Name and Sub-name Tokens	3	Generate a cluster value for the 4-character double metaphone of each token in the distilled name and distilled sub-name.

Note : The clustering algorithms use data attributes that have been normalized (for example, converted to upper case and symbols stripped) and whitespace removed. This allows clustering and matching to be performed in a case-insensitive manner and be tolerant to the spacing within attributes.

3.2.4.1 Examples

The following record data is used to provide examples of the cluster values that are generated by the entity clustering algorithms:

Attribute	Value
name	Oracle UK
subname	Cambridge
phone	+441223228400
website	http://www.oracle.com/uk
taxnumber	RGW432D243224
vatnumber	999111
address1	296 Cambridge Science Park
city	Cambridge
postalcode	CB4 0WD
uid1	00021-53563
eid1	gbr0008873323

The cluster values that are generated using a `clusterlevel` setting of `3` (Exhaustive) are as follows:

Cluster Prefix	Cluster Values
APC	APC^1^296^CB40W
TAX	TAX^1^RGW432D243
VAT	VAT^1^999111

Cluster Prefix	Cluster Values
PLN	PLN^1^228400
NSD	NSD^1^ORACLECAMBRIDGE
EID1	EID1^1^GBR0008873323
UID1	UID1^1^0002153563
NPC	NPC^2^ORAC^CB4
NMP	NMP^2^ARKL^296C^CB4
WS	WS^2^ORACLE
NMA	NMA^2^ARKL^CAMBRIDGES
NSM	NSM^3^ARKL^KMPR
NST	NST^3^ARKL
	NST^3^KMPR

3.2.5 Address Clustering Algorithms

The following clustering algorithms are provided for matching address data:

Prefix	Cluster Name	Level	Description
PPC	Premise and Postal Code	1	Premise, first number word, or if no number word first 8 of premise. If no premise first 8 of address1 + Postal code first 5, if no postal code, first 8 of city. **Note:** With matching services, leading zeroes are stripped only on numeric `postalcodes` to avoid a numeric `postalcode` reinterpreted as a number by an external programs where leading zeroes are automatically stripped. For example, Excel may reformat numeric `postalcodes` as a number by removing the leading zeroes. This is enabled by default in the `edq-cds-daas.properties` Run Profile. If there are any alpha characters present, the leading zeroes are not stripped.
PC	Postal Code	3	PostalCode, whole value.
A12	Address1 and Address2	2	Address1 distilled, first 10. Address2 distilled, first 10.
A1C	Address1 and City	2	Address1 distilled, first 5. City, First 8.
FA	Full Address	1	Full Address distilled, first 12. Cluster not generated if there are fewer than 12 characters.
FAN	Full Address No Number Words	2	Address lines 1-4, concatenated, number words stripped, first 10. Cluster not generated if there are fewer than 10 characters.

Note:

- A **Number word** is a word with one or more numbers within it. for example, 234 and 2A are both number words.
- The clustering algorithms use data attributes that have been normalized (for example, converted to upper case and symbols stripped) and whitespace removed. This allows clustering and matching to be performed in a case-insensitive manner and be tolerant to the spacing within attributes.

3.2.5.1 Examples

The following record data is used to provide examples of the cluster values that are generated by the address clustering algorithms:

Attribute	Value
`address1`	2529 CINCINNATI ST
`address2`	APT 6
`city`	LOS ANGELES
`adminarea`	CA
`postalcode`	90033

Note : During Cluster Key generation, `ST` is distilled out of the `address1` field, and `APT` is distilled out of the `address2` field. This is because they are common addressing components that are less important identifiers than the remainder of the address line, and removing them produces more accurate clusters.

The cluster values that are generated using a `clusterlevel` setting of `3` (Exhaustive) are as follows:

Cluster Prefix	Cluster Values
`PPC`	`PPC^1^2529^90033`
`PC`	`PC^3^90033`
`A12`	`A12^2^2529CINCIN^6`
`A1C`	`A1C^2^2529C^LOSANGEL`
`FA`	`FA^1^2529CINCINNA`
`FAN`	`FAN^2^CINCINNATI`

3.3 Using Individual Matching

The matching design for individuals in CDS is based on Name as the primary identifier for individuals, purely because it should always be present, rather than because it is a strong identifier. However, in general, the aim of the services is only to return matches where the Name matches (using one of a wide variety of matching techniques) and at least one other secondary identifier (such as Email Address,

Address, Date of Birth, any Phone Number, or Social Security Number) also matches (again using a variety of techniques).

In large data sets, there are likely to be a large number of individuals with the same or similar names, but if none of the secondary information matches, it is highly unlikely to be the same person. Even if the secondary information is unpopulated on one or both records, and a match is a little more likely in theory, the absence of the information makes it nearly impossible for a user or data steward to determine if the individual is the same even if in direct contact with the individual. For this reason, matches such as this are not considered using the default rules.

However, matches where only one of the secondary identifiers match (for example, where an email address matches but the address is entirely different) are presented, and offer a strong route to improved data quality, as it is very likely to be the same person (they could, for example, have simply moved house).

3.3.1 Name-Only Matching

To allow matches on sparse data, you may want matching for individuals after entering only names (no other details). Name-only match rules are enabled by default in the CDS matching process.

In order to make use of these rules, the calling application needs to make the following adjustments to properties, by either the input value to the interface or via the run profile property:

- Lower the `matchthreshold` property (suggested value 50), since name-only match rules have a match score of less than the default (70).
- Change the `clusterlevel` property to level 3, to include the FLM (First Name, Last Name Metaphone) cluster - which is the only cluster that creates cluster values on name only.

You are recommended to use these changes in real-time, since they are likely to produce many loose matches. See the following recommendations for data set sizes.

Maximum Size of Data Set	Recommended Maximum Cluster Level	Batch/Real-time	Maximum Expected Cluster Size
1 million records	3 (Exhaustive)	Real-time only	500 (FLM cluster)
40 million records	2 (Typical)	Batch or real time	500
40+ million records	1 (Limited)	Batch or real time	Depends on data size

3.3.2 Using Individual Name Matching

The rules for matching individual names include the use of pre-matching transformations and various matching comparisons in order to handle the following types of variance between different representations of what may be the same individual name:

- Names written in different writing systems/scripts, for example, '?????' and 'Zoran'.
- Variants of the same name, for example, 'Bill' and 'William'.
- Different levels of name completeness, for example, 'Joseph Andrew Harris' and 'Joseph Harris'.

- Name tokens in a different order, for example, 'Lacazette Jacques' and 'Jacques Lacazette'.
- Abbreviated forms of names, for example, 'Chris' and 'Christian'.
- Typographic differences, for example, 'Michael' and 'Micheal'.
- The use of initials, for example, 'A M' and 'Alexander Martin'.
- Changes of surname due to marriage, for example, 'Paula Jones' and 'Paula Lewis' at the same address.
- Various combinations of the above types of variance.

The match rules are organized into groups of rules where all rules in each group have the same name matching rule, but different rules on secondary identifiers (such as address, email address, phone number and so on). The following table lists all of the groups, and therefore all of the name matching rules used.

Note : In this table the pipe character is used to indicate a separator between the input given name and family name attributes (for example, Given Name= Martin, Family Name=Smith is written as 'Martin | Smith'). Where no pipe character is used, this means the Full Name is used in the match rule.

Name Matching Rule	**Example Name Match**
Script full name exact	????? ????????????? ??????? =????? ????????????? ???????
Name exact	Martin | Fox = Martin | Fox
Standardized given name	Bill | Lewis = William | Lewis
Given name abbreviated	Chris | Smith = Christina | Smith
Standardized given name abbreviated	Abell | Hernandez = Abelson | Hernandez
Script full name any order	??????? ????? ????????????? =????? ????????????? ???????
Given name similar and sounds like	Yngrid | Martin = Ingrid | Martin
First name similar and sounds like	Yngrid Elisabeth | Martin = Ingrid Martin
Additional given names	Michael John | Smith = John | Smith
Standardized full name	Mehmood Mahomed = Mahmoud Mohammed
Script full name has additional names	????? ??????? =????? ????????????? ???????
Additional names	Mary Jones Steward = Mary Jones
Script full name typos	????? ????????????? ??????? =????? ????????????? ????????
Standardized given name abbreviated; family name typos	Abell | Hernandez = Abelson | Hernandes
Full name typos, all words	Mary Cloire Jonez = Mary Claire Jones
First name first three; family name typos	Ros Susan | Jonez = Rose Susan | Jones
Full name initials in order; additional names	G A | Smith = Gordon Alfred | Smith
Standardized first name only; female	Jacklin | Jones = Jacqueline | Smith

3.3.3 Using Individual Secondary Identifier Matching

For each individual name match rule, and therefore within each match rule group, a number of match rules exist, each with different levels of matching on secondary identifiers, such as Company Name, Email Address, Address, Date of Birth, and phone numbers.

The following table is a guide to the criteria needed to match on each rule. These criteria are combined with the name matching rule in order to determine which match rule is hit, and therefore the score of the match.

Note:

- All matching on secondary identifiers uses prepared versions of the secondary identifiers; for example, all address match rules are applied on prepared versions of the addresses, after various word and phrase standardizations are applied.
- A rule is not included for every combination of secondary identifiers matching; for example, there is no rule that requires a match on *both* Date of Birth and Phone number, as both of the identifiers are suitably strong that even if only one of the attributes match, the match should be generated and scored highly.

Secondary Identifier Match Rule	Description
DOB; e-mail	Date of birth and e-mail match exactly.
Address; e-mail	Address and e-mail match exactly.
E-mail; phone number	E-mail and any phone number match exactly.
Company; address	All tokens in the shorter company name match in the longer company name, and the address matches exactly.
Tax number	Tax number matches exactly.
National ID number	National ID number matches exactly.
E-mail	E-mail matches exactly.
Address	Address matches exactly.
Phone	Any phone number matches exactly.
Premise; subpremise; postal code starts with	Address matches by extracted premise, subpremise and postal code **Note:** With matching services, leading zeroes are stripped only on numeric `postalcodes` to avoid a numeric `postalcode` reinterpreted as a number by an external programs where leading zeroes are automatically stripped. For example, Excel may reformat numeric `postalcodes` as a number by removing the leading zeroes. This is enabled by default in the `edq-cds-daas.properties` Run Profile. If there are any alpha characters present, the leading zeroes are not stripped.
Premise; no subpremise; postal code starts with	Address matches by extracted premise and postal code, and there is no data in either `subpremise` field.
DOB	Date of birth matches exactly.

Secondary Identifier Match Rule	Description
Phone last N digits	Any phone number matches using the last N digits (tby default, the last 6 digits.)
Company; postal code	All tokens in the shorter company name match in the longer company name, and the postal code matches exactly.
Address all words	All words in the shorter address match in the longer address.
DOB similar	Dates of birth are a close match (a day/month transposition match using the default comparison settings).
Tax number typos	Tax number matches with a Character Edit Distance of 1 or 2.
National ID number typos	National ID number matches with a Character Edit Distance of 1 or 2.
E-mail typos	E-mail matches with a Character Edit Distance of 1 or 2.
Address all words typos	All words in the shorter address match in the longer address with a Character Error Tolerance of 20%.
Address similar; postal code	Address matches with a Character Match Percentage of 65 or more, and the postal code matches exactly.
Address similar; first address one word	Address matches with a Character Match Percentage of 65 or more, and there is at least one token match in the first line of the address.
Company	All tokens in the shorter company name match in the longer company name.

It is also possible to perform matching or elimination of Individual records using custom unique identifiers, see Section 3.5, "Using ID Matching."

3.4 Using Entity Matching

As with individuals, the design for EDQ-CDS Entity matching is based around the name, but with acknowledgement that the name is a weaker identifier in the context of an Entity, as Entities change name more frequently than individuals. Also, there tends to be less secondary information on Entity records. As a result, Entity matching is based largely on Name and Location (Address) attributes, though matching on additional identifiers such as URLs and Tax Numbers is also provided.

Note : It is significantly harder to match entities (as opposed to individuals) between different writing systems, as the process of transliteration — and even transcription — is much less likely to be successful. Very often, the only way to recognize that a company is the same when written in two different languages is to hold huge dictionaries of all possible company names and their appropriate translations (rather than transliterations or transcriptions). In most cases, such data is simply not available though if it is available it can be plugged into EDQ-CDS in order to improve results.

3.4.1 Using Entity Name Matching

The rules for matching entity names include the use of pre-matching transformations and various matching comparisons in order to handle the following types of variance between different representations of what may be the same entity name:

- Entity names written in different writing systems.
- Entity names with or without suffixes, for example, 'Oracle LTD' and 'Oracle'.
- Entity names containing abbreviated terms or suffixes, for example, 'Oracle Limited' and 'Oracle LTD'.
- Character order and spelling differences/errors in entity names, for example, 'Oracle' and 'Oralce'.
- Entity names with different levels of name completeness, for example, 'ABC Technology Consultants LTD' and 'ABC Technology LTD'.
- Entity name tokens appearing in a different order, for example, 'Cambridge Science Park LTD' and 'Science Park Cambridge'.
- Entity Names where part or all of the name is reduced to an acronym, for example, 'Oracle Catering' and 'O.C.'.
- Potential matches where there is no name match at all but strongly matching secondary identifiers (for example, if a company has been renamed there may be two records with identical VAT numbers).

The match rules are organized into groups of rules where all rules in each group have the same name matching rule, but different rules on secondary identifiers (such as address, or URL). The following table lists all of the groups, and therefore all of the entity name matching rules used.

> **Note :** In the following table, where a name matching rule uses the 'full name', this means it applies to the entity full name identifier, a concatenation of the entity name and sub-name attributes. The pipe (|) character is used to separate the entity name and sub-name were the sub-name attribute is required to provide an example match.

Entity Name Matching Rule	Example Entity Name Match
Script full name exact	???????? ????-? ??? = ???????? ????-? ???
Full name exact	TCHIBO GMBH = TCHIBO GMBH
Standardized full name exact	ORACLE UK LTD | READING = ORACLE UK LIMITED | READING
Script full name without suffixes exact	????????? ??????? ??????????????? ?????????? = ????????? ??????? ???????????????
Full name without suffixes exact	ORACLE = ORACLE CORPORATION
Full name without suffixes similar and sounds like	ORACLE CAMBRIDGE SCIENCE PARK = ORACLE CAMBRIDGE PARK SCIENCE
Script full name out of order	???????? ????-? ??? = ???????? ??? ????-?
Script full name without suffixes all words out of order	????????? ??????? = ????????? ???????????????

Entity Name Matching Rule	Example Entity Name Match
Full name without suffixes all words out of order	CAMBRIDGE SCIENCE PARK LTD = SCIENCE PARK CAMBRIDGE
Script full name has additional names	????????? ??????? ?????????? | ?????? = ????????? ??????? ??????????????? ?????????? | ??????
Script entity name without suffixes exact	????????? ??????? ??????????????? ?????????? |?????? = ????????? ??????? ??????????????? ?????????? | ???????
Entity name without suffixes exact	ORACLE CORPORATION | CAMBRIDGE = ORACLE | READING
Full name all words shorter with typos	Oracle Inc | Cambridge =Oracl | Cambridge
Script entity name without suffixes starts with	????????? ??????? ??????????????? ??????????| ?????? = ????????? ??????? ??????????| ???????
Entity name without suffixes starts with	ABC TECHNOLOGY CONSULTANTS LTD = ABC TECHNOLOGY LTD
Script full name without suffixes all words shorter with typos	????????? ??????? ??????????????? ??????????= ????????? ????????
Full name without suffixes all words shorter with typos	Federal Mogull | Camshafts Inc = Federal Mogul Camshafts Castings Ltd
Script full name typos	????????? ??????? ??????????| ?????? = ????? ??????? ??????????| ??????
Full name typos	ABD SERVICES LTD = ABC SERVICES LTD
Script full name without suffixes typos	????????? ??????? ??????????????? ?????????? = ??????? ??????? ???????????????
Full name without suffixes typos	ABD ENGINEERING LTD = ABC ENGINEERING
Script entity name without suffixes starts with	????????? ??????? ??????????????? ?????????? = ?????????? ?????????
Entity name without suffixes starts with	ABC LIMITED| CAMBRIDGE = ABC PHARMACEUTICALS LIMITED| READING
Non-name rules	N/A - These rules are used in order to raise matches where only the secondary data (such as, VAT number or exact address) matches.
Standardized full name acronym exact	CSC= Computer Science Corporation
Full name without suffixes acronym exact	CSC = Computer Science Collaborations Ltd
Full name without suffixes acronym contains	US House of Representatives = United States House of Representatives
Entity name without suffixes loose typos	SASKATCHEWAN MINISTRY OF HEALTH = MANITOBA MINISTRY OF HEALTH
Entity name without suffixes first token	DANVERS BANCORP INC = DANVERS MUNICIPAL FEDERAL CREDIT UNION

3.4.2 Using Entity Secondary Identifier Matching

For each Entity Name match rule, and therefore within each match rule group, a number of match rules exist. Each has different levels of matching on secondary identifiers, such as Address, Website Address, Tax Number, VAT Number or Phone Number.

The following table is a guide to the criteria needed to match on each rule. These criteria are combined with the entity name matching rule in order to determine which match rule is triggered, and therefore the score of the match.

Note:

- All matching on secondary identifiers uses prepared versions of the secondary identifiers; for example, all address match rules are applied on prepared versions of the addresses, after various word and phrase standardizations are applied.
- A rule is not included for every combination of secondary identifiers matching - for example, there is no rule that requires a match on **both** Tax Number and VAT Number, as both of the identifiers are suitably strong that even if only one of the attributes match, the match should be generated and scored highly.
- Not all Secondary Identifier Match Rules are enabled by default for all Match Rule Groups (especially the latter Groups) because the combination of looser name rule and looser secondary rule would lead to an increased incidence of vague matches. You can enable these rules in EDQ or using a run profile.

Secondary Identifier Match Rule	Description
Address	Address matches exactly.
Premise; subpremise; postal code starts with	Address matches by extracted premise, subpremise and postal code. **Note:** With matching services, leading zeroes are stripped only on numeric `postalcodes` to avoid a numeric `postalcode` reinterpreted as a number by an external programs where leading zeroes are automatically stripped. For example, Excel may reformat numeric `postalcodes` as a number by removing the leading zeroes. If there are any alpha characters present, the leading zeroes are not stripped.
Premise; no subpremise; postal code starts with	Address matches by extracted premise and postal code, and there is no data in either `subpremise` field.
Address all words	All words in the shorter address match in the longer address.
Address all words typos	All words in the shorter address match in the longer address with a Character Error Tolerance of 20%.
Website; phone number	The website address and any phone number match exactly.
Tax number	The tax number matches exactly.
VAT number	The VAT number matches exactly.
Address 1 typo; city; country	The address is similar and both the city and country matches exactly.
Address similar; postal code	Address matches with a Character Match Percentage of 65 or more, and the postal code matches exactly.
Phone	Any phone number matches exactly.

Secondary Identifier Match Rule	Description
Phone last N digits	Any phone number matches using the last N digits (by default, the last 6 digits.)
Tax number typos	The tax number matches with a Character Edit Distance of 1 or 2.
VAT number typos	The VAT number matches with a Character Edit Distance of 1 or 2.
Postal code	The postal code matches exactly.
City; country	The city and country match exactly.
Website	The website address matches exactly.
Website stem	The stem part of the website address matches exactly.
City	The full city name matches exactly.
Address similar; first address one word	Address matches with a Character Match Percentage of 65 or more, at least one word matches in the first address line.
Country	The country name matches exactly.
No address	The address matches when it is missing in one or both of the records.
Address conflict	The addresses do not match at all. By default, this rule is only active for the first few primary identifier groups involving an exact name match. For example, if the addresses are different you must be confident that the names are the same and understand that it is a very loose match.

It is also possible to perform matching or elimination of Entity records using custom unique identifiers, see Section 3.5, "Using ID Matching."

3.5 Using ID Matching

The ID Matching rules in EDQ-CDS allow matching (or elimination) based solely on custom unique identifiers, without the need for a name match of some kind.

Matching and elimination is provided for Entity and Individual Matching, but not Address Matching.

Note:

- Unique ID (UID) matching is always performed before EID matching. Therefore, if two records are matched by unique identifiers, they cannot then be eliminated.
- These identifiers are always compared in standardized form; for example, values that differ only in case or additional non-alphanumeric character are considered identical. for example, the following values are identical for the purposes of ID matching:
  - AB123456789
  - ab123-456-789
  - ab12345 6789
  - ab#123456789

3.5.1 Using Unique ID Matching

The UID Match rules are held in the `[I005] UID` and `[E005] UID` match group of the Individual and Entity Match processes respectively. For example, the match groups for Individual matches are as follows:

- `[I005A] Match UID1`
- `[I005B] Match UID2`
- `[I005C] Match UID3`

To use these rules, map the required data in the records to one or more of the **uid** attributes. The matching rules will always match two records sharing a common unique identifier, even if none of the other attributes match.

Note:

- The `uid` attributes accept multiple values in the form of a pipe delimited list. A match will be returned between two records if any one of a multiple set of attribute values is matched.
- Matching between `uid` attributes is not possible, for example, `uid1` values cannot be matched with `uid2` or `uid3` values.

Example

The `Passport Number` field in a series of records is configured as the `uid1` attribute. Therefore, the following records are returned as a match:

Record ID	First Name	Last Name	`uid1` (Passport Number)	Match?
1	Fred	Smith	12345678	Yes
2	John	Doe	12345678	Yes

The following records with multiple values in the `uid1` field are also matched:

Record ID	First Name	Last Name	`uid1` (Passport Number)	Match?
1	Fred	Smith	12312312 | 67867867	Yes

Record ID	First Name	Last Name	uid1 (Passport Number)	Match?
2	John	Doe	67867867 | 23423423	Yes

The SSN field for the same set of records is configured as the uid2 attribute. The uid1 and uid2 fields are not cross matched; even though the uid1 value of Record 1 matches the uid2 value of Record 2:

Record ID	First Name	Last Name	uid1 (Passport Number)	uid2 (SSN)	Match?
1	Fred	Smith	12312312	67867867	No
2	John	Doe	67867867	12312312	No

3.5.2 Using Elimination IDs

The Elimination ID (EID) Match rules are held in the [ELIM015] EID ELIMINATIONS group of the Entity and Individual Match processes:

- [ELIM015A] ELIMINATE EID1
- [ELIM015B] ELIMINATE EID2
- [ELIM015C] ELIMINATE EID3

To use these rules, map the required data in the records to one or more of the eid attributes. The EID matching rules will always return a "No Match" result for two records that do not share a common value in an eid attribute, even if all other attributes match. The exception to this is if the two records are matched using a uid attribute, as UID matching is performed before EID matching.

Note:

- eid attributes accept multiple values in the form of a pipe delimited list. A "No Match" result will be returned between two records if the attribute is populated for both records and the two records have no EID value in common.
- Eliminating possible matches by comparing values between different eid attributes is not possible, for example, eid1 values cannot be compared with eid2 or eid3 values.

Example

The SSN field in a series of records is configured as the eid1 attribute. Therefore, the following records are eliminated as a possible match:

Record ID	First Name	Last Name	eid1 (SSN)	Eliminate?
1	John	Doe	12345678	Yes
2	John	Doe	87654321	Yes

The following records with multiple values in the eid1 field are also eliminated as a possible match, as none of the values match:

Record ID	First Name	Last Name	eid1 (SSN)	Eliminate?
1	John	Doe	12312312 | 23423423	Yes
2	John	Doe	45645645 | 67867867	Yes

The Passport field for the same set of records is configured as the eid2 attribute. The eid1 and eid2 fields are not compared, and therefore a "No Match" result is returned and the records are eliminated as a possible match:

Record ID	First Name	Last Name	eid1 (SSN)	eid2 (Passport Number)	Eliminate?
1	John	Doe	12312312	67867867	Yes
2	John	Doe	67867867	12312312	Yes

Finally, there are two identical values in the eid1 fields of the following records, and therefore they are *not* eliminated as a possible match:

Record ID	First Name	Last Name	eid1 (SSN)	Eliminate?
1	John	Doe	12312312 | 23423423	No
2	John	Doe	45645645 | 12312312	No

3.6 Using Address Matching

The rules for matching addresses include the use of pre-matching transformations and various matching comparisons in order to handle variance between different representations of what may be the same address, for example:

- Addresses containing abbreviated terms or suffixes.
- Character order and spelling differences/errors in addresses.
- Addresses with different levels of completeness.
- Addresses where extracted premise and sub-premise match, and other components of the address are in a different order or missing on one side.

The following table lists all of the rules provided:

Address Match Rule Code	Address Match Rule Description
[A010]	Address exact, postal code exact
[A020]	Address exact, no postal code
[A030]	Address lines 1 and 2 exact, city exact, postal code exact
[A040]	Address lines 1 and 2 exact, city exact, postal code starts with
[A050]	Address all words, subpremise exact, premise exact, postal code exact
[A060]	Address all words, subpremise exact, premise exact, postal code no conflict
[A070]	Address 1 exact, address 2 no conflict, subpremise exact, premise exact postal code exact

Address Match Rule Code	Address Match Rule Description
[A080]	Address 1 exact, address 2 no conflict, subpremise exact, premise exact, postal code starts with
[A090]	Address 1 exact, address 2 no conflict, subpremise exact, premise exact, postal code no conflict
[A100]	Address all words typos, subpremise exact, premise exact, postal code exact
[A110]	Address all words typos, subpremise exact, premise exact, postal code no conflict
[A120]	Address 1 exact, address 2 no conflict, postal code exact
[A130]	Address 1 exact, address 2 no conflict, postal code starts with
[A140]	Address 1 exact, subpremise exact, premise exact, postal code exact
[A150]	Address 1 exact, subpremise exact, premise exact, postal code starts with
[A160]	Address 1 exact, subpremise no conflict, premise no conflict, postal code exact
[A170]	Address 1 exact, subpremise no conflict, premise no conflict, postal code starts with
[A180]	Address all words, subpremise no conflict, premise no conflict, postal code exact
[A190]	Address all words, subpremise no conflict, premise no conflict, postal code no conflict
[A200]	Address 1 all words, subpremise exact, premise exact, postal code exact
[A210]	Address 1 all words, subpremise exact, premise exact, postal code starts with
[A220]	Address 1 all words, subpremise no conflict, premise no conflict, postal code exact
[A230]	Address 1 all words, subpremise no conflict, premise no conflict, postal code starts with
[A240]	Address1 common string 7+, subpremise exact, premise exact, postal code exact
[A250]	Address all words, postal code exact
[A260]	Address similar, subpremise exact, premise exact, postal code exact
[A270]	Address 1 all words, address 2 no conflict, postal code exact
[A280]	Address 1 all words, address 2 no conflict, postal code starts with
[A290]	Address all words typos, postal code exact
[A300]	Address 1 exact, subpremise exact, premise exact, postal code no conflict
[A310]	Address 1 all words, subpremise exact, premise exact, postal code no conflict
[A320]	Address 1 exact, postal code exact
[A330]	Address 1 exact, postal code starts with
[A340]	Subpremise exact, premise exact; postal code exact
[A350]	Subpremise exact, premise exact, postal code starts with
[A360]	Address all words
[A370]	Address all words typos
[A380]	Address similar; postal code
[A390]	Address similar; first address one word

The following table provides examples of matches by Match Rule Code only, with the key fields highlighted in bold text where required:

Address Match Rule Code	Address Component	Record	Matched Record
[A010]	address1	901 GOLF CLUB RD	901 GOLF CLUB RD
	city	WESTWOOD	WESTWOOD
	subadminarea	PLUMAS	PLUMAS
	adminarea	CA	CA
	postalcode	96137	96137
	country	US	US
[A020]	As for [A010], but the **postalcode** field in both records is blank.		
[A030]	address1	1201 BEECH ST	1201 BEECH ST
	address2	APT 104F	APT 104F
	city	PALO ALTO	PALO ALTO
	subadminarea	**SANTA CLARA**	**SAN MATEO**
	adminarea	CA	CA
	postalcode	94303	94303
	country	US	US
[A040]	As [A030], except the v field in one address starts with the same characters as the other, but is not identical.		
[A050]	address1	**5 Hogskoleringen**	**Hogskoleringen 5**
	city	Trondheim	Trondheim
	adminarea		SØR-TRØNDELAG
	postalcode	**7491**	**7491**
	country	Norway	Norway
[A060]	As [A050], except one or both of the **postalcode** fields are blank.		
[A070]	address1	**Heinrichboeckingstr 10-14**	**Heinrichboeckingstr 10-14**
	address2	Service Zentrum Merzig	
	city	Saarbrücken	Saarbrücken
	adminarea		SAARLAND
	postalcode	66121	66121
	country	Germany	Germany
[A080]	Same as [A070], except the postalcode field in one address starts with the same characters as the **postalcode** field in the other, but is not identical.		
[A090]	Same as [A070], except one or both of the `postalcode` fields are blank.		
[A100]	address1	**HOGSKOLERINGE 5**	**HOGSKOLERINGEN 5**
	city	Trondheim	Trondheim
	postalcode	9491	9491
	country	Norway	Norway

Address Match Rule Code	Address Component	Record	Matched Record
[A110]	Same as [A100], except one or both of the **postalcode** fields are blank.		
[A120]	address1	**Marshfield Bank**	**Marshfield Bank**
	address2	**WOOLSTANWOOD**	
	city	Crewe	Crewe
	postalcode	**CW28UY**	**CW28UY**
	country	UK	UK
[A130]	Same as [A120], except the **postalcode** field in one address starts with the same characters as the **postalcode** field in the other, but is not identical.		
[A140]	address1	**Apt Y302**	**APT Y302**
	address2	**1605** Sherringtowne Ave	**1605** Sherington Ave
	city	NEWPORT BEACH	NEWPORT BEACH
	adminarea	Orange	Orange
	postalcode	**92663-9087**	**92663-9087**
	country	US	US
[A150]	Same as [A140], except the **postalcode** field in one address starts with the same characters as the **postalcode** field in the other, but is not identical.		
[A160]	address1	1728 **Corporate Xing**	1728 **Corporate Xing**
	address2	**Suite1**	
	city	O Fallon	O Fallon
	adminarea	ILLINOIS	IL
	postalcode	62269-3734	62269-3734
	city	US	US
[A170]	Same as [A160], except the **postalcode** field in one address starts with the same characters as the **postalcode** field in the other, but is not identical.		
[A180]	address1	Block 16	16 Dunsinane Ave
	address2	Dunsinane Avenue	
	address3	Dunsinane Industrial Estate	
	city	Dunsinane	Dunsinane
	postalcode	DD23QT	DD23QT
	country	UK	UK
[A190]	As [A180], except one or both of the **postalcode** fields are blank.		
[A200]	address1	**26701 QUAIL CRK**	**26701 QUAIL CRK APT 107**
	address2	**APT 107**	
	city	ALISO VIEJO	LAGUNA HILLS
	postalcode	92656-1089	92656-1089
	country	US	US
[A210]	Same as [A200], except the **postalcode** field in one address starts with the same characters as the **postalcode** field in the other, but is not identical.		

Address Match Rule Code	Address Component	Record	Matched Record
[A220]	address1	**Folkes Road**	**Unit 12 Folkes Road**
	address2	Hayes Trading Estate	Lye
	address3	Lye	
	city	Stourbridge	Stourbridge
	postalcode	**DY98RN**	**DY98RN**
	country	UK	UK
[A230]	Same as [A220], except the **postalcode** field in one address starts with the same characters as the **postalcode** field in the other, but is not identical.		
[A240]	address1	**101/61 NAWANAKORN INDUSTRY**	**101/61 NAVANAKORN INDUSTRY**
	address2	SELFLEMENT PHAHONYOTHIN	PAHOLYOTHIN KLONGNUENG
	city	KLONGLAUNG	KHLONG LUANG
	postalcode	**12120**	**12120**
	country	Thailand	Thailand
[A250]	address1	Blyth House	Blyth House
	address2	**130** Hordern Road	Hordern Road
	city	Wolverhampton	Wolverhampton
	postalcode	WV60HS	WV60HS
	country	UK	UK
[A260]	address1	**21001** State Route 739	**21001** Sr Rt **739**
	address2	7	
	city	Raymond	Raymond
	postalcode	**43067**	**43067**
	country	United States	United States
[A270]	address1	Lancaster House Aviation Way	Aviation Way
	address2		Southend Airport
	city	SOUTHEND ON SEA	SOUTHEND ON SEA
	postalcode	SS26UN	SS26UN
	country	UK	UK
[A280]	Same as [A270], except the **postalcode** field in one address starts with the same characters as the **postalcode** field in the other, but is not identical.		
[A290]	address1	**Blythe House**	**Blyth House**
	address2	130 **Hordern Road**	**Hordern Road**
	city	Wolverhampton	Wolverhampton
	postalcode	**WV60HS**	**WV60HS**
	country	UK	UK
[A300]	Same as [A140], except one or both of the **postalcode** fields are blank.		

Address Match Rule Code	Address Component	Record	Matched Record
[A310]	Same as [A200], except one of both of the **postalcode** fields are blank.		
[A320]	address1	**Network House**	**Network House**
	address2	1 Ariel Way	Wood Lane
	city	London	London
	postalcode	**W127SL**	**W127SL**
	country	UK	UK
[A330]	Same as [A320], except the **postalcode** field in one address starts with the same characters as the **postalcode** field in the other, but is not identical.		
[A340]	address1	College Business Park	College Business Park
	address2	Park	Coldhams Lane
	city	Cambridge	
	postalcode	**CB13HD**	**CB13HD**
	country	United Kingdom	United Kingdom
[A350]	Same as [A340], except the **postalcode** field in one address starts with the same characters as the **postalcode** field in the other, but is not identical.		
[A360]	address1	938 Miller St	Medical Ctr Blvd
	address2	**Medical Center Boulevard**	
	city	Winston Salem	Winston- Salem
	postalcode	27157	27157
	country	United States	United States
[A370]	address1	**Humberstone Avenue**	24 **Humberston Avenue**
	address2	**Humberstone**	**Humberston**
	city	GRIMSBY	GRIMSBY
	postalcode	DN364SX	DN364SP
	country	UK	UK
[A380]	address1	5**Sidings Court**	Greyfriars House
	address2	White Rose Way	**Sidings Court**
	city	DONCASTER	DONCASTER
	postalcode	**DN45NU**	**DN45NU**
	country	UK	UK
[A390]	address1	120 **Howard** St	120 **Howard** St
	address2		STE 200
	city	San Fransisco	San Fransisco
	adminarea	CA	CA
	postalcode	94105-1622	94105-1615
	country	United States	United States

4

Customizing Customer Data Services Pack

This chapter describes how EDQ-CDS can be customized to take advantage of some of the more advanced features of the product.

This chapter includes the following sections:

EDQ-CDS has been designed to perform well with minimal customization. Ready-to-use, the application can perform clustering and matching of individual, entity and address data in connected supported applications with little or no configuration changes required.

4.1 Using Stand-Alone Batch Matching

EDQ-CDS is designed to process customer data from any external system or stand-alone source. By default, pre-configured batch jobs are provided that work with a set of staging tables. Reconfiguring the product to process data from other sources, such as a text file, is straightforward.

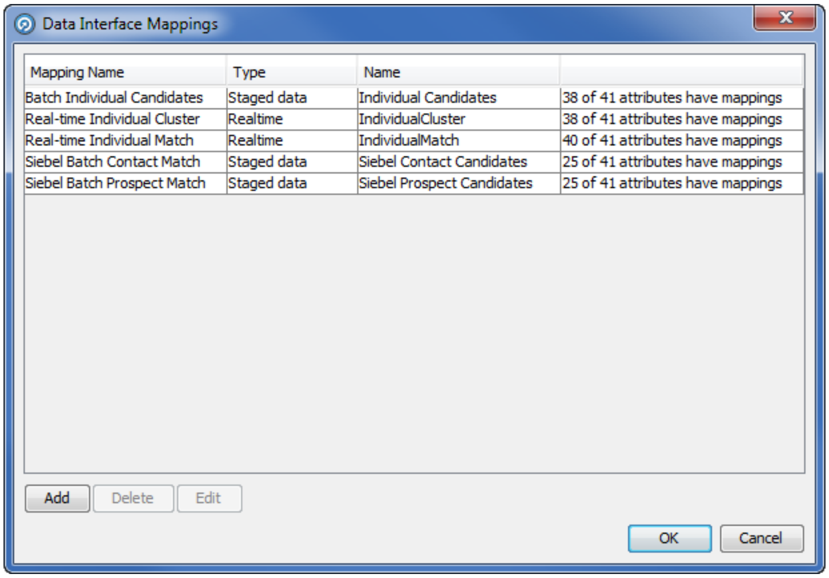
In order to reuse the batch data matching services provided, it is necessary to create new input and output mappings for the data interfaces. The following sections use examples that demonstrate how to do this and how to run matching using a modified copy of an existing job configuration.

4.1.1 Using Stand-Alone Individual Batch Matching

You can create a new stand-alone individual batch matching job using the following example steps:

1. Ensure that no jobs are currently running.
2. In the EDQ-CDS project, create a new server-side data store named **File In: Individuals** that points to the structured text file containing the customer data to be processed. It is important that this is created as a server-side data store in order to be used within a job definition.
3. Create a new snapshot named **Individuals** using the **File In: Individuals** data store as a source.
4. Create the Input Data Interface mappings as follows:

a. Right-click the **Individual Candidates** data interface and select **Mappings...** to open the **Data Interface Mappings** dialog.

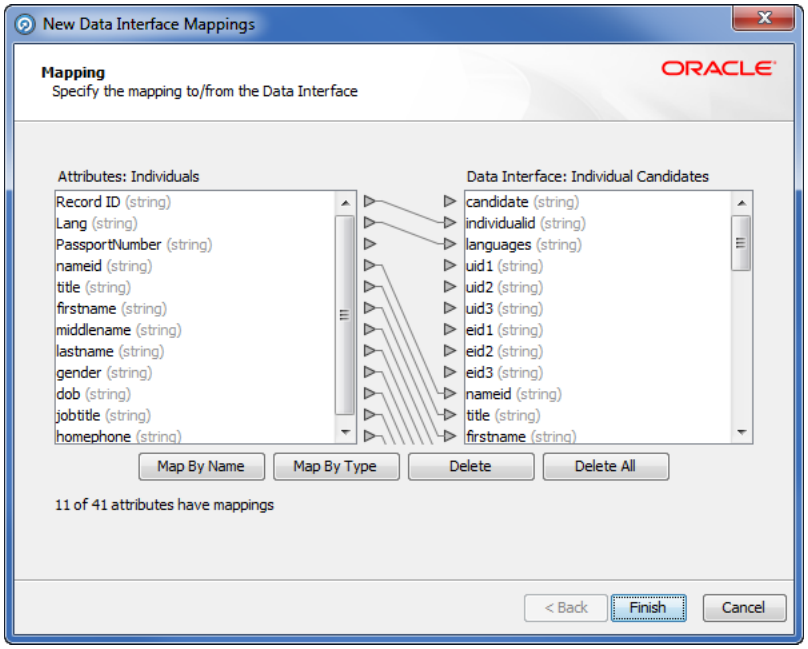


Mapping Name	Type	Name	
Batch Individual Candidates	Staged data	Individual Candidates	38 of 41 attributes have mappings
Real-time Individual Cluster	Realtime	IndividualCluster	38 of 41 attributes have mappings
Real-time Individual Match	Realtime	IndividualMatch	40 of 41 attributes have mappings
Siebel Batch Contact Match	Staged data	Siebel Contact Candidates	25 of 41 attributes have mappings
Siebel Batch Prospect Match	Staged data	Siebel Prospect Candidates	25 of 41 attributes have mappings

b. Click **Add** to open the **New Data Interface Mappings** dialog.

c. Select the **Individuals** snapshot as the source and click **Next**. The Staged data default type is used.

d. Map the Customer Data Attributes on the left of the dialog to the Data Interface Attributes on the right as follows:



Mapping

Specify the mapping to/from the Data Interface

Attributes: Individuals

- Record ID (string)
- Lang (string)
- PassportNumber (string)
- nameid (string)
- title (string)
- firstname (string)
- middlename (string)
- lastname (string)
- gender (string)
- dob (string)
- jobtitle (string)
- homephone (string)

Data Interface: Individual Candidates

- candidate (string)
- individualid (string)
- languages (string)
- uid1 (string)
- uid2 (string)
- uid3 (string)
- eid1 (string)
- eid2 (string)
- eid3 (string)
- nameid (string)
- title (string)
- firstname (string)

11 of 41 attributes have mappings

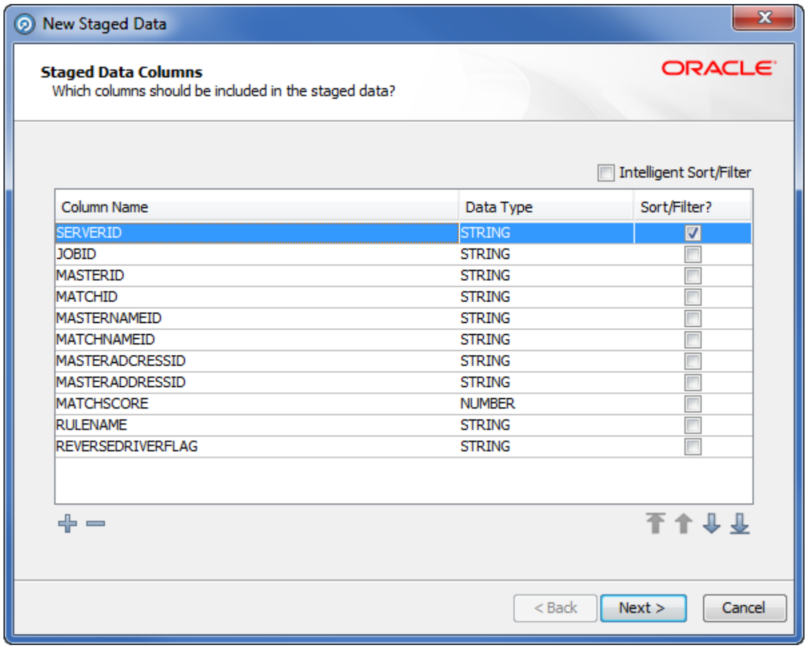
Note : In some instances, it may be necessary to construct a process that reads from the snapshot and reshapes the data to match the Data Interface, see Section 4.1.2, "Converting Data to the Interface Format."

e. Click **Next**.

f. Name the data interface mapping **Individual Candidates** and click **Finish** to save.

g. Click **OK**.

5. Create a new Staged Data named **Individual Matches** with the following columns:



Staged Data Columns

Which columns should be included in the staged data?

□ Intelligent Sort/Filter

Column Name	Data Type	Sort/Filter?
SERVERID	STRING	☑
JOBID	STRING	□
MASTERID	STRING	□
MATCHID	STRING	□
MASTERNAMEID	STRING	□
MATCHNAMEID	STRING	□
MASTERADCRESSID	STRING	□
MASTERADDRESSID	STRING	□
MATCHSCORE	NUMBER	□
RULENAME	STRING	□
REVERSEDRIVERFLAG	STRING	□

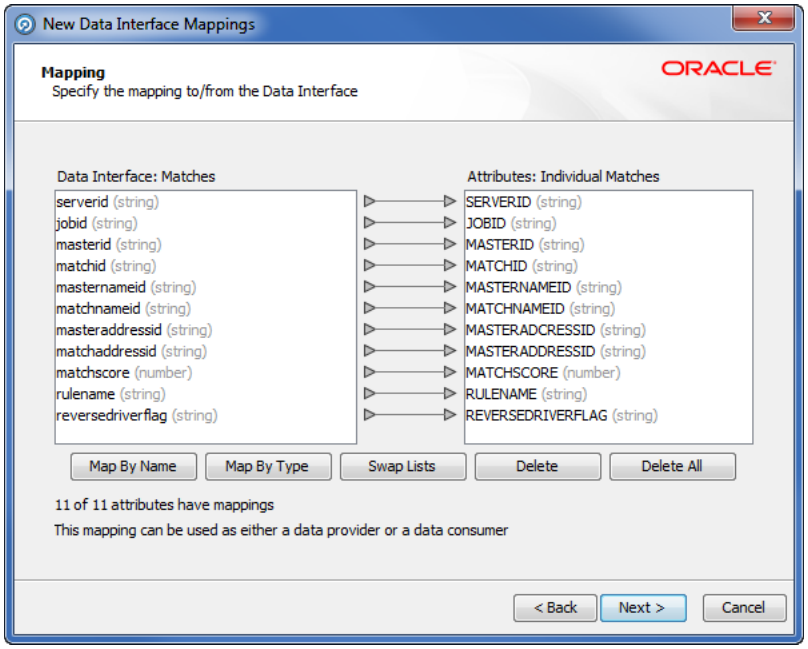
6. Create the Output Data Interface mappings as follows:

a. Right-click the **Matches** data interface and select **Mappings...** to open the **Data Interface Mappings** dialog.

b. Click **Add** to open the **New Data Interface Mappings** dialog.

c. Select the **Individual Matches** staged data as the target and click **Next**.

d. Map the **Matches** data interface attributes on the left to the **Result Staged Data** attributes on the right as required.



New Data Interface Mappings

Mapping

Specify the mapping to/from the Data Interface

Data Interface: Matches	Attributes: Individual Matches
serverid (string)	SERVERID (string)
jobid (string)	JOBID (string)
masterid (string)	MASTERID (string)
matchid (string)	MATCHID (string)
masternameid (string)	MASTERNAMEID (string)
matchnameid (string)	MATCHNAMEID (string)
masteraddressid (string)	MASTERADCRESSID (string)
matchaddressid (string)	MASTERADDRESSID (string)
matchscore (number)	MATCHSCORE (number)
rulename (string)	RULENAME (string)
reversedriverflag (string)	REVERSEDRIVERFLAG (string)

11 of 11 attributes have mappings

This mapping can be used as either a data provider or a data consumer

e. Click **Next**.

f. Name the Data Interface mapping **Individual Matches** and give it a description, then click **Finish**.

g. Click **OK** to close the dialog.

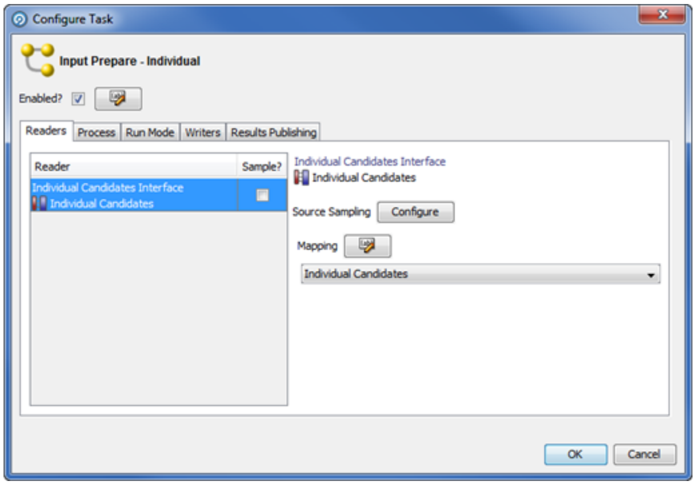
7. Create a new server-side delimited text data store called **File Out: Individual Matches** to use as a target for the match results. Alternatively, the data can be written to a database if required.

8. Create a new export called **Matches to File Out: Individual Matches** that uses the **Matches** data interface as the source to export from, and the **File Out: Individual Matches** as the target for the export.

9. Create and configure a job to run matching as follows:

a. Create a copy of the **Batch Individual Match** job, rename it **Batch Individual Match using Text File**, and then open it.

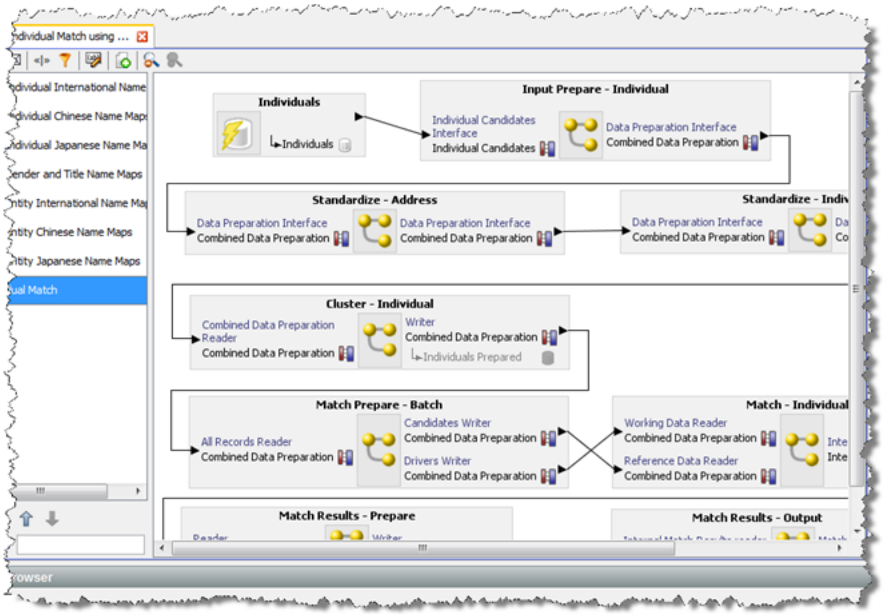
b. Open the **Individual Match** job phase, change the source of the input data by double-clicking on the **Individual Candidates** data interface and selecting the **Individual Candidates** mapping.



c. Click **OK** to apply the changes. The job configuration is modified accordingly and the old snapshot and staged data items are disconnected.

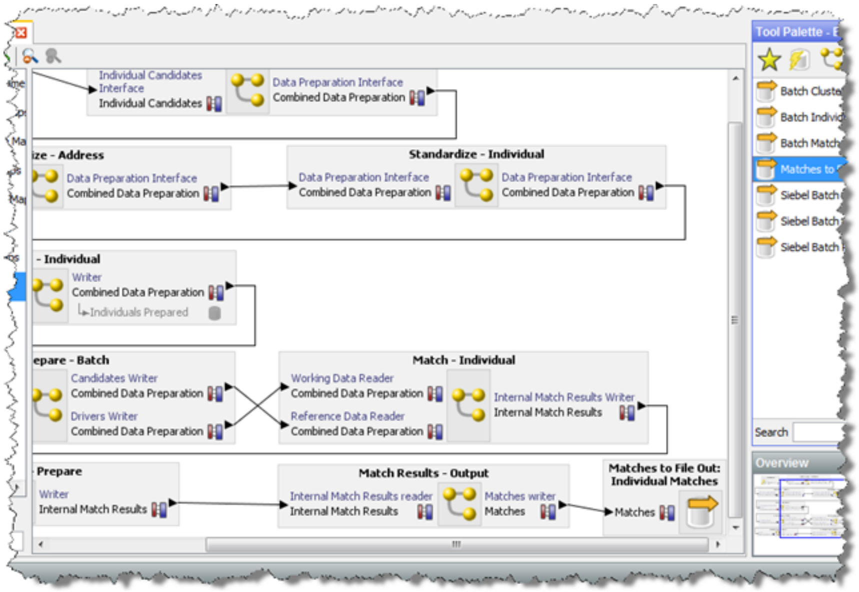
d. Delete the **Individual Candidates** snapshot task.

e. Drag the **Individuals** snapshot from the **Snapshot** in the **Tool Palette** into the open job phase and make sure it is connected to the **Individual Candidates** mapping.


ndividual Match using ...
dividual International Name
dividual Chinese Name Maps
dividual Japanese Name Ma
ender and Title Name Maps
ntity International Name Ma
ntity Chinese Name Maps
ntity Japanese Name Maps
ual Match
Individuals
Individuals
Input Prepare - Individual
Individual Candidates Interface
Individual Candidates
Data Preparation Interface
Combined Data Preparation
Standardize - Address
Data Preparation Interface
Combined Data Preparation
Data Preparation Interface
Combined Data Preparation
Standardize - Indiv
Data Preparation Interface
Combined Data Preparation
Cluster - Individual
Combined Data Preparation Reader
Combined Data Preparation
Writer
Combined Data Preparation
Individuals Prepared
Match Prepare - Batch
All Records Reader
Combined Data Preparation
Candidates Writer
Combined Data Preparation
Drivers Writer
Combined Data Preparation
Match - Individual
Working Data Reader
Combined Data Preparation
Reference Data Reader
Combined Data Preparation
Match Results - Prepare
Match Results - Output
rowser

f. Drag the **Matches to File Out: Individual Matches** export task from the **Export** in the **Tool Palette** into the open job phase and connect it to **Match Results - Output**.

g. Delete the **Batch Matches** export task.


Individual Candidates Interface
Individual Candidates
Data Preparation Interface
Combined Data Preparation
ize - Address
Data Preparation Interface
Combined Data Preparation
Standardize - Individual
Data Preparation Interface
Combined Data Preparation
Data Preparation Interface
Combined Data Preparation
- Individual
Writer
Combined Data Preparation
Individuals Prepared
epare - Batch
Candidates Writer
Combined Data Preparation
Drivers Writer
Combined Data Preparation
Match - Individual
Working Data Reader
Combined Data Preparation
Reference Data Reader
Combined Data Preparation
Internal Match Results Writer
Internal Match Results
Prepare
Writer
Internal Match Results
Match Results - Output
Internal Match Results reader
Internal Match Results
Matches writer
Matches
Matches to File Out: Individual Matches
Matches
Tool Palette - E
Batch Cluste
Batch Individ
Batch Match
Matches to
Siebel Batch
Siebel Batch
Siebel Batch
Search
Overview

10. Close the job and save the configuration changes.

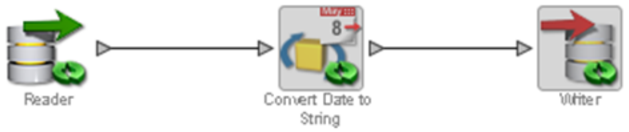
4.1.2 Converting Data to the Interface Format

It may not always be possible to directly map the input source to the candidates interface if:

- fields are of the wrong data type (for example, "Date of Birth" in a date field); or
- fields need transforming to a compatible format/structure (for example, Individual names in a full name field).

If this is the case, then the input data should be run through a custom EDQ process to convert the data as appropriate as in the following example steps:

1. Ensure that no jobs are currently running.
2. Create a data store and snapshot for the input data as in steps 2 and 3 from Section 4.1, "Using Stand-Alone Batch Matching."
3. In the EDQ-CDS project, right-click the **Processes** node in the Project Browser and select **New Process...** to open the **New Process** wizard.
4. Select the snapshot created in step 2 as the data source.
5. Click **Next**.
6. On the last page of the wizard, rename the process **Transform Individuals**, then click **Finish** button to create the process.
7. On the Process canvas, add the necessary processors to transform the data to the interface format. For example, use a **Convert Date to String** processor to convert a date of birth in date format to the required format for the Candidates interface (for example, either yyyyMMdd, MM/dd/yyyy, yyyy-MM-dd or dd-MMM-yy).
8. Add a Writer processor to the process canvas and connect it to the process data stream:



9. In the **Writer Configuration** dialog, select the **Individual Candidates** data interface and map the attributes accordingly.

10. Create and configure a new job as follows:

    a. Make a copy of **Batch Individual Match** job, renaming it **Batch Transformed Individual Match**.

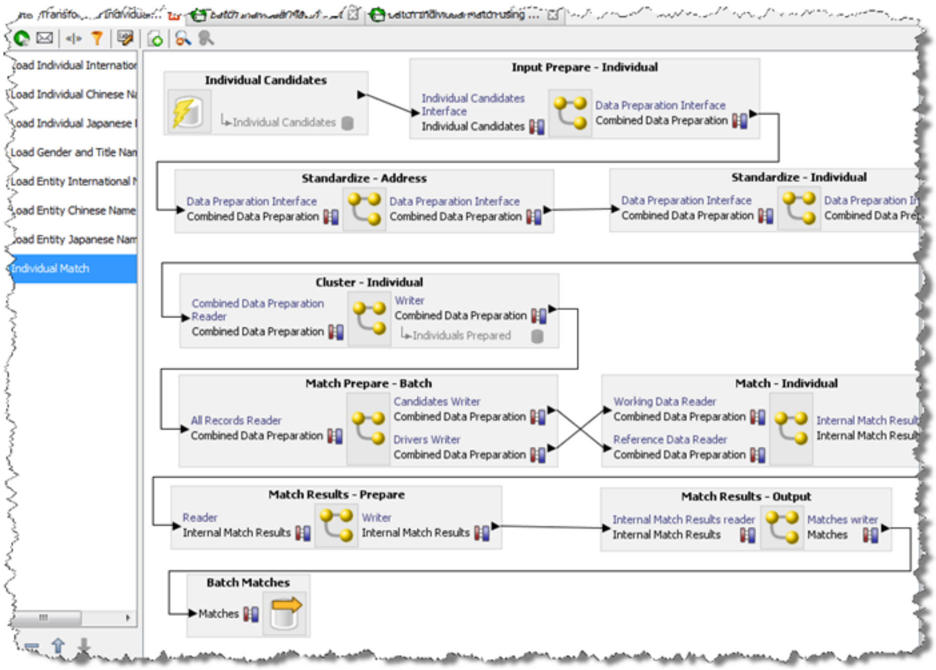
    b. Open the new job.

    c. Double-click on the **Individual Match** job phase.

    d. Drag the **Individuals** snapshot task from the **Snapshot** tool palette onto the **Individual Match** phase of the job.

    e. Double-click the **Individual Candidates** interface and select the **Individual Candidates** mapping.

    f. Click **OK** to apply the changes. The job configuration will be modified accordingly and the snapshot and staged data items will be disconnected. Delete both these items by deleting the Snapshot task. The start of the job phase should now appear as follows:



    g. Use steps 9.d. - 10 of Section 4.1, "Using Stand-Alone Batch Matching" from step 9.d onwards, remembering to modify the job configuration to include the new transformation process and use the modified data interface mappings.

4.2 Using Cleaning Services

The cleaning processes provided with EDQ-CDS are provided as templates only, with the exception of the Address Cleaning process which is fully functional and uses

EDQ-AV for address verification and standardization. The Individual and Entity cleaning processes are intended to be customized to meet the data standardization requirements of the implementation.

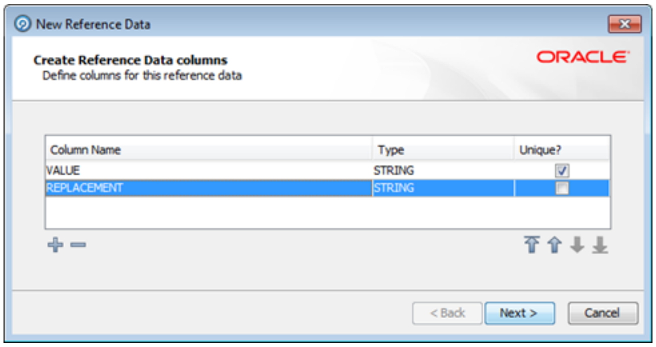
4.2.1 Customizing the Cleaning Services

The examples in the following sections demonstrate modifying the cleaning services provided with EDQ-CDS.

4.2.1.1 Standardizing Job Titles

Modify the Individual Cleaning service to standardize job titles as in the following example steps.

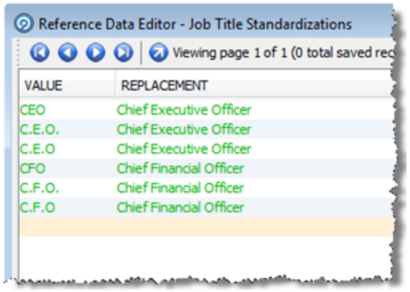
1. Ensure that no jobs are currently running.
2. In the EDQ-CDS project, create a new Reference Data set with the columns as follows:



Create Reference Data columns
Define columns for this reference data

Column Name	Type	Unique?
VALUE	STRING	☑
REPLACEMENT	STRING	☐

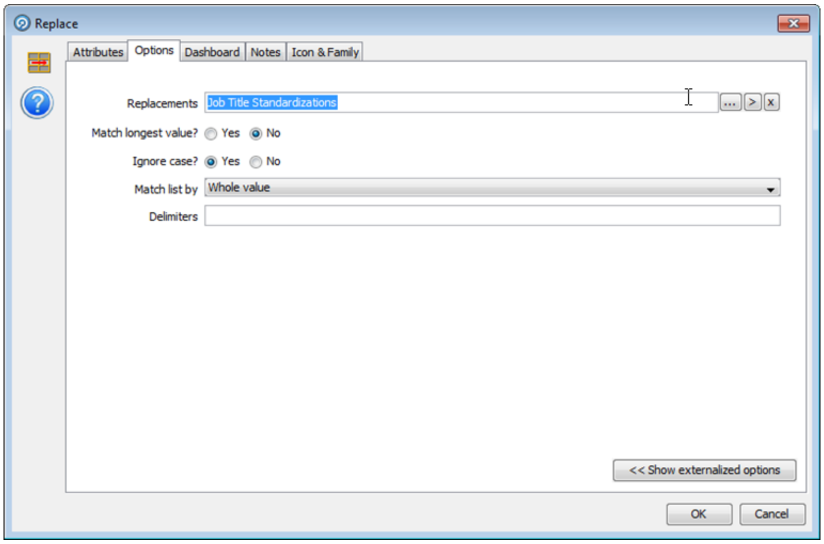
3. Click **Next** through the **New Reference Data** wizard with the name **Job Title Standardizations**.
4. Click **Finish** to close the wizard. The **Reference Data Editor** dialog opens.
5. Add the required job title standardizations; for example:



Viewing page 1 of 1 (0 total saved rec

VALUE	REPLACEMENT
CEO	Chief Executive Officer
C.E.O.	Chief Executive Officer
C.E.O	Chief Executive Officer
CFO	Chief Financial Officer
C.F.O.	Chief Financial Officer
C.F.O	Chief Financial Officer

6. Open the **Clean - Individual** process.
7. Add a new **Replace** processor to the Process Canvas and connect it to the output of the **Upper Case the Name Attributes** processor.
8. In the **Processor Configuration** dialog, set the **jobtitle** attribute as the Input field, and on the Options tab select the **Job Title Standardizations** Reference Data in the **Replacements** field.



Replacements: Job Title Standardizations

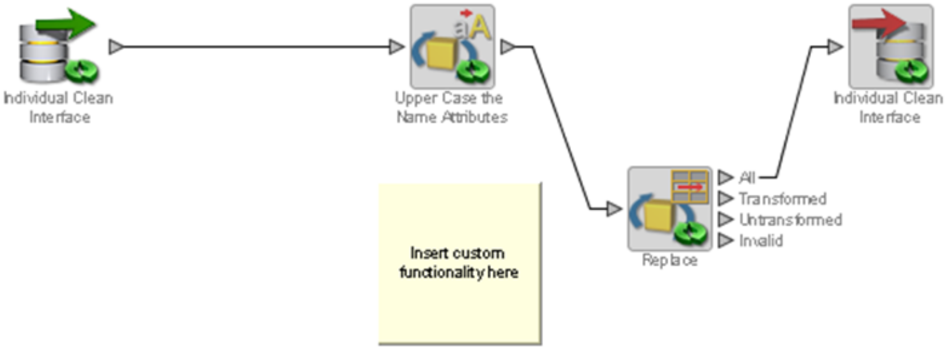
Match longest value? ○ Yes ● No

Ignore case? ● Yes ○ No

Match list by: Whole value

Delimiters:

9. Click **OK** to close the processor configuration dialog.
10. Connect the **All** output of the **Replace** processor to the **Writer**, then click **OK** without making any changes to the **Writer** configuration.
11. On the Process Canvas delete the direct link between the **Upper Case** processor and the **Writer**.


Individual Clean Interface
Upper Case the Name Attributes
Individual Clean Interface
All
Transformed
Untransformed
Invalid
Replace
Insert custom functionality here

12. Close the process and save the changes.
13. Test the modified cleaning service.

4.2.2 Changing Country-Specific Address Cleaning Settings

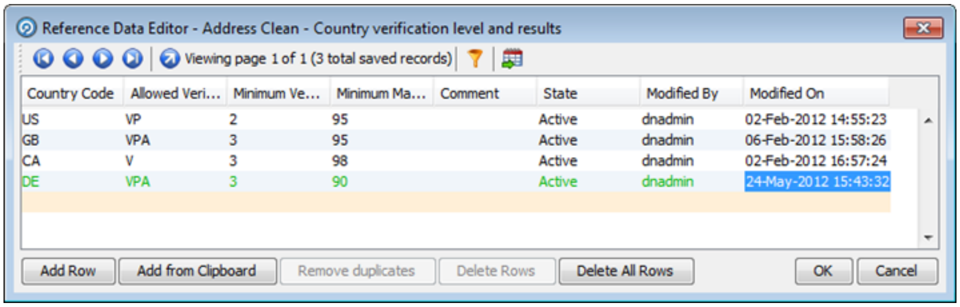
The default settings (Allowed Verification Results, Minimum Verification Level and Minimum Match Score) used in the Address Cleaning process that uses EDQ-AV can be overridden on a per-country basis by simply modifying reference data.

4.2.2.1 Reducing the Strictness of German Address Validation

Modify the EDQ-AV settings to reduce how strictly German addresses will be validated as in the following example steps.

1. Ensure that no jobs are currently running.
2. In the **EDQ-CDS** project edit the **Address Clean - Country verification level and results** Reference Data.
3. Add the following row:

- Country Code: DE
- Allowed Verification Results: VPA
- Minimum Verification Level: 3
- Minimum Match Score: 90



Reference Data Editor - Address Clean - Country verification level and results

Viewing page 1 of 1 (3 total saved records)

Country Code	Allowed Veri...	Minimum Ve...	Minimum Ma...	Comment	State	Modified By	Modified On
US	VP	2	95		Active	dnadmin	02-Feb-2012 14:55:23
GB	VPA	3	95		Active	dnadmin	06-Feb-2012 15:58:26
CA	V	3	98		Active	dnadmin	02-Feb-2012 16:57:24
DE	VPA	3	90		Active	dnadmin	24-May-2012 15:43:32

4. Click **OK** to close the dialog.

4.3 Adjusting Matching

This section explains how you can change the EDQ matching settings.

4.3.1 Changing the Match Clusters To Use During Matching

By default, the clusters that are used during matching depend on the value of the `clusterlevel` setting. All clusters for the specified level and all lower levels are applied. It is possible to customize the system to turn off particular clusters on an individual basis. However, this is only necessary if greater granularity than the three standard cluster levels is required.

The methods for controlling which Match Clusters are used differs for Batch and Real-Time processing. The following sections contain examples to show you how to modify the clusters used.

4.3.1.1 Turning Off Clusters in Individual Batch Matching

Modify match process to turn off clusters during individual batch matching as in the following example steps.

1. Ensure that no jobs are currently running.
2. In the **EDQ-CDS** project, open the **Match - Individual** process.
3. Double-click on the **Individuals - Match** processor to open the processor tab.
4. Select the Cluster icon, and select or unselect the Cluster options as required.

> **Note:** You should always select the **Real-time Cluster** option otherwise real-time matching will no longer operate. The match processors are shared between real-time and batch jobs.

5. Close the tab, and click **Yes** to save the changes.

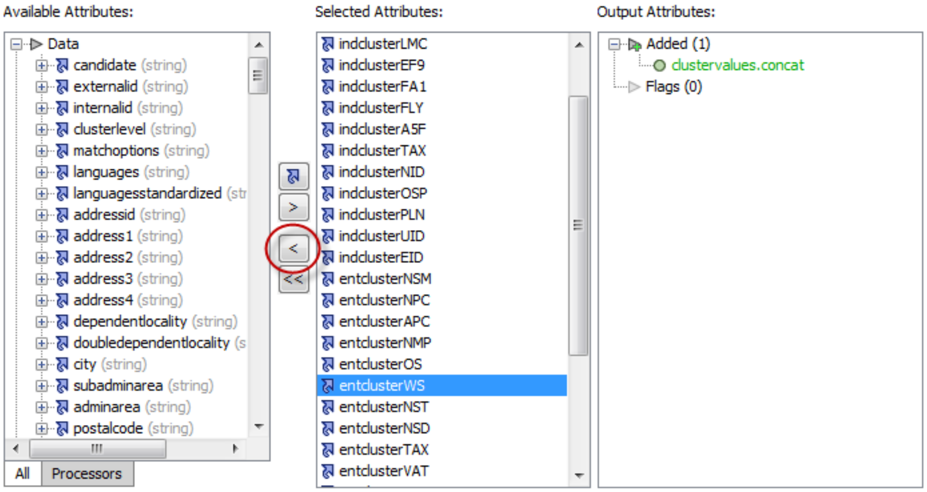
4.3.1.2 Turning Off Clusters in Entity Real-Time Matching

In Real-Time matching, each driving record is compared against every other record in the input set; clustering is performed as a separate, prior call. Therefore, in order to turn off a cluster it must be suppressed at the time of generation.

Modify match process to turn off clusters during entity real-time matching as in the following example steps.

> **Note :** This will only affect new records, unless all cluster keys are re-created.

1. Ensure that no jobs are currently running.
2. In the **EDQ-CDS** project, open the **Cluster Results – Realtime Output** process.
3. Double-click on the **Concatenate All Clusters** processor to open the **Processor Configuration** dialog.
4. Select the Cluster Attributes in the **Selected Attributes** list as appropriate and click on the left-arrow button to remove them. For example, `entclusterWS`, the Website cluster as in the following:



5. Click **OK** to close the dialog.
6. Close the process and save the configuration changes.

4.3.2 Changing Match Rule Enablement

Match rule enablement is externalized in this release. You can override this behavior by adding the `name...address conflict` properties to your `edq-cds.properties` file then editing the values as in the following example:

```
# Disable all entity "name...address conflict" type rules.
phase.*.process.Match\ -\ Entity.[E010V]\ Script\ full\ name\ exact\;\ address\
conflict.entity_match_rules_enabled = false
phase.*.process.Match\ -\ Entity.[E020V]\ Full\ name\ exact\;\ address\
conflict.entity_match_rules_enabled = false
phase.*.process.Match\ -\ Entity.[E030V]\ Standardized\ full\ name\ exact\;\
address\ conflict.entity_match_rules_enabled = false
phase.*.process.Match\ -\ Entity.[E040V]\ Script\ full\ name\ without\ suffixes\
exact\;\ address\ conflict.entity_match_rules_enabled = false
```

```
phase.*.process.Match\ -\ Entity.[E050V]\ Full\ name\ without\ suffixes\ exact\;\
address\ conflict.entity_match_rules_enabled = false
```

Capitalization must be respected and characters must be escaped as required. The asterisk (`*`) character denotes a wildcard, which specifies that the above rule applies to all phases and all processes.

4.3.3 Turning off Unused Match Functionality

The value of the `matchthreshold` setting is used to control the strength of matches that are returned from the Matching services by filtering out results that fall below the specified threshold. Match rules with a priority score below this value are effectively redundant.

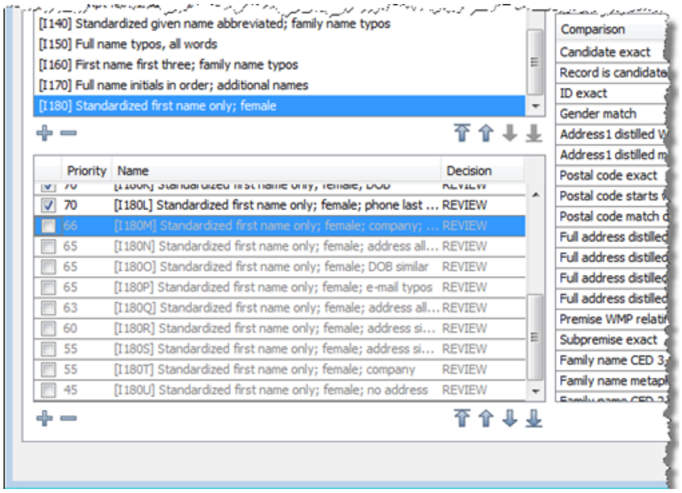
Also, the match processes output a number of additional attributes which are not used in the default configuration and can be removed without loss of functionality. These attributes may be required for use in customizations of EDQ-CDS. For more information, see Section 4.3.3, "Turning off Unused Match Functionality."

4.3.3.1 Disabling Rules with Lower Scores

The `matchthreshold` setting has been configured to have a value of `70`, so all Match rules with a lower priority score will be disabled.

The following example steps show you how to disable Match rules for any Match process (for example, **Match - Individual**, **Match - Entity** or **Match - Address**):

1. Ensure that no jobs are currently running.
2. In the **EDQ-CDS** project, open the Match process.
3. Double-click the Match processor to open the **Match Configuration** tab.
4. Double click the **Match** sub-processor icon to open the **Match Configuration** dialog.
5. Select the **Match Rules** tab and select the last Match group.
6. Clear the check box beside each Match rule with a Match Priority score lower than `70` to disable it.



[I140] Standardized given name abbreviated; family name typos
[I150] Full name typos, all words
[I160] First name first three; family name typos
[I170] Full name initials in order; additional names
[I180] Standardized first name only; female

	Priority	Name	Decision
☑	70	[I180K] Standardized first name only; female; DOB	REVIEW
☑	70	[I180L] Standardized first name only; female; phone last ...	REVIEW
□	66	[I180M] Standardized first name only; female; company; ...	REVIEW
□	65	[I180N] Standardized first name only; female; address all...	REVIEW
□	65	[I180O] Standardized first name only; female; DOB similar	REVIEW
□	65	[I180P] Standardized first name only; female; e-mail typos	REVIEW
□	63	[I180Q] Standardized first name only; female; address all...	REVIEW
□	60	[I180R] Standardized first name only; female; address si...	REVIEW
□	55	[I180S] Standardized first name only; female; address si...	REVIEW
□	55	[I180T] Standardized first name only; female; company	REVIEW
□	45	[I180U] Standardized first name only; female; no address	REVIEW

7. Repeat for each Match group until all rules with a score less than `70` have been disabled.
8. Click **OK** to close the dialog.

9. Close the process and save the configuration changes.

4.3.4 Reviewing Matches in EDQ

The EDQ-CDS Matching services return only those records that matched with a score equal to or greater than the `matchthreshold` setting, and for those records it only returns the record ID, rule name and score. It is useful to be able to view the full record details during rule tuning in order to analyze matches. The Match Review application is a helpful tool in this process.

4.3.4.1 Enabling Match Review in Individual Batch Matching

You can enable match review for individual batch matching as in the following example steps.

1. Ensure that no jobs are currently running.
2. In the **EDQ-CDS** project, open the **Match - Individual** process.
3. Double-click on the **Match Individuals** processor to open the **Match Configuration** dialog.
4. Click **Advanced Options**.
5. From the Review System list, select **Match Review**, and then click **OK**. This makes the Assign Relationship Review option active.
6. Click **Assign Relationship Review**.
7. In the dialog displayed, select the appropriate user or user group in the **Assigned To** drop-down field.
8. Click **OK** to close the dialog.
9. Close the process and save the configuration changes.
10. Open the **Batch Individual Match** job.
11. Locate the Match phase, right-click on the **Match Prepare** task and select **Configure**. The **Task Configuration** dialog opens.
12. Select the **Process** tab, and check the **Enable Sort/Filter in Match**? option.
13. Click **OK** and close the job, saving changes when prompted.
14. Run the job from Director with the appropriate run profile and no run label to regenerate the data.

> **Note:** In order to generate Match Review data, you must run jobs without a run label.

Matches can be reviewed as follows:

1. On the Launchpad page, click **Match Review** icon.

> **Note :** If this application is not visible then you will need to publish it via the launchpad server configuration pages.

2. Login as a user with the appropriate security permissions (for example, a user that is a member of the group selected in step 5).
3. Select **Match - Individual** in the **Reviews** list in the left-hand panel to view the Match Review statistics.
4. Click the **Launch Review Application** link to start reviewing matches for the selected Review.

4.3.4.2 Changing the Decision Key

The Decision Key consists of a set of input attributes that are used in a hashing algorithm to re-apply (that is, 'remember') manual match decisions. This means that any manual match decisions made on a pair of records will be re-applied on subsequent runs of the matching process as long as the data values in the attributes that make up the decision key remain the same.

So, for example, if matching individuals using name and address details, and one of the manually matched records changes, you may want to reappraise the records rather than apply a manual decision that was made based on different data. However, if the value in another attribute changes, you may consider there to be no real change to the details of the record used in matching. For example, a Balance attribute containing a numerical amount might be input to a matching process as it may be used in the output selection logic, but a change to the attribute value should not cause a reappraisal of the decision to match, or not match the record against another.

By default, all attributes that have been mapped to identifiers are included in the Decision Key (unless the match processor has been upgraded from a previous version - see note below). However, you can change the Decision Key to use all the attributes input into a match processor, or customize the key by selecting exactly which attributes make up the key. For example, if you want always to re-apply match decisions as long as the records involved are the same, even if the data of those records changes, you can select only the primary key attributes of records in each source involved in matching.

4.3.4.3 Changing the Decision Key after Decisions Have Been Made

In general, you should decide how to configure the Decision Key before making a matching process operational and assigning its results for review. However, if decisions have already been made when the construction of the Decision Key changes, EDQ will make its best effort to retain those decisions within the following limitation:

If an attribute that was formerly used in a Decision Key is no longer input to match processor, it will not be possible to reapply any decisions that were made using that key

This means that adding attributes to a Decision Key can always be done without losing any previous decisions, providing each decision is unique based on the configured key columns.

Note that it is still possible to remove an attribute from a Decision Key and migrate previous decisions, by removing it from the Decision Key in this tab but keeping the input attribute in the match processor for at least one complete run with the same set of data as run previously. Once this has been done it will be safe to remove the attribute from the matching process.

4.4 Modifying Reference Data Used in Matching

This section explains how you can modify your data to improve matching and provides examples to aid you.

4.4.1 Stripping Words/Phrases from Name Fields

It is possible to customize the system to strip certain words and phrases from names that are deemed to be noise and/or add little information, and therefore may lead to potential missed matches.

4.4.1.1 Removing Noise from Individual Names

Name fields in customer data systems are often overfilled with additional (non-name) information, either because there are no other suitable fields available or due to errors made by Data Entry users. Common examples include "Fred SMITH (DO NOT CALL)" and "John DOE (DECEASED)". This extraneous information can be removed during name standardization when a "distilled" name is created for use in matching.

Use the following example steps to remove noise from individual names:

1. Ensure that no jobs are currently running.
2. In the **EDQ-CDS - Initialize Reference Data** project open the **Strip List – Titles Latin** Reference Data.
3. Add the following rows to the Reference Data set:
   - DO NOT CALL
   - DECEASED
4. Click **OK** to close the dialog.
5. Re-run the **MAIN Initialize Reference Data** job from the Server Console to re-prepare the Reference Data files that are used by the Matching services.

> **Note :** The Real-Time services will use the modified Reference Data sets the next time the full **Real-time START ALL** job (which re-snapshots the prepared Reference Data from files) is run.

To remove words and phrases from individual names in non-Latin scripts use the reference data **Strip List – Individual Script Strip List** Reference Data . This Reference Data set is used as a replacement map and should have a blank value in the second column.

4.4.1.2 Removing Noise from Entity Names

Noise words and phrases or common business words (including suffixes) in Entity names that add little value in matching can be removed during name standardization when a "distilled" name is created. An example of such a noise word is "International", which is often found in organization name fields.

Due to the high frequency of occurrence of this term it is often omitted or shortened when entering the name, which may lead to potential matches being missed. Therefore it may be more appropriate to remove the term and all known variants for the purposes of matching.

Use the following example steps to remove noise from entity names:

1. Ensure that no jobs are currently running.

2. In the **EDQ-CDS - Initialize Reference Data** project open the **Strip List – Entity Latin** Reference Data.
3. Add the following rows to the Reference Data set:
   - INTERNTL
   - INTL
   - INT
4. Click **OK** to close the dialog.
5. Re-run the **MAIN Initialize Reference Data** job from the Server Console to prepare the data.

To remove words and phrases from entity names in non-Latin scripts use the **Strip List – Entity Script Suffixes** Reference Data.

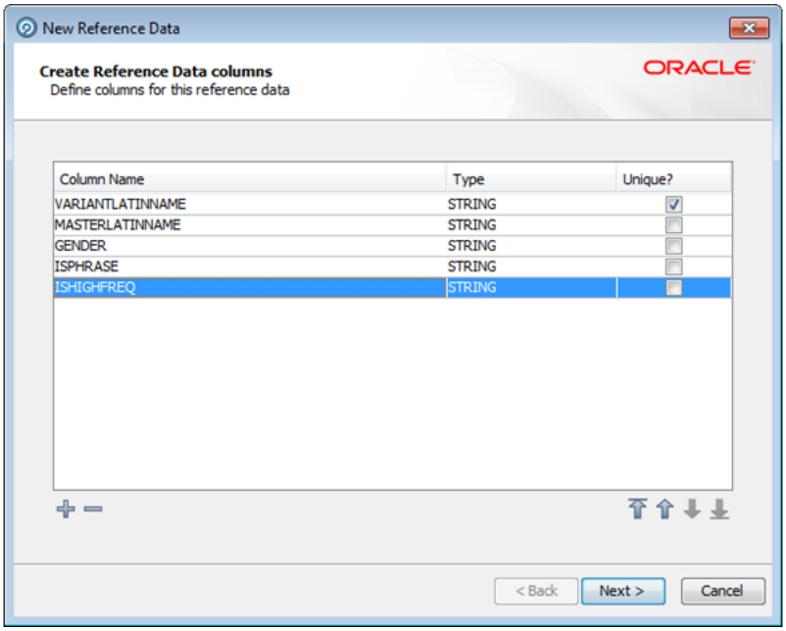
4.4.2 Changing Name Standardization

EDQ-CDS uses a name standardization technique in order to match name variants. It is supplied with a large collection of common name variants for various language domains. It is possible to customize these lists.

> **Note :** If a name standardization is changed or added, the subsequent results may be eliminated during Conflict Resolution. For further details, see Section 4.4.3, "Resolving Conflicts".

4.4.2.1 Adding Individual Name Standardizations

1. Ensure that no jobs are currently running.
2. In the **EDQ-CDS - Initialize Reference Data** project create a new Reference Data set with columns as in the following:


New Reference Data
Create Reference Data columns
Define columns for this reference data
ORACLE
< Back
Next >
Cancel

Column Name	Type	Unique?
VARIANTLATINNAME	STRING	☑
MASTERLATINNAME	STRING	☐
GENDER	STRING	☐
ISPHRASE	STRING	☐
ISHIGHFREQ	STRING	☐

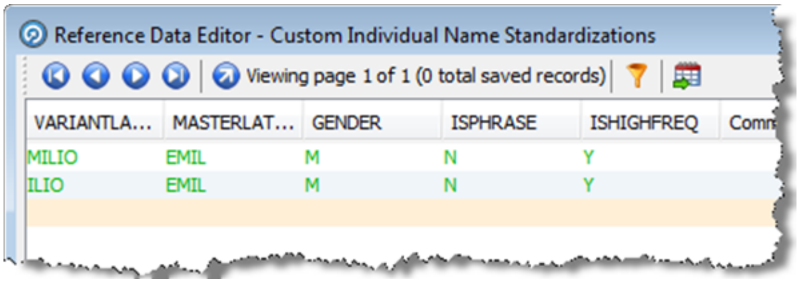
3. Click **Next** through the **New Reference Data** wizard and name it **Custom Individual Name Standardizations**.

4. Click **Finish** to close the dialog.

5. The **Reference Data Editor** dialog will open. Add the required name standardizations, where:

   - VARIANTLATINNAME is the name to be standardized.
   - MASTERLATINNAME is the standardized version of variant name.
   - GENDER takes the value M for male, F for Female, or U for unknown or ambiguous.
   - ISPHRASE takes the value N for single token names and Y for multi-token names containing whitespace.
   - ISHIGHFREQ is set to Y.

   Note : It is important to ensure that data is entered in upper case and that variant names only have a single master across all language domains.



Reference Data Editor - Custom Individual Name Standardizations

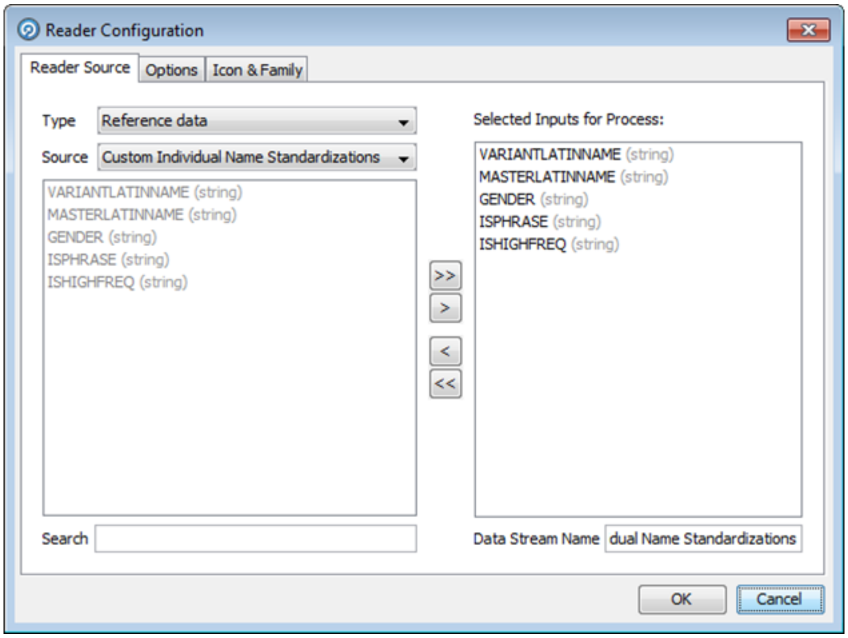
Viewing page 1 of 1 (0 total saved records)

VARIANTLA...	MASTERLAT...	GENDER	ISPHRASE	ISHIGHFREQ	Comn
MILIO	EMIL	M	N	Y	
ILIO	EMIL	M	N	Y	

6. Click **OK** to close the dialog.

7. Open the **[D] Initialise Individual Latin to Latin Data** process.

8. Add a **Reader** process to the Process Canvas and configure it to use the **Custom Individual Name Standardizations** Reference Data as the source, selecting all attributes for input to the process.



Reader Configuration

Type: Reference data

Source: Custom Individual Name Standardizations

VARIANTLATINNAME (string)
MASTERLATINNAME (string)
GENDER (string)
ISPHRASE (string)
ISHIGHFREQ (string)

Search:

Selected Inputs for Process:

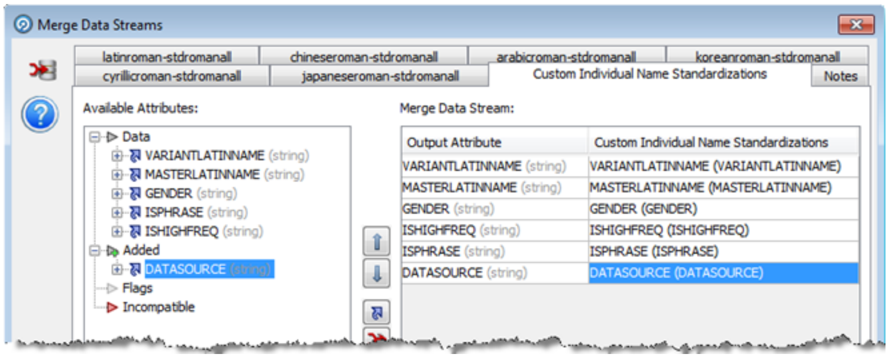
VARIANTLATINNAME (string)
MASTERLATINNAME (string)
GENDER (string)
ISPHRASE (string)
ISHIGHFREQ (string)

Data Stream Name: dual Name Standardizations

9. Add a new **Add String Attribute** processor to the process canvas and connect the reader to the new processor. In the processor configuration dialog rename the new attribute DATASOURCE and set the attribute value to CUSTOM.

10. Connect the output of the **Add String Attribute** processor to the **Merge Data Streams** processor.

11. In the **Custom Individual Name Standardizations** tab of the **Processor Configuration** dialog associate the Available Attributes with the Output Attributes in the **Merged Data Stream** area:



Output Attribute	Custom Individual Name Standardizations
VARIANTLATINNAME (string)	VARIANTLATINNAME (VARIANTLATINNAME)
MASTERLATINNAME (string)	MASTERLATINNAME (MASTERLATINNAME)
GENDER (string)	GENDER (GENDER)
ISHIGHFREQ (string)	ISHIGHFREQ (ISHIGHFREQ)
ISPHRASE (string)	ISPHRASE (ISPHRASE)
DATASOURCE (string)	DATASOURCE (DATASOURCE)

12. Click **OK** to close the dialog.

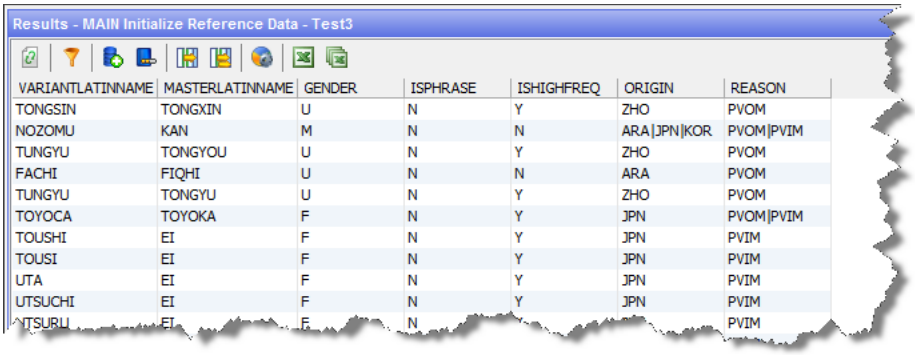
13. Close the process and save the configuration changes.

14. Re-run the **MAIN Initialize Reference Data** job from the Server Console to prepare the data.

4.4.3 Resolving Conflicts

Conflict resolution is performed to resolve issues arising when name standardization rules try to standardize names to more than one Master name. For example, if there is a rule that maps "Jon" to a Master of "John" and another that maps "Jon" to "John-Boy", there is a conflict. This conflict is resolved by assessing the importance of each Master name in the given standardization data. The best candidate is then selected as the primary Master, and other standardization maps conflicting with it are removed and quarantined.

As part of conflict resolution, each removed record is assigned one or more Reason Codes explaining why it is in conflict. These codes are displayed in the **REASON** column in the Server Console **Results** window:



VARIANTLATINNAME	MASTERLATINNAME	GENDER	ISPHRASE	ISHIGHFREQ	ORIGIN	REASON
TONGSIN	TONGXIN	U	N	Y	ZHO	PVOM
NOZOMU	KAN	M	N	N	ARA|JPN|KOR	PVOM|PVIM
TUNGYU	TONGYOU	U	N	Y	ZHO	PVOM
FACHI	FIQHI	U	N	N	ARA	PVOM
TUNGYU	TONGYU	U	N	Y	ZHO	PVOM
TOYOCA	TOYOKA	F	N	Y	JPN	PVOM|PVIM
TOUSHI	EI	F	N	Y	JPN	PVIM
TOUSI	EI	F	N	Y	JPN	PVIM
UTA	EI	F	N	Y	JPN	PVIM
UTSUCHI	EI	F	N	Y	JPN	PVIM
TSURU	EI	F	N			PVIM

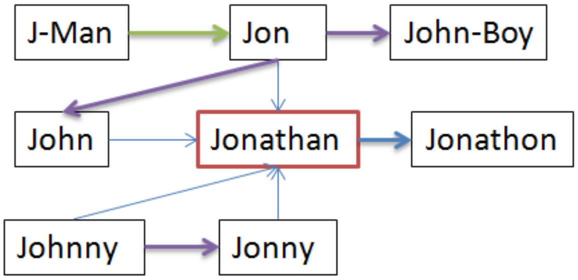
The Reason Codes are as follows:

- **PIV**: The Primary record of a cluster of records (for example, the best Master identified for a set of equivalences) is also present as a variant to other Masters. All the instances where this Primary name is a variant are removed.
- **PVOM**: The records that are variants of the current Primary are also variants of other Masters. All the records for these variants pointing to other Masters are removed.
- **PVIM**: The records that are variants of the current Primary are also Masters to other variants. All the records where this variant is a Master are removed.
- **PIVCUTOFF**: Whereas the other removals take place after identification of Primary clusters, there comes a time where it is not efficient to continue to identify the Primaries, and the remaining records where the Master name also exists as a variant have all the variant versions removed in a final cull of records that violate integrity.

Expanding on the simple example given at the beginning of this section, let us assume that there are the following name standardization rules:

Master	Primary
J-MAN	JON
JOHN	JONATHAN
JOHNNY	JONNY
JON	JOHN
JON	JONATHAN
JON	JOHN-BOY
JONNY	JONATHAN
JONATHAN	JONATHON
JOHNNY	JONATHAN

These rules contain a number of inherent conflicts. This is illustrated in the following diagram in which JONATHAN is identified as the Primary:

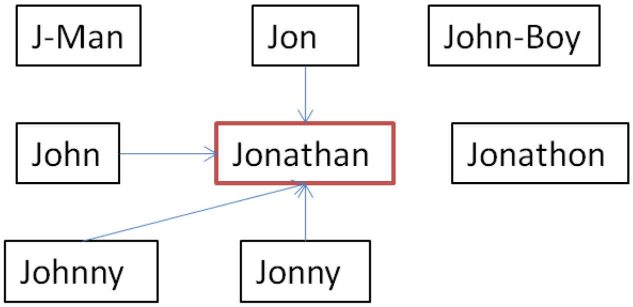


The arrows indicate the following:

Arrow Type	Reason for Conflict
→	N/A (No conflict exists)
➜	PIV

Arrow Type	Reason for Conflict
→	PVIM
→	PVOM

The conflict resolution rules will discard the mappings that cause conflicts, as follows:



Resulting in the following mappings being created:

Name	Primary
JOHN	JONATHAN
JON	JONATHAN
JONNY	JONATHAN
JOHNNY	JONATHAN

5

Installing and Using Data Quality Health Check

This chapter describes how you can use the EDQ-CDS Data Quality Health Check functionality.

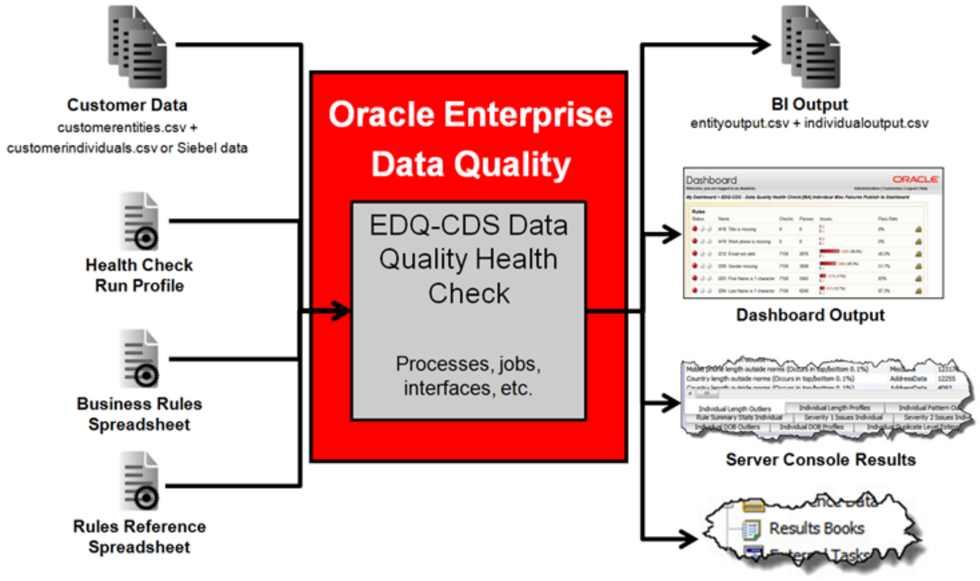
This chapter includes the following sections:

Data Quality Health Check extends the capability of the EDQ-CDS, allowing you to perform batch data quality checking of your data before it has been normalized or standardized. The results can then be viewed in the Server Console, a Business Intelligence (BI) tool, in the EDQ Results books, or published to the Dashboard as required. As a component of EDQ-CDS, Data Quality Health Check can be integrated with Siebel or used in stand-alone mode.

EDQ-CDS Data Quality Health Check will primarily be of use to anyone requiring a view of the quality of raw data, from Data Stewards who require a data-level view of data quality issues, to Operations Analysts and Executives who require Dashboard information for analysis, reporting and planning purposes. Additionally, it is useful for Data Professionals that want to analyze the technical aspects of data, and to EDQ-CDS users seeking to ensure their CDS processes are performing efficient deduplication.

5.1 Architecture

The following illustrates how you can use EDQ-CDS Data Quality Health Check to process your data and view the results:



5.1.1 Multiple Child Entities

Some data will feature multiple child entities, for example, more than one address might be assigned to each record. When such records are processed and passed to EDQ, one record per child is created.

Therefore, the Data Quality Health Check results often list a greater number of records than are initially taken in. It is important to remember this when viewing results in Server Console or Dashboard.

5.2 Installing Data Quality Health Check

The section explains how to install Data Quality Health Check. While Data Quality Health Check is part of the EDQ-CDS distribution, it is not necessary to fully configure EDQ-CDS in order to use Data Quality Health Check. EDQ Health Check has the same prerequisites as the EDQ-CDS.

Siebel integrations require the installation of the Siebel Connector Release 11.1.1.7.3 or later. For more information, see *Oracle Enterprise Data Quality Customer Data Services Pack Siebel Integration Guide*.

5.2.1 Installation Components

The components necessary to install Data Quality Health Check are all contained within the EDQ-CDS distribution, and are therefore installed by unzipping the `config.zip` folder in the distribution over the `oedq.local.home` folder of the EDQ installation.

The EDQ-CDS Health Check components are:

- `edq-cds-data-quality-health-check-n.n.n.(nnn).dxi` - the packaged EDQ project containing the EDQ-CDS data quality services.
- `dq-health-check-business-rules-individual.xls` - Individual Business Rules spreadsheet, which defines the data quality checks performed for individuals.
- `dq-health-check-business-rules-entity.xls` - Entity Business Rules spreadsheet, which defines the data quality checks performed for entities.

- `edq-cds-data-quality-health-check.properties` - the default Run Profile.
- `customerentities.csv` - Sample Entity data.
- `customerindividuals.csv` - Sample Individual data
- `rulesreference.xls` - Spreadsheet categorizing the error codes present in the Business Rules spreadsheets.

5.2.2 Installing the Software

If you have installed EDQ-CDS, then Data Quality Health Check is installed and no further installation tasks are necessary.

To install Data Quality Health Check without the presence of EDQ-CDS, use the following procedure.:

> **Note:** If the EDQ-CDS server uses a different landing area path from that set during installation (by default, `oedq.local.home/landingarea`), the `landingarea` folder created when the `config.zip` is extracted must be copied over the existing `landingarea` folder.

1. Extract the `config.zip` file over the `oedq.local.home` folder of the EDQ installation.
2. Restart the EDQ Server.
3. Start the EDQ Director client, and log on as a user with the permission to create projects (Administrator or Project Owner).
4. Open the `edq-cds-data-quality-health-check-n.n.n.(nnn).dxi` package file by either:
   - selecting **Open Package File...** on the **File** menu and browsing to the `.dxi` file;
   - right-clicking on an empty part of the Project Browser, selecting **Open Package File...**, and browsing to the `.dxi` file; or
   - dragging and dropping the file onto the Project Browser.
5. Drag the whole **EDQ-CDS - Data Quality Health Check** project onto the **Projects** node.
6. Right-click on the `.dxi` file, and select **Close Package File**.

5.2.3 Verifying the Installation

Data Quality Health Check comes with two sample `.csv` files in the `landingarea/dqhealthcheck` folder. These files can be used to test the installation is working correctly.

The sample files are:

- `customerentities.csv` - Sample Entity data.
- `customerindividuals.csv` - Sample Individual data.

The default jobs provided with Data Quality Health Check are configured to run against these files.

To verify the installation, run either (or both) of the **Run Entity Data Quality Health Check** or **Run Individual Data Quality Health Check** jobs in Server Console, remembering to select the edq-cds-data-quality-health-check.properties Run Profile.

Note:

Data Quality Health Check uses its own internal reference data, and therefore does not need the CDS Initialize project to be run before it is used.

Do not attempt to run any of the Siebel jobs manually; these jobs are designed to be invoked automatically by the Siebel Connector.

Check the Event Log and Results in Server Console to ascertain whether the job (or jobs) have completed correctly. If so, then the installation has been successful.

Finally, purge the results of the job or jobs in the Server Console and Dashboard:

- **Server Console**:

  Select the **Results** view, right click the job in the **Job History** area, select the **Purge data for run label [Name of Run Label]** option.

- **Dashboard**:

  Open Dashboard Administration, expand the **Audit** tree in the **Audits & Indexes** area, right click on the **Data Quality Health Check** audit and select **Purge**.

5.3 Configuring Data Quality Health Check

This section explains how to configure Data Quality Health Check.

5.3.1 Configuring Business Rules

The Business Rules are set in two .xls files supplied with Data Quality Health Check, located in the oedq_local_home/business rules folder.

- dq-health-check-business-rules-individual.xls - Individual Business Rules spreadsheet.
- dq-health-check-business-rules-entity.xls - Entity Business Rules spreadsheet.

There is an additional spreadsheet - rulesreference.xls - in the oedq_local_home/landingarea/dqhealthcheck folder which has two main functions: it is used to control which rules in the Business Rules spreadsheets are used when running Data Quality processes, and also to construct rules statistics.

Note : By default, the Individual and Entity rules that are used by EDQ-AV are enabled in the rulesreference.xls spreadsheet. If EDQ-AV is not installed these rules must be disabled to prevent inaccurate reporting in the Dashboard.

The **Enabled** column in the rulesreference.xls spreadsheet controls which rules are enabled and which are disabled, the two possible values being yes and no. Therefore, if any existing rules are edited or new rules added to the Business Rules spreadsheets, the changes must be reflected in the rulesreference.xls sheet. Any changes made must preserve the separation of rule types, which object (Individual or Entity) they relate to, and their associated rule and error codes.

The rules fall into the following categories:

- Population checks – Check that a field is not blank. For example, `ER205 - Check if Name is missing`.
- List checks - Check that the data contains only values from a specified list. For example, `IR202 - Check if Upper Case gender is a valid value`.
- Length checks - Check that the data is of a specified length,. For example, `IR203 - Check first name is > 1 char`.
- Format checks - Check that the data conforms to a pattern or regular expression. For example, `IR212 - Check if email is valid format`.
- Contains checks - Check that the data contains a value from a list; for example `IR428 - Check if full name is clear of entity hints`.
- Suspect data checks - Check that the data exhibits any common data entry "cheats". For example, `ER411 - Check if unusual characters in name`.
- Value checks – Check that the field value is in the correct range. For example, `IR430 - Check if DOB is very old (<1900)`.
- Dependent attribute checks – Check that two attribute values are consistent, for example, if the value in one attribute is dependent on the value in another attribute. For example, `IR302 - Check if gender and title are consistent`.
- Duplicate checks - Compare combinations of data attributes to estimate potential levels of record duplication. This is not full EDQ-CDS matching, and therefore is designed to run in a fraction of the time. Examples of comparisons include:
  - `IR401 - Check if fname address1 are flagged dupe`
  - `IR403 - Check if fname email are flagged dupe`
  - `IR408 - Check if lname tax no are flagged dupe`

For information on customizing existing and creating new Business Rules using these spreadsheets, see the "Defining Business Rules" topic in the *Oracle Enterprise Data Quality Director Online Help*.

5.3.2 Configuring the Run Profile

The `edq-cds-data-quality-health-check.properties` Run Profile is divided into the following sections:

5.3.2.1 Publish to Dashboard Setting

This setting controls whether the results of the Health Check jobs are published to the Dashboard:

```
phase.Publish\ to\ Dashboard.enabled = yes
```

The default value is `yes`. Change to `no` to prevent the results being published.

Note : The value must always be in lower case, `yes` or `no`.

5.3.2.2 Source Input File Encoding

These settings specify the source of the input files for individual and entity data, the field separator used, and the encoding employed. The default settings are included as in the following:

```
phase.*.snapshot.*.Entity_Input_CSV_File_Location =
\\dqhealthcheck\\customerentities.csv
phase.*.snapshot.*.Entity_Input_CSV_File_Field_Separator = \,
phase.*.snapshot.*.Entity_Input_CSV_File_Encoding = UTF-8
phase.*.snapshot.*.Individual_Input_CSV_File_Location =
\\dqhealthcheck\\customerindividuals.csv
phase.*.snapshot.*.Individual_Input_CSV_File_Field_Separator = \,
phase.*.snapshot.*.Individual_Input_CSV_File_Encoding = UTF-8
```

The file and folder location specified *must* be in the `landingarea` folder.

The encoding of the input file must be a valid encoding for EDQ delimited text Data Stores. The escape character - backslash "\" - must be used if the desired separator is a reserved character, for example, a comma. A list of valid encoding formats can be found in the **Edit Data Store** dialog in EDQ.

5.3.2.3 Publish Results as CSV Setting

This setting controls whether the results of the Health Check jobs are published in the form of a `.csv` file for use in a BI tool:

```
phase.Export\ BI\ Data.enabled = no
```

The default value is `no`. Set to `yes` to publish the data to the `.csv` file.

Note : The value must always be in lower case, `yes` or `no`.

5.3.2.4 Export Check Results

If export is enabled, these settings specify the destination of the exported file, the field separator and encoding. The default settings are included as in the following:

```
phase.*.Export.*.Entity_Output_CSV_File_Location =
\\dqhealthcheck\\entityoutput.csv
phase.*.Export.*.Entity_Output_CSV_File_Field_Separator = \,
phase.*.Export.*.Entity_Output_CSV_File_Encoding = UTF-8
phase.*.Export.*.Individual_Output_CSV_File_Location =
\\dqhealthcheck\\individualoutput.csv
phase.*.Export.*.Individual_Output_CSV_File_Field_Separator = \,
phase.*.Export.*.Individual_Output_CSV_File_Encoding = UTF-8
```

Note : The encoding of the export file must be valid for EDQ delimited text Data Stores. A list of valid encoding formats can be found in the **Edit Data Store** dialog in EDQ.

5.3.2.5 Address Verification Country Code

If EDQ-AV is installed, this setting should be assigned the ISO two-character country code to be used by default. For example, if the country code is not specified in the data supplied:

```
phase.*.process.*.Default\ AV\ Country\ Code
```

The default value is US. Any codes that are entered here are expected to comply with the ISO-3166-1-alpha-2 specification.

5.3.2.6 Results Book Settings

To create EDQ Results Books populated with Individual and/or Entity profiling data, uncomment the following settings.

Note : The first six lines are for the Individual Profiling Results book, and the last two are for the Individual Rules Results book. It is possible to populate one or both of these books as required.

For Individual data, these settings will populate the Individual Profiling Results Book with drillable results of all profilers and the Individual Rules Results Book with a drillable view of rule failures.

```
phase.Profile\ Individual\ Misc\ Data.enabled = no
phase.Profile\ Individual\ Misc\ Data\ With\ Results\ Book.enabled = yes
phase.Profile\ Individual\ Address\ Data.enabled = no
phase.Profile\ Individual\ Address\ Data\ With\ Results\ Book.enabled = yes
phase.Profile\ Individual\ Alt\ Phone Data.enabled = no
phase.Profile\ Individual\ Alt\ Phone\ Data\ With\ Results\ Book.enabled = yes
phase.Process\ Rule\ Failures\ to\ Outputs.enabled = no
phase.Process\ Rule\ Failures\ to\ Outputs\ With\ Results\ Book.enabled = yes
```

For Entity data, these settings will populate the Entity Profiling Results Book with drillable results of all profilers and the Entity Rules Results Book with a drillable view of the rule failures:

```
phase.Profile\ Entity\ Misc\ Data.enabled = no
phase.Profile\ Entity\ Misc\ Data\ With\ Results\ Book.enabled = yes
phase.Profile\ Entity\ Address\ Data.enabled = no
phase.Profile\ Entity\ Address\ Data\ With\ Results\ Book.enabled = yes
phase.Profile\ Entity\ Alt\ Phone\ Data.enabled = no
phase.Profile\ Entity\ Alt\ Phone\ Data\ With\ Results\ Book.enabled = yes
phase.Make\ Analysis\ and\ Server\ Console\ Output.enabled = no
phase.Make\ Analysis\ and\ Server\ Console\ Output\ With\ Results\ Book.enabled =
yes
```

5.3.2.7 Staged Data Visibility Settings Within Server Console

These settings control which Staged Data items are visible in Server Console.

The first setting - `stageddata.*.visible = no` - makes all Staged Data items invisible by default. The remaining settings then make specific Staged Data items visible.

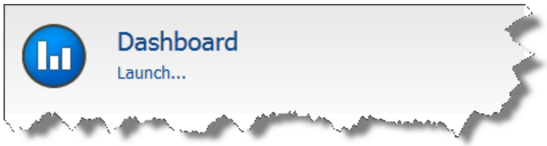
By default, detailed data in the DQ Health Check Analysis Output tab in the Server Console Results screen is hidden. This is because the level of detail is seldom required for most purposes. To view this data, set the following properties in the Run Profile to `yes`:

- `stageddata.Individual\ DQ\ Health\ Check\ Analysis\ Output.visible =`
- `stageddata.Entity\ DQ\ Health\ Check\ Analysis\ Output.visible =`

5.4 Configuring the Dashboard

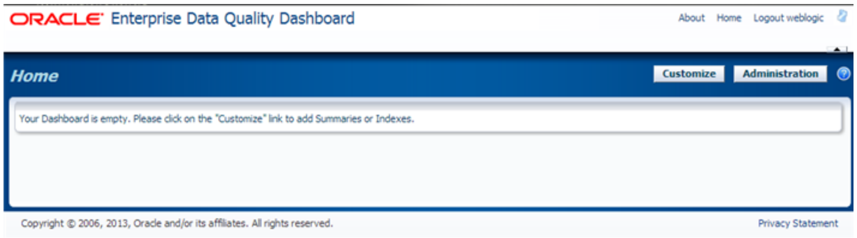
By default, the Health Check results are published to the Dashboard.

The Dashboard is accessed from the EDQ Launchpad:



To configure Health Check results on the Dashboard, use the following procedure:

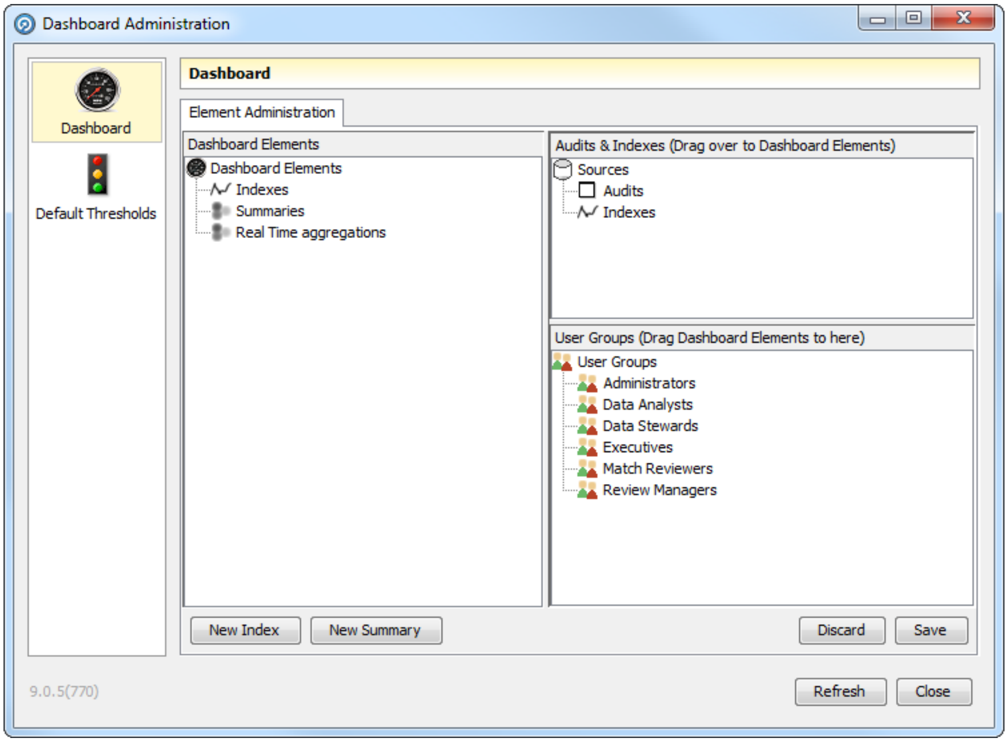
1. Open the Dashboard.
2. On the main Dashboard, click **Administration**.



Your Dashboard is empty. Please click on the "Customize" link to add Summaries or Indexes.

Copyright © 2006, 2013, Oracle and/or its affiliates. All rights reserved.

The Dashboard Administration is displayed:

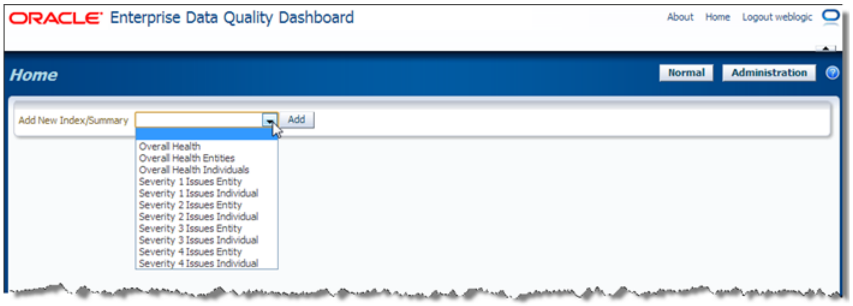


3. Create the Summaries and Indexes as required.

> **Note :** Any rules added to the Summaries should correspond with those enabled in the `rulesreference.xls` spreadsheet. If a disabled rule is included in a Summary or Index it will always be red-flagged, regardless of the results of enabled rules.

4. Return to the Dashboard and click **Customize**.

5. Select the Data Quality results to view in the **Add New** drop-down field. For example:



6. Click **Add**. The selected item is added to the Home view.

Once this configuration procedure is complete, it is possible to choose which Summaries and Indexes to add to the Initial view, to drill down into the results. For full details of how to do this, see *Oracle Enterprise Data Quality Dashboard Online Help*.

5.4.1 Example: Dashboard By Severity

This is an example of a Dashboard configuration that groups rules into Summaries by severity, and then into Indexes.

The first letter of the Health Check rule audit codes indicates the record type ("I" for Individual and "E" for Entity), and the first number indicates the severity level (1, 2, 3 or 4). For example, code E203 is an Entity rule with a severity level of 2.

Create eight summaries to contain the Individual and Entity rule results for severity levels 1 to 4:

- Severity 1 Issues Individual
- Severity 2 Issues Individual
- Severity 3 Issues Individual
- Severity 4 Issues Individual
- Severity 1 Issues Entity
- Severity 2 Issues Entity
- Severity 3 Issues Entity
- Severity 4 Issues Entity

Then create the following Indexes:

Name	Contents
Overall Health Individuals	Contains all the Individual Summaries.
Overall Health Entities	Contains all the Entity Summaries.
Overall Health	Contains the Individual and Entity Summaries.

5.4.1.1 Creating the Summaries

1. Open EDQ Dashboard, and click **Administration** to open the **Dashboard Administration** window.

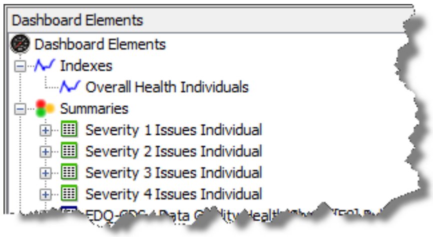
2. Click **New Summary**.

3. Enter **Severity 1 Issues Individual** in the **Add Summary** pop-up.
4. Click **OK**. The new Summary is displayed in the **Summaries** node of the **Dashboard Elements** area.
5. In the **Audits and Indexes** area, expand the **Audits** branch, then expand the **EDQ-CDS – Data Quality Health Check/[I8A] Individual Misc Failures Publish to Dashboard** branch.
6. Click and drag **I101** and **I102** from the **[I8A] Individual Misc Failures Publish to Dashboard** audits list to the **Severity 1 Issues Individual** Summary.
7. Click and drag the **Severity 1 Issues Individual** Summary to the **Administrators** node in the **User Group** area.
8. Click **Save**.
9. Repeat for the remaining summaries.

5.4.1.2 Creating the Indexes

This example assumes all the Summaries detailed in the previous sections have been configured.

1. Open EDQ Dashboard, and click **Administration** to open the **Dashboard Administration** window.
2. Click **New Index**.
3. Name the new Index "Overall Health Individuals".



Dashboard Elements

- Dashboard Elements
  - Indexes
    - Overall Health Individuals
  - Summaries
    - Severity 1 Issues Individual
    - Severity 2 Issues Individual
    - Severity 3 Issues Individual
    - Severity 4 Issues Individual

4. Click and drag the following Summaries into the new Index:
   - Severity 1 Issues Individual
   - Severity 2 Issues Individual
   - Severity 3 Issues Individual
   - Severity 4 Issues Individual
5. Click **Save**.
6. Repeat for the remaining Indexes.

5.4.2 Example - Dashboard By Business Function

This is an example of a Dashboard configuration that groups rules into Summaries by Business Function.

1. Create the following Summaries:

Name	Contents
Account	■ Name Details ■ Identifiers ■ Identifier outliers ■ Address details ■ Address detail outliers ■ Potential duplicates
Contact	■ Name details ■ Identifiers ■ Identifier outliers ■ Address details ■ Address detail outliers ■ Potential duplicates

The rules to be included in each Summary are detailed in Section 5.8, "Dashboard Example Summaries." Ensure that all these rules are enabled.

2. Create the following Indexes:

Name	Contents
Overall Health Account	Containing all the Account-based Summaries.
Overall Health Contacts	Containing all the Contact-based Summaries.
Overall System Health	Containing all the Summaries you created.

5.5 Running Health Check Jobs and Viewing Results

This section describes how to run Health Check jobs and view the results.

5.5.1 Running a Health Check

Health Check jobs can be run either from Siebel, in stand-alone mode from Server Console, or in EDQ-CDS.

If running from Server Console, it may be necessary to prepare the data first.

There are six Health Check jobs:

- Perform Entity Technical Analysis
- Perform Individual Technical Analysis
- Run Entity Data Quality Health Check
- Run Individual Data Quality Health Check
- Siebel Batch Account Health Check
- Siebel Batch Contact Health Check

5.5.1.1 Using the Siebel-Attached Mode

Before Health Check can be used with Siebel, the Siebel Connector must be installed and Siebel must be configured accordingly. For more information, see *Oracle Fusion*

Middleware Installing and Customizing Enterprise Data Quality Customer Data Services Pack.

To run a Health Check job in Siebel, open Server Manager and access the Data Quality Manager component. The two jobs that should be run from Siebel are:

- Siebel Batch Account Health Check
- Siebel Batch Contact Health Check

> **Note :** The other Health Check jobs should not be configured to run from Siebel. It is possible to do this, but they will not return any results. They must always be run from Server Console or EDQ.

Additionally, any settings changed in the Run Profile must also be changed in the `dnd.properties` file to ensure that the changes are accurately reflected in a Siebel batch run.

5.5.1.2 Using the Stand-Alone Mode

The Technical Analysis and Run Entity/Individual Quality Health Check jobs are designed to be run from EDQ or Server Console.

If the data to be checked can be provided in exactly the same format as the sample data files (for example, `.csv` files with column headings as described in Section 5.7, "Understanding Data Interfaces"), simply save these files to the `landingarea\dqhealthcheck` folder using the same file names as (overwriting) the sample data files.

However, if the data is provided in a different format EDQ should be configured to use this data by mapping the available fields to the Health Check input interface. To do this, use the following procedure:

1. Open Director.
2. Create a new Data Store that points at the data.
3. Create a new snapshot using this Data Store as the source.
4. Add and configure a new mapping to the relevant Data Interface (**Entity Data** or **Individual Data**).
5. Edit the relevant job (**Run Entity** or **Run Individual Data Quality Health Check**), adding the new Snapshot and selecting the new Data Interface mapping.

For full details on how to prepare data, see the following topics in the *Oracle Enterprise Data Quality Director Online Help*:

- "Connecting to a Data Store"
- "Adding a Snapshot "
- "Managing Data Interfaces"
- "Running Jobs using Data Interfaces"

5.5.2 Viewing Data Quality Health Check Results

Health Check results can be produced as four output types:

- Business Intelligence (BI) output;
- EDQ Dashboard results;

- Server Console results; and
- Results Books in EDQ.

5.5.2.1 BI Output

Health Check can produce two comma-separated files containing Individual and Entity results data. This output is intended for detailed analysis using an external Business Intelligence application.

The files are:

- `entityoutput.csv`
- `individualoutput.csv`

Records passed into Health Check will cause one or more rows to be generated, depending on the content of each record and how many errors are discovered within each record.

> **Note :** The separators, and file names and locations within the landing area can be configured in the Run Profile.

The most important metadata attributes in the `.csv` files are as follows:

Column	Description
`entityid` / `individualid`	The id of the original record.
`Data Stream`	This field identifies the origin of the row:`Misc Data` - A record fields.`AddressData` - An address.`AltPhoneData` - An `altphone` field.
`Rule ID`	The ID of the rule triggered, if applicable.
`Rule Label`	The label of the rule triggered, if applicable.
`Error Code`	The code of the error, if applicable.
`Error Severity`	The severity level of the error, if applicable.
`Error Message`	The error message returned, if applicable.

The logic is as follows:

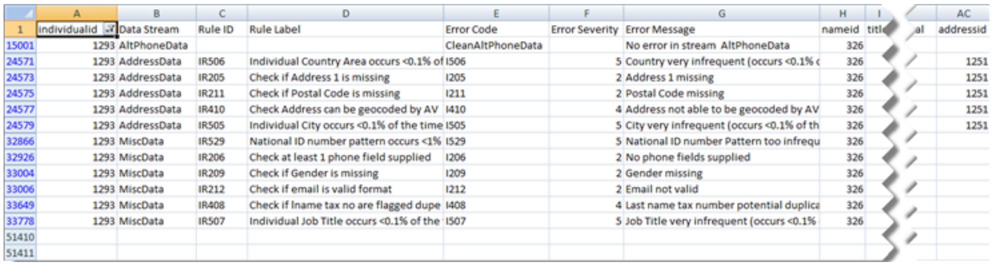
- Each record passed into Health Check returns at least one row in the corresponding `.csv` file.
- At least one row is generated per record. If there is an error in the record data, this is indicated in the Error Code, Error Severity and Error Message columns.
- An additional row is generated per address or `altphone` field within each record. Again, if there is a single error in an address or `altphone` field, this is indicated in the Error columns.
- However, if a record, address or `altphone` field contains more than one error, then a row is generated for each additional error above one.

For example, if an individual record has:

- no address or `altphone` value and no errors: 1 row.
- no address or `altphone` value, and one error: 1 row.
- no address or `altphone` value, and two errors: 2 rows.

- an address, but no `altphone`: 2 rows.
- an address and an `altphone`: 3 rows.
- an address containing a single error, and an `altphone`: 3 rows.
- an address containing two errors, and an `altphone`: 4 rows.

The following is a complex example. The record with `individualid` `1293` has returned 12 rows:



	A	B	C	D	E	F	G	H	I		AC
1	individualid	Data Stream	Rule ID	Rule Label	Error Code	Error Severity	Error Message	nameid	titl	al	addressid
15001	1293	AltPhoneData			CleanAltPhoneData		No error in stream AltPhoneData	326			
24571	1293	AddressData	IR506	Individual Country Area occurs <0.1% of	I506	5	Country very infrequent (occurs <0.1% c	326			1251
24573	1293	AddressData	IR205	Check if Address 1 is missing	I205	2	Address 1 missing	326			1251
24575	1293	AddressData	IR211	Check if Postal Code is missing	I211	2	Postal Code missing	326			1251
24577	1293	AddressData	IR410	Check Address can be geocoded by AV	I410	4	Address not able to be geocoded by AV	326			1251
24579	1293	AddressData	IR505	Individual City occurs <0.1% of the time	I505	5	City very infrequent (occurs <0.1% of th	326			1251
32866	1293	MiscData	IR529	National ID number pattern occurs <1%	I529	5	National ID number Pattern too infrequ	326			
32926	1293	MiscData	IR206	Check at least 1 phone field supplied	I206	2	No phone fields supplied	326			
33004	1293	MiscData	IR209	Check if Gender is missing	I209	2	Gender missing	326			
33006	1293	MiscData	IR212	Check if email is valid format	I212	2	Email not valid	326			
33649	1293	MiscData	IR408	Check if lname tax no are flagged dupe	I408	4	Last name tax number potential duplica	326			
33778	1293	MiscData	IR507	Individual Job Title occurs <0.1% of the	I507	5	Job Title very infrequent (occurs <0.1%	326			
51410											
51411											

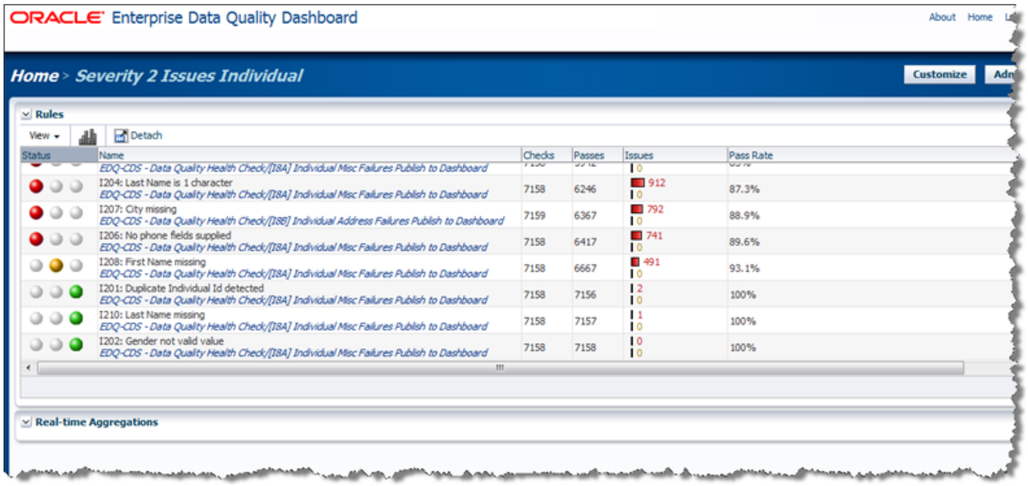
It has the following:

- One `altphone` field, free of errors.
- Five errors associated with one address.
- Six errors associated with other fields in the record (for example, Misc Data.)

Note : In the example file, the `addressid` in each row is identical, which shows that only one address is associated with the record. The illustration does not show this because of the limit of the screen size.

5.5.2.2 EDQ Dashboard

The results published to the Dashboard are dependent on the enabled Business Rules, see Section 5.6, "Managing Business Rules". The following Dashboard example illustrates the variations of results and statuses:



Home > Severity 2 Issues Individual

Status	Name	Checks	Passes	Issues	Pass Rate
	EDQ-CDS - Data Quality Health Check/[I8A] Individual Misc Failures Publish to Dashboard			0	
🔴 ⚪ ⚪	I204: Last Name is 1 character EDQ-CDS - Data Quality Health Check/[I8A] Individual Misc Failures Publish to Dashboard	7158	6246	912 0	87.3%
🔴 ⚪ ⚪	I207: City missing EDQ-CDS - Data Quality Health Check/[I8B] Individual Address Failures Publish to Dashboard	7159	6367	792 0	88.9%
🔴 ⚪ ⚪	I206: No phone fields supplied EDQ-CDS - Data Quality Health Check/[I8A] Individual Misc Failures Publish to Dashboard	7158	6417	741 0	89.6%
⚪ 🟠 ⚪	I208: First Name missing EDQ-CDS - Data Quality Health Check/[I8A] Individual Misc Failures Publish to Dashboard	7158	6667	491 0	93.1%
⚪ ⚪ 🟢	I201: Duplicate Individual Id detected EDQ-CDS - Data Quality Health Check/[I8A] Individual Misc Failures Publish to Dashboard	7158	7156	2 0	100%
⚪ ⚪ 🟢	I210: Last Name missing EDQ-CDS - Data Quality Health Check/[I8A] Individual Misc Failures Publish to Dashboard	7158	7157	1 0	100%
⚪ ⚪ 🟢	I202: Gender not valid value EDQ-CDS - Data Quality Health Check/[I8A] Individual Misc Failures Publish to Dashboard	7158	7158	0 0	100%

The results from attributes associated with the Individual or Entity record (such as, `name`, `title`, `email` and so on) are based on distinct Individual and Entity records identified by a unique record ID.

Checks on the `altphone` attribute and address-related attributes are performed separately so that the number of results produced correctly reflects the number of child entities processed.

Similarly, results from the `altphone` field are based on distinct alternate phone numbers in Individual and Entity records, as it is possible to have multiple `altphone` values per record.

The results from attributes associated with addresses (such as, `city`, `postalcode`, `country` and so on) are based on distinct address records identified by a unique address id because it is possible to process multiple addresses for a given Individual or Entity.

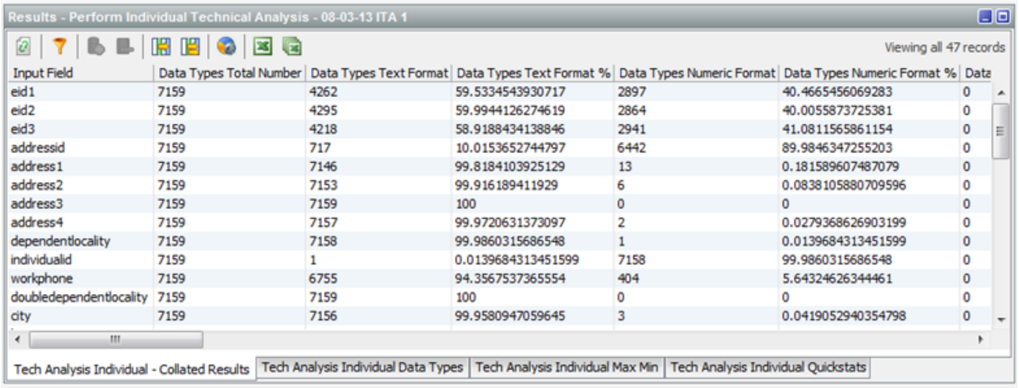
The number of checks for a given published rule in the Dashboard may vary depending on the type of data being checked, and will always relate to the total population of the type of data. So the "total" figures displayed may vary according to data type.

For example, if 500,000 records were passed from the customer system, with a total of 650,000 addresses attached, and a total of 550,000 alternate phone numbers associated with them, then all results will show:

- all address-related rule failures/passes as a percentage of 650,000;
- all alternate-phone-related rule failures/passes as a percentage of 550,000; and
- all remaining rule failure/passes as a percentage of 500,000.

5.5.2.3 Server Console

When run in Server Console, the Technical Analysis jobs profile the data by data type, maximum and minimum values and quick stats:



Results - Perform Individual Technical Analysis - 08-03-13 ITA 1

Viewing all 47 records

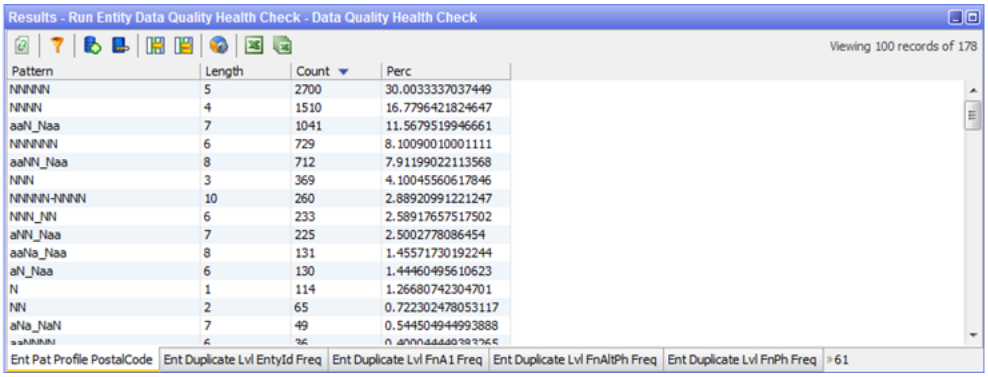
Input Field	Data Types Total Number	Data Types Text Format	Data Types Text Format %	Data Types Numeric Format	Data Types Numeric Format %	Data
eid1	7159	4262	59.5334543930717	2897	40.4665456069283	0
eid2	7159	4295	59.9944126274619	2864	40.0055873725381	0
eid3	7159	4218	58.9188434138846	2941	41.0811565861154	0
addressid	7159	717	10.0153652744797	6442	89.9846347255203	0
address1	7159	7146	99.8184103925129	13	0.181589607487079	0
address2	7159	7153	99.916189411929	6	0.0838105880709596	0
address3	7159	7159	100	0	0	0
address4	7159	7157	99.9720631373097	2	0.0279368626903199	0
dependentlocality	7159	7158	99.9860315686548	1	0.0139684313451599	0
individualid	7159	1	0.0139684313451599	7158	99.9860315686548	0
workphone	7159	6755	94.3567537365554	404	5.64324626344461	0
doubledependentlocality	7159	7159	100	0	0	0
city	7159	7156	99.9580947059645	3	0.0419052940354798	0

Tech Analysis Individual - Collated Results
Tech Analysis Individual Data Types
Tech Analysis Individual Max Min
Tech Analysis Individual Quickstats

The Health Check jobs perform audit checks on the data and populate the EDQ Dashboard and BI `.csv` files depending on your run profile configuration.

Note : Running the jobs in Server Console does *not* populate the Health Check Results Books.

An example of the Server Console Results, depending on the Run Profile, is as follows:



Results - Run Entity Data Quality Health Check - Data Quality Health Check

Viewing 100 records of 178

Pattern	Length	Count	Perc
NNNNN	5	2700	30.0033337037449
NNNN	4	1510	16.7796421824647
aaN_Naa	7	1041	11.5679519946661
NNNNNN	6	729	8.10090010001111
aaNN_Naa	8	712	7.91199022113568
NNN	3	369	4.10045560617846
NNNNN-NNNN	10	260	2.88920991221247
NNN_NN	6	233	2.58917657517502
aNN_Naa	7	225	2.5002778086454
aaNa_Naa	8	131	1.45571730192244
aN_Naa	6	130	1.44460495610623
N	1	114	1.26680742304701
NN	2	65	0.722302478053117
aNa_NaN	7	49	0.544504944993888
aaNNNN	6	36	0.400044449383265

Ent Pat Profile PostalCode
Ent Duplicate Lvl EntyId Freq
Ent Duplicate Lvl FnA1 Freq
Ent Duplicate Lvl FnAltPh Freq
Ent Duplicate Lvl FnPh Freq
»61

5.5.2.4 Results Books

If activated in the Health Check Run Profile, the following Results Books can be populated:

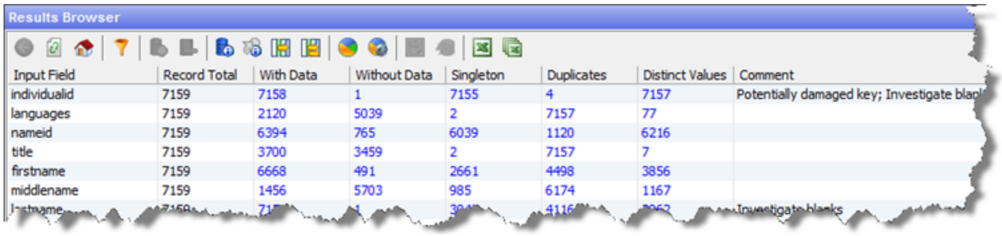
- Entity Profiling Results
- Entity Rules Results
- Entity Technical Analysis
- Individual Profiling Results
- Individual Rules Results
- Individual Technical Analysis

The Technical Analysis Results Books are populated by the corresponding Technical Analysis jobs. The Profiling Results and Rules Results Books are populated by the corresponding Health Check jobs.

Consider the following:

- When running these jobs, select the **edq-cds-data-quality-health-check** Run Profile, but *do not* specify a Run Label.
- The Results Books are *only* populated if the Data Quality jobs are run from EDQ. Running the jobs either from Siebel or Server Console will not populate Results Book data.
- The Business Object grouping of rules in Results Books is pulled from the `Business Object` column in `rulesreference.xls` where each rule is associated with a business object text value. To reclassify rules, edit the `Business Object` column.
- The Technical Analysis jobs only use customer data and publish the analysis results to Server Console or in Results Books only.

It is possible to drill-down through these results for further analysis. Drillable results are links (highlighted in blue):



Results Browser

Input Field	Record Total	With Data	Without Data	Singleton	Duplicates	Distinct Values	Comment
individualid	7159	7158	1	7155	4	7157	Potentially damaged key; Investigate blan
languages	7159	2120	5039	2	7157	77	
nameid	7159	6394	765	6039	1120	6216	
title	7159	3700	3459	2	7157	7	
firstname	7159	6668	491	2661	4498	3856	
middlename	7159	1456	5703	985	6174	1167	
lastname	7159	[illegible]	1	[illegible]	4116	[illegible]	Investigate blanks

5.6 Managing Business Rules

This section provides several examples describing how to turn on, edit and add business rules.

5.6.1 Example - Turning on a Rule

The Entity rule `ER418 - Country is missing` is disabled by default.

To turn the rule on:

1. Navigate to the **`oedq_local_home/landingarea/dqhealthcheck`** folder.
2. Open the **`rulesreference.xls`** file.
3. Select the **Address** tab.
4. Find the E418 rule row, and change the value of the cell in the **Enabled** column to **yes**.
5. Save the file.
6. If required, open the Dashboard Administration application to add the rule to an appropriate Summary.

To disable the rule again, repeat this procedure, changing the cell value back to **no**.

Note : If a rule that is included in a Dashboard Summary is disabled, it will still be displayed in the Summary with no results returned. Therefore, it is recommended that any disabled rules be removed from Dashboard Summaries so they do not influence overall pass or failure indicators.

5.6.2 Example - Editing Rules: Adding an Extra Common Title

The `titles` tab in the `dq-health-check-business-rules-individual.xls` file is used by rule `IR411- Check Upper Case Title is in the list`.

The following procedure shows how to ensure the rule also checks for the term "PROFESSOR" as an common title:

1. Navigate to the **`oedq_local_home/businessrules`** folder, and open the **`dq-health-check-business-rules-individual.xls`** spreadsheet.
2. Select the **titles** tab.
3. Add **PROFESSOR** to the bottom of the list in column A of the worksheet.
4. Save the file.

5.6.3 Example - Editing a Rule: Changing a Value Check

This example describes how to change the value check of the `IR430 - Check if DOB is very old (<1900)` to check for birthdates older than 1890.

1. Navigate to the **`oedq_local_home/businessrules`** folder, and open the **`dq-health-check-business-rules-individual.xls`** spreadsheet.
2. Select the **Rules** tab, and scroll to the **IR430** rule.
3. Change the Rule Label to **Check if DOB is very old (<1890)**.
4. Scroll to the **Check1** column, and change the cell value to **chGreaterThan1890**.

5. Select the **Checks** tab, and select the two rows that describe the **chGreaterThan1900** check. In an unmodified sheet, these are normally rows 39 and 40.
6. Copy the rows, and paste them below the existing Checks.
7. Edit the Description, Check Name and Option 1 cells, replacing "1900" with **1890**.
8. Save the file.
9. Open **Director**, and navigate to the **Processe**s node of the EDQ-CDS Data Quality Health Check project in the Project Browser.
10. Double click the **[I8A] Individual Misc Failures Publish to Dashboard** process.
11. In the Process Canvas, locate the following processors in the **Misc Checks** group:
    - IR430 DOB Year is older than 1900 Enabled
    - IR430 DOB Year is older than 1900
12. Edit the labels of these processors (for example, change "1900" to "1890").
13. Double click the **IR430 DOB Year is older than 1890** processor.
14. Select the **Dashboard** tab in the Processor dialog.
15. Edit the rule name to read **I430: DOB year is older than 1890**.
16. Click **OK**.
17. Close the process, saving the changes made.
18. Navigate to the **`oedq_local_home/landingarea/dqhealthcheck`** folder.
19. Open the **rulesreference.xls** file.
20. Select the **Misc** tab and find the IR430 rule.
21. Change the Description to **DOB year is older than 1890**.
22. Save the file.

5.6.4 Example - Editing a Rule: Changing the Severity Level

This example describes how to change the severity level of rule `IR308 - Check if email is missing` from 3 to 2.

1. Navigate to the **`oedq_local_home/businessrules`** folder, and open the **`dq-health-check-business-rules-individual.xls`** spreadsheet.
2. Select the **rules** tab and locate the **IR308** rule.
3. Scroll to the **Error Severity** column and change the cell value to **2**.
4. Save the file.
5. If Severity Summaries have already been configured for Dashboard, open Dashboard Administration, remove the IR308 rule from the Severity 3 Summary and add it to the Severity 2 Summary.

5.6.5 Example - Adding a Rule

This example describes how to add a rule to check that a delivery address post code field passed into the `customstring1` attribute in individual records contains no more than 9 digits, excluding punctuation (for example, conforms to the US zip code format). For this rule to be effective, it will be necessary to clean the field data first by removing any spaces or punctuation marks. This will ensure that only the alphanumeric content is checked

There are eight stages to adding this rule:

1. Confirm the field is passed to the Business Rules processor.
2. Check field format to check the results of a previously-run job in the Server Console Results window, specifically the DQ Health Check Analysis Output tab. This tab is not visible by default. Therefore, before running through this example ensure the `stageddata.Individual\ DQ\ Health\ Check\ Analysis\ Output.visible` attribute is set to `Yes`.
3. Insert pre-processing to reformat the field data.
4. Edit the Business Rules spreadsheet.
5. Edit the **`rulesreference.xls`** spreadsheet.
6. Change the Business Rules Check processor.
7. Configure for Dashboard.
8. Update the Dashboard Summaries.

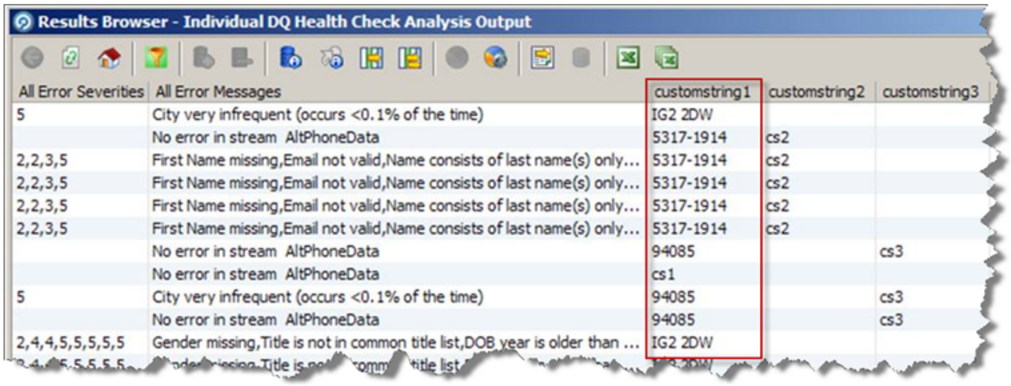
> **Note:** The following examples require a solid understanding of process design in Director and the associated permissions.

Confirming the Field is Passed to the Business Rules Processor

1. Open **Director**, and navigate to the **Processes** node of the **EDQ-CDS - Data Quality Health Check** project in the Project Browser.
2. Double click the **[I6A] Run Misc Business Rules** process.
3. Double-click the **Business Rules Check** processor in the **Business Rules Execution** group at the bottom of the Process Canvas.
4. In the **Attributes** tab of the processor dialog, scroll through the **Attributes** field to confirm the **customstring1** attribute is included.
5. Click the **Identify** tab.
6. Check the Identifier assigned to the **customstring1** Input Attribute (`atCustomString1` in a default installation).

Checking the `field` Format

1. Start Server Console.
2. In the **Results** view, select a previous run of Health Check.
3. The **DQ Health Check Analysis Output** tab should be displayed at the bottom of the window by default. Scroll across to view the **customstring1** column and check the format of the results. In the example image, the format is clearly incorrect: as one field contains a space and the other a hyphen it is not limited to alphanumeric data only:



Results Browser - Individual DQ Health Check Analysis Output

All Error Severities	All Error Messages	customstring1	customstring2	customstring3
5	City very infrequent (occurs <0.1% of the time)	IG2 2DW		
	No error in stream AltPhoneData	5317-1914	cs2	
2,2,3,5	First Name missing,Email not valid,Name consists of last name(s) only...	5317-1914	cs2	
2,2,3,5	First Name missing,Email not valid,Name consists of last name(s) only...	5317-1914	cs2	
2,2,3,5	First Name missing,Email not valid,Name consists of last name(s) only...	5317-1914	cs2	
2,2,3,5	First Name missing,Email not valid,Name consists of last name(s) only...	5317-1914	cs2	
	No error in stream AltPhoneData	94085		cs3
	No error in stream AltPhoneData	cs1		
5	City very infrequent (occurs <0.1% of the time)	94085		cs3
	No error in stream AltPhoneData	94085		cs3
2,4,4,5,5,5,5,5	Gender missing,Title is not in common title list,DOB year is older than ...	IG2 2DW		

4. Close Server Console.

Inserting Pre-Processing to Format Field Data

As the format of the data in the `customstring1` field does not match the required 9-character alphanumeric format, some pre-processing of the data is required before it is passed to the Business Rules Check processor. Also, to avoid affecting the output of the processor, the pre-processing will be performed on a copy of the `customstring1` data that will then be passed to the check.

1. Return to **Director**.
2. Add a **Concatenate** processor to the **[I16A] Run Misc Business Rules** process, positioning it immediately before the Business Rules Check processor.
3. Configure the processor to take a copy of the customstring1 string, called **customstring1ForChk**.
4. Follow this processor with a **Remove Whitespace** and **Denoise** processor, configuring them to clean the customstring1ForChk data.
5. Save the changes. Leave Director open, as further changes to the Business Rules Check processor are required.

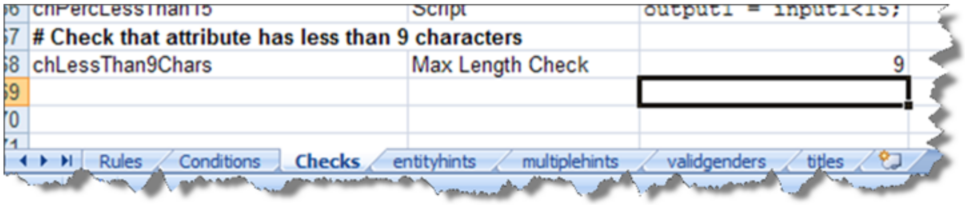
Editing the Business Rules Spreadsheet

It is now possible to edit the `dq-health-check-business-rules-individual.xls` spreadsheet. This involves adding a new Check, Condition and Business Rule.

> **Note :** The Condition is required in order to ensure that the rule is not applied in circumstances where the `customstring1ForChk` field is not present in the data being analyzed.

1. Navigate to the `oedq_local_home/businessrules` folder, and open the `dq-health-check-business-rules-individual.xls` spreadsheet.
2. Click the **Checks** tab.
3. Create a new entry for a check specifying a maximum length of nine characters.

> **Note :** The wording describes the check taking place. In order to fail entries of more than nine characters, the check performed is actually whether the entries are nine characters long or less.



56	chPercLessThan15	Script	output1 = input1<15;
57	# Check that attribute has less than 9 characters		
58	chLessThan9Chars	Max Length Check	9
59			
70			

4. Click the **Conditions** tab.

5. Copy and paste the **coCustomString1_supplied** row into an empty row at the bottom of the sheet.

6. Edit the Condition Name and Attribute or Check cells of the new entry to read **coCustomStringForChk_supplied** and **coCustomStringForChk** respectively.

7. Click the **Rules** tab.

8. Add a new line describing the rule, applying the following values:

   - Rule ID: **IR391**
   - Rule Label: **Custom String 1 (denoised) greater than 9 chars**
   - Disable: Leave blank.
   - Apply to Attribute: **atCustomString1ForChk**
   - Condition: **coCustomString1ForChk_supplied**
   - Error Code: **I391**
   - Error Severity: **3**
   - Error Message: **Custom String 1 (denoised) is greater than 9 characters**
   - Check1: **chLessThan9Chars**

9. Click **Save** and close the spreadsheet.

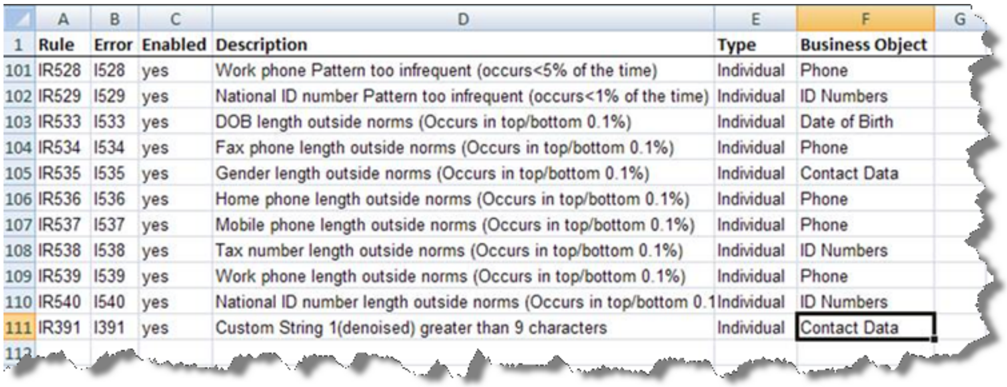
Editing the `rulesreference.xls` Spreadsheet

1. Navigate to the **`oedq_local_home/landingarea/dqhealthcheck`** folder.

2. Open the **`rulesreference.xls`** file.

3. Click the **Misc** tab.

4. Add the details of the new rule to the bottom of the worksheet, as illustrated in following:



	A	B	C	D	E	F	G
1	Rule	Error	Enabled	Description	Type	Business Object	
101	IR528	I528	yes	Work phone Pattern too infrequent (occurs<5% of the time)	Individual	Phone	
102	IR529	I529	yes	National ID number Pattern too infrequent (occurs<1% of the time)	Individual	ID Numbers	
103	IR533	I533	yes	DOB length outside norms (Occurs in top/bottom 0.1%)	Individual	Date of Birth	
104	IR534	I534	yes	Fax phone length outside norms (Occurs in top/bottom 0.1%)	Individual	Phone	
105	IR535	I535	yes	Gender length outside norms (Occurs in top/bottom 0.1%)	Individual	Contact Data	
106	IR536	I536	yes	Home phone length outside norms (Occurs in top/bottom 0.1%)	Individual	Phone	
107	IR537	I537	yes	Mobile phone length outside norms (Occurs in top/bottom 0.1%)	Individual	Phone	
108	IR538	I538	yes	Tax number length outside norms (Occurs in top/bottom 0.1%)	Individual	ID Numbers	
109	IR539	I539	yes	Work phone length outside norms (Occurs in top/bottom 0.1%)	Individual	Phone	
110	IR540	I540	yes	National ID number length outside norms (Occurs in top/bottom 0.1	Individual	ID Numbers	
111	IR391	I391	yes	Custom String 1(denoised) greater than 9 characters	Individual	Contact Data	
112							

5. Click **Save** and close the spreadsheet.

Changing the Business Rules Check Processor

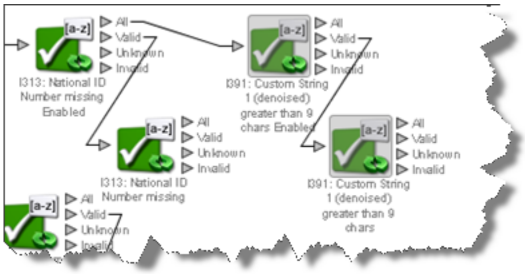
The Business Rules Check processor must be changed to use the reformatted field:

1. Return to **Director**.
2. Double-click the **Business Rules Check** processor in the **[I6A] Run Misc Business Rules** process.
3. On the **Attributes** tab of the **Processor** dialog, add the **customstring1ForChk** attribute to the Input Attributes.
4. Click the **Identify** tab.
5. Find the **atCustomString1ForChk** identifier, and assign the **customstring1ForChk** input attribute in the drop-down field to it.
6. Save the changes and close the dialog.

Configuring for Dashboard

If the Dashboard is used, it is necessary to make further changes to publish the results of the new rule.

1. In **Director**, open the **[I8A] Individual Misc Failures Publish to Dashboard** process.
2. Make a copy of a group of two of the Audit processors. For the purposes of this example, copy the I313 Audit Processors.
3. Paste the copies onto the canvas to the right of the I313 Processors.
4. Rename both copied processors: **I391: Custom String 1 (denoised) greater than 9 chars Enabled** and **I391: Custom String 1 (denoised) greater than 9 chars**.
5. Connect the **All** output of the I313: National ID Number missing processor to the I391: Custom String 1 (denoised) greater than 9 chars Enabled processor. The processors should now appear as in the following:


[a-z]
All
Valid
Unknown
Invalid
I313: National ID Number missing Enabled
I391: Custom String 1 (denoised) greater than 9 chars Enabled
I313: National ID Number missing
I391: Custom String 1 (denoised) greater than 9 chars

6. Double-click the **I391: Custom String 1 (denoised) greater than 9 chars Enabled** processor.
7. In the Processor dialog, click the **Options** tab.
8. Set the Regular Expression field to **I391** .
9. Click **Save** and close the dialog.
10. Repeat steps 7 to 9 for the **I391: Custom String 1 (denoised) greater than 9 chars** processor.
11. Click the **Dashboard** tab.
12. Set the **Rule Name** field to "I391: Custom String 1 (denoised) greater than 9 chars".
13. Save changes, and close the process.

Updating the Dashboard Summaries

Once the **Individual Data Quality Health Check** job has been run again, it is possible to add the new rule to the required Summary in the Dashboard Administration application.

5.7 Understanding Data Interfaces

This section describes the two Health Check data interfaces and all of the attributes contained in each of them.

- Section 5.7.1, "Individual Data"
- Section 5.7.2, "Entity Data"

5.7.1 Individual Data

All the Individual Data attributes are strings:

Attribute	Description
`individualid`	Unique identifier of the individual (e.g customer, employee or contact).
`languages`	Three-character Siebel language code. Only used by EDQ-CDS in name standardization to help determine whether a name containing Kanji is Japanese or Chinese.
`nameid`	Unique identifier for the name. Used by EDQ-CDS to distinguish between different names for the same individual when multiple child entities are used. For more information, see Chapter 2, "Using Business Services."
`title`	
`firstname`	
`middlename`	
`lastname`	
`gender`	M or F.
`dob`	Date of Birth in one of the formats listed in the ***Date Formats** EDQ Reference Data set.
`jobtitle`	
`homephone`	
`workphone`	
`mobilephone`	
`faxphone`	
`alternatephone`	
`email`	
`taxnumber`	
`nationalidnumber`	Social Security Number (US) or equivalent.
`accountname`	The name of the account (for example, entity) to which this individual belongs, if relevant.

Attribute	Description
uid1	Unique ID 1 **NOTE**: The Unique ID fields are used in EDQ-CDS to match records based on custom unique identifiers, such as passport or tax numbers. For more information, see Chapter 3, "Using Matching."
uid2	Unique ID 2.
uid3	Unique ID 3.
eid1	Elimination ID 1. **Note**: The Elimination ID fields are used in EDQ-CDS to eliminate possible matches between records based on custom unique identifiers, such as passport or tax numbers. For more information, see Chapter 3, "Using Matching."
eid2	Elimination ID 2.
eid3	Elimination ID 3.
addressid	Unique identifier for the address, used in EDQ-CDS to distinguish between different addresses for the same individual when multiple child entities are used. For more information, see Chapter 2, "Using Business Services."
address1	Line 1 of the address.
address2	Line 2 of the address.
address3	Line 3 of the address.
address4	Line 4 of the address.
dependentlocality	A smaller population center data element than `city`, for example, a Turkish neighborhood.
doubledependentlocality	The smallest population center data element, dependent on both the contents of the `city` and `dependentlocality` fields. For example, UK Village.
city	
subadminarea	The smallest geographic data element within a country. For example, USA County.
adminarea	The most common geographic data element within a country. For example, USA State or Canadian Province.
postalcode	
country	Country name or ISO 2 char code. **Note**: The output will always be the full Country name, even if the input is the country ISO code.
customstring1	The `customstring` fields are placeholders for data attributes that require analysis in Health Check but do not match to any of the standard interface attributes.
customstring2	
customstring3	
customstring4	
customstring5	
customstring6	
customstring7	

Attribute	Description
customstring8	
customstring9	
customstring10	

5.7.2 Entity Data

All the Entity Data attributes are strings.

Attribute	Description
nameid	Unique identifier for the name, used in EDQ-CDS to distinguish between different names for the same entity when multiple child entities are used. For more information, see Chapter 2, "Using Business Services."
entityid	Unique record identifier.
languages	Three-character Siebel language code. Only used in EDQ-CDS for name standardization to help determine whether a name containing Kanji is Japanese or Chinese.
name	Organization name, for example, "Oracle Corporation UK".
subname	Department or site, for example, "Reading" or "Accounts Payable".
phone	
alternatephone	
website	
taxnumber	
vatnumber	
uid1	Unique ID 1 **Note**: The Unique ID fields are used in EDQ-CDS to match records based on custom unique identifiers, such as passport or tax numbers. For more information, see Chapter 3, "Using Matching."
uid2	Unique ID 2.
uid3	Unique ID 3.
eid1	Elimination ID 1. **Note**: The Elimination ID fields are used in EDQ-CDS to eliminate possible matches between records based on custom unique identifiers, such as passport or tax numbers.For more information, see Chapter 3, "Using Matching."
eid2	Elimination ID 2.
eid3	Elimination ID 3.
addressid	Unique identifier for the address.
address1	
address2	
address3	
address4	

Attribute	Description
dependentlocality	A smaller population center data element than city, for example, a Turkish neighborhood.
doubledependentlocality	The smallest population center data element, for example, a UK village.
city	
subadminarea	The smallest geographic data element within a country, for example, US county.
adminarea	The most common geographic data element within a country, for example, US state, Canadian province, UK county.
postalcode	
country	
customstring1	The customstring fields are placeholders for data attributes that require analysis in Health Check but do not match to any of the standard interface attributes.
customstring2	
customstring3	
customstring4	
customstring5	
customstring6	
customstring7	
customstring8	
customstring9	
customstring10	

5.8 Dashboard Example Summaries

These tables contain the rules to be included in the Summaries described in Section 5.4.2, "Example - Dashboard By Business Function."

Account - Name Details

Audit Code	Description
E101	Full Name missing
E202	Name is 1 character
E205	Name missing
E302	Sub Name is 1 character
E303	SubName missing
E408	Name contains potential multiples hints
E409	Sub Name contains potential multiples hints
E411	Unusual characters in name
E412	Unusual characters in subname

Account - Identifiers

Audit Code	Description
E102	Entity Id missing
E204	No phone fields supplied
E304	Tax Number missing
E305	VAT Number missing
E306	Website missing
E307	Website not valid
E410	Alternate phone is missing
E413	Unusual characters in phone
E417	Alternate phone is missing
E419	Phone is missing

Account - Identifier Outliers

Audit Code	Description
E420	Alt Phone appears to have less than 2 digits present
E510	Alt phone length outside norms (Occurs in top/bottom 0.1%)
E520	Alt phone Pattern too infrequent (occurs<5% of the time)
E421	Phone appears to have less than 2 digits present
E504	Tax Number too frequent (occurs>5% of the time)
E505	VAT Number too frequent (occurs>5% of the time)
E506	Website too frequent (occurs>5% of the time)
E513	Phone length outside norms (Occurs in top/bottom 0.1%)
E514	Tax number length outside norms (Occurs in top/bottom 0.1%)
E515	VAT number length outside norms (Occurs in top/bottom 0.1%)
E521	Phone Pattern too infrequent (occurs<5% of the time)
E523	Tax number Pattern too infrequent (occurs<1% of the time)
E524	VAT number Pattern too infrequent (occurs<1% of the time)
E525	Website Pattern too frequent (occurs>5% of the time)

Account - Address Details

Audit Code	Description
E203	Address 1 missing
E206	Postal Code missing
E207	City missing
E301	Address not able to be verified by AV processor
E308	Addresses 2 and 3 missing
E407	Address not able to be geocoded by AV processor

Audit Code	Description
E414	Address 2 is missing
E415	Address 3 is missing
E416	Admin area is missing
E418	Country is missing

Account - Address Detail Outliers

Audit Code	Description
E501	Admin Area very infrequent (occurs <0.1% of the time)
E502	City very infrequent (occurs <0.1% of the time)
E503	Country very infrequent (occurs <0.1% of the time)
E511	City length outside norms (Occurs in top/bottom 0.1%)
E512	Country length outside norms (Occurs in top/bottom 0.1%)
E522	Postal code Pattern too infrequent (occurs<1% of the time)

Account - Potential Duplicates

Audit Code	Description
E201	Duplicate Entity Id detected
E401	Full name address1 potential duplicate
E402	Full name alt phone potential duplicate
E403	Full name phone potential duplicate
E404	Full name website potential duplicate
E405	Name tax number potential duplicate
E406	Name VAT number potential duplicate

Contact - Name Details

Audit Code	Description
I101	Full Name missing
I203	First Name is 1 character
I204	Last Name is 1 character
I208	First Name missing
I210	Last Name missing
I301	Name consists of last name(s) only
I304	Middle name is 1 character
I310	Middle name missing
I411	Title is not in common title list
I418	Title is missing

Audit Code	Description
I420	Unusual characters in first name
I421	Unusual characters in last name
I422	Unusual characters in middle name
I428	Full Name contains potential entity hints
I429	Full Name contains potential multiples hints

Contact - Identifiers

Audit Code	Description
I102	Individual Id missing
I206	No phone fields supplied
I212	Email not valid
I302	Gender and title are not consistent
I305	Account Name is missing
I307	DOB missing
I308	Email missing
I311	Tax Number missing
I312	DOB in future
I313	National ID Number missing
I413	Alternate phone is missing
I415	Fax phone is missing
I416	Home phone is missing
I417	Mobile phone is missing
I419	Work phone is missing
I423	Unusual characters in alternate phone
I424	Unusual characters in fax phone
I425	Unusual characters in home phone
I426	Unusual characters in mobile phone
I427	Unusual characters in work phone

Contact - Identifier Outliers

Audit Code	Description
I202	Gender not valid value
I209	Gender missing
I430	DOB year is older than 1900
I433	Alt Phone appears to have less than 2 digits present
I434	Home Phone appears to have less than 2 digits present
I435	Mobile Phone appears to have less than 2 digits present

Audit Code	Description
I436	Work Phone appears to have less than 2 digits present
I437	Fax Phone appears to have less than 2 digits present
I501	Account Name too frequent (occurs>5% of the time)
I502	Email too frequent (occurs>5% of the time)
I503	Tax Number too frequent (occurs>5% of the time)
I508	Title very infrequent (occurs <0.1% of the time)
I509	National ID Number too frequent (occurs>5% of the time)
I510	DOB day in year too frequent (occurs >1% of the time)
I511	DOB Year too frequent (occurs >5% of the time)
I512	DOB Month too frequent (occurs >10% of the time)
I513	DOB Day In Week too frequent (occurs >15% of the time)
I514	DOB Day in Month too frequent (occurs >5% of the time)
I520	Alt phone Pattern too infrequent (occurs<5% of the time)
I521	DOB Pattern too infrequent (occurs<5% of the time)
I522	Email Pattern too frequent (occurs>5% of the time)
I523	Fax phone Pattern too infrequent (occurs<5% of the time)
I524	Home phone Pattern too infrequent (occurs<5% of the time)
I525	Mobile phone Pattern too infrequent (occurs<5% of the time)
I527	Tax number Pattern too infrequent (occurs<1% of the time)
I528	Work phone Pattern too infrequent (occurs<5% of the time)
I529	National ID number Pattern too infrequent (occurs<1% of the time)
I530	Alt phone length outside norms (Occurs in top/bottom 0.1%)
I533	DOB length outside norms (Occurs in top/bottom 0.1%)
I534	Fax phone length outside norms (Occurs in top/bottom 0.1%)
I536	Home phone length outside norms (Occurs in top/bottom 0.1%)
I537	Mobile phone length outside norms (Occurs in top/bottom 0.1%)
I538	Tax number length outside norms (Occurs in top/bottom 0.1%)
I539	Work phone length outside norms (Occurs in top/bottom 0.1%)
I540	National ID number length outside norms (Occurs in top/bottom 0.1%)

Contact - Address Details

Audit Code	Description
I205	Address 1 missing
I207	City missing
I211	Postal Code missing
I303	Address not able to be verified by AV processor
I306	Addresses 2 and 3 missing

Audit Code	Description
I410	Address not able to be geocoded by AV processor
I412	Admin area is missing
I414	Country is missing
I431	Address 2 is missing
I432	Address 3 is missing

Contact - Address Detail Outliers

Audit Code	Description
I504	Admin Area very infrequent (occurs <0.1% of the time)
I505	City very infrequent (occurs <0.1% of the time)
I506	Country very infrequent (occurs <0.1% of the time)
I526	Postal code Pattern too infrequent (occurs<1% of the time)
I531	City length outside norms (Occurs in top/bottom 0.1%)
I532	Country length outside norms (Occurs in top/bottom 0.1%)

Contact - Potential Duplicates

Audit Code	Description
I201	Duplicate Individual Id detected
I401	Full name address1 potential duplicate
I402	Full name alt phone potential duplicate
I403	Full name email potential duplicate
I404	Full name fax phone potential duplicate
I405	Full name home phone potential duplicate
I406	Full name mobile phone potential duplicate
I407	Full name work phone potential duplicate
I408	Last name tax number potential duplicate
I409	Last name national id number potential duplicate