Oracle FCCM Cloud Service

Customer Screening Matching Guide

Release 25.11.01
G45954-01
November 2025

Oracle FCCM Cloud Service Customer Screening Matching Guide, Release 25.11.01

G45954-01

Contents

Preface

Audience i
Help i
Related Resources i
Diversity and Inclusion i
Documentation Accessibility i
Conventions ii
Comments and Suggestions ii

1 Introduction

2 Matching

2.1 Introduction 1
2.2 How It Works 4
2.3 Date Matching 5
2.4 Processing Null Values 6

3 Scoring Methods

3.1 Jaro Winkler 1
3.2 Levenshtein 1
3.3 Individual Name 2
3.4 Individual SAN 5
3.5 Individual PEP 5
3.6 Individual EDD 6
3.7 Entity Name 6
3.8 Entity SAN 7
3.9 Entity PEP 7
3.10 Entity EDD 7

4 Rulesets

4.1 Out of the Box Rulesets 1
4.1.1 Match Rules 1
4.2 Modifying the Ruleset 2

5 Data Processing

5.1 Transliteration 1
5.1.1 Original Script Matching 1
5.1.2 Input Fields for Individual Screening 1
5.1.3 Input fields for Entity Screening 2
5.2 Analyzers 2
5.2.1 Synonyms 8
5.2.2 Stopwords 8
5.3 Treating Special and Accented Characters 9

Preface

Matching Guide provides a flexible and customizable strategy for matching customer records to watch list records for Oracle Financial Crime and Compliance Management Customer Screening Cloud Service.

Audience

This document is intended for users who are responsible for provisioning and activating Oracle FCCM Cloud services or for adding other users who would manage the services, or for users who want to develop Oracle Cloud applications.

Help

Use Help Icon  to access help in the application. If you don't see any help icons on your page, click your user image or name in the global header and select Show Help Icons. Not all pages have help icons. You can also access the https://docs.oracle.com/en/ to find guides and videos.

Related Resources

For more information, see these Oracle resources:

- Oracle Public Cloud: http://cloud.oracle.com
- Community: Use https://community.oracle.com/customerconnect/ to get information from experts at Oracle, the partner community, and other users.
- Training: Take courses on Oracle Cloud from https://education.oracle.com/oracle-cloud-learning-subscriptions.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

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 customer access to and use of Oracle support services will be pursuant to the terms and conditions specified in their Oracle order for the applicable services.

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.

Comments and Suggestions

Please give us feedback about Oracle Applications Help and guides! You can send an e-mail to: https://support.oracle.com/portal/.

1

Introduction

Oracle Financial Crime and Compliance Management Cloud Service makes use of the Cloud Matching Service, which provides a flexible and customizable strategy for matching customer records to watch list records. Sanctions screening typically requires the business to employ tightly-defined, zero-tolerance matching policies that will identify every possible match against a sanctions list. In these cases, the additional review work of lower probability matches will be necessary.

By contrast, a business carrying out PEP screening may choose a strategy of finding and investigating only the most likely matches against the PEP list, and the additional work required to confirm or eliminate weaker matches may not be cost-effective for the business.

Oracle Financial Crime and Compliance Management employs matching rules widget to configure the rules for screening.

These can be enabled and disabled as needed, to tune the behavior of Oracle Financial Crime and Compliance Management Cloud Service to your requirements. The matching rules are built around name matching.

Other identifiers are also used in the matching rules, but their main purpose is to rank matches by strength, and thereby to enable a most-likely approach to review potential matches.

For example, strong matches to Sanctions lists should be regarded as the most urgent matches, requiring immediate attention. Strong matches to PEP records will require follow-up, but may not be so urgent. Looser matches to PEP records may not be worth the time and operational cost of review.

In general, the looser the match rule, the more likely it is to raise false positives. It is not possible to eliminate all false positives, especially if there is a requirement to identify all true matches.

Tuning the matching strategy is, therefore, a trade-off between the proportion of true matches that are not detected and the work required to manually eliminate false positives. This will be evident in the examples in this document.

2
Matching

This chapter explains the matching logic and related sections.

Topics:

- Introduction
- How It Works
- Date Matching
- Processing Null Values

2.1 Introduction

This section provides overview of the matching.

Each match rule defines a set of criteria, specified as comparisons, that the pair of records must satisfy in order to qualify as a match under that rule. The match rule also defines a decision to be applied to any records which satisfy the conditions of the rule. Most rules have a Review decision, meaning matches that hit the rule need to be reviewed. However, there are also elimination rules, where if the records being compared meet the rule's criteria, a No Match decision is reached, and the two records will not be considered a match.

The rules are applied as a decision table, so if a pair of records qualifies as a match under a rule higher in the table, it will not be compared using any rules below that. All rules are configured to operate on a case-insensitive basis. Unless stated otherwise, all noise and whitespace characters are removed or normalized before matching.

Matches are generated based on a defined set of attributes for each rule. A weighted average of the score is generated for each of the attribute level matches.

There are two types of Matching services:

- Real-Time Processing
- Batch Processing

> **Note**
>
> Source for this document means the customer dataset.
> Target for this document means the watchlist.

In Real-Time processing, the values in the respective tags of candidate requests, are matched against a column in the target table. The screening application explicitly passes the strings as values in the request which forms "the strings to be matched" against "all the values in a column name". Then, based on the matches received for the source string from search engine, the score and the feature vector for the matched strings (source and target) are generated. Scores which exceed the configured thresholds are taken and collected.

Table 2-1 Match Types Descriptions and Examples

Ruleset Name	Description	Example
Exact	Considers two values and determines whether they match exactly. Applies only if Exact Match is selected. It does not apply when using Fuzzy Match.	If the source attribute is “John smith” and target attribute is “John smith”, then the match is an exact match.
Character Edit Distance (CED)	Considers two String tokens and determines how closely they match each other by calculating the minimum number of character edits (deletions, insertions and substitutions) needed to transform one value into the other.	For entities, stop words are not considered. • If the source attribute is “John smith” and target attribute is “Jon smith”, then the CED is 1 since the letter 'h' is missing between the source attribute and target attribute. • If the entity names are Oracle Financial Corporation and Finance Orcl Pvt. Ltd., then only Oracle Financial and Finance Orcl are considered for matching as corporation, Pvt., and Ltd. are stop words. The CED for Orcl is 2 and CED for finance is 3, so the overall CED is 3.
Character Match Percentage (CMP)	Determines how closely two values match each other by calculating the Character Edit Distance between two String tokens and considering the length of the shorter of the two tokens, by character count.	If the source attribute is “John smith” and target attribute is “Jon smith”, then the CMP is calculated using the formula (length of shorter string – CED) * 100 / length of longer string. In this case, it is (9-1) * 100/8 = 77.77%.

Table 2-1 (Cont.) Match Types Descriptions and Examples

Ruleset Name	Description	Example
Word Edit Distance (WED)	Determines how well multi-word String values match each other by calculating the minimum number of word edits (word insertions, deletions and substitutions) required to transform one value to another.	If the source attribute is “John smith” and target attribute is “Jon smith”, then the WED is calculated by checking the number of words that did not match with the target words after allowing for character tolerance, which is the number of words in the source attribute that did not match the target attribute. For example, the source string is Yohan Russel Smith and target string is Smith Johaan Rusel. First, we determine the CED for each word: • Yohan matches with Johann with a CED of 2. • Russel matches with Rusel with a CED of 1. • Smith matches with Smith with a CED of 0. If we consider a character tolerance of 1, we can observe the following: • Russel with a character tolerance of 1 match with Rusel. • Smith with a character tolerance of 0 matches with Smith. • Yohan with a character tolerance of 2 does not match with Johann as the character tolerance is 1. Based on these observations, we can conclude that one word does not match. This means that the WED is 1.
Word Match Percentage (WMP)	Determines how closely, by percentage, two multi-word values match each other by calculating the Word Edit Distance between two Strings and taking into account the length of the longer or the shorter of the two values, by word count.	The WMP is calculated using the formula (WMC/minimum word length) * 100. If the source attribute is “John smith” and target attribute is “Jon smith”, then the WMP is calculated as (2/5) * 100 = 40 %.

Table 2-1 (Cont.) Match Types Descriptions and Examples

Ruleset Name	Description	Example
Word Match Count (WMC	Determines how closely two multi-word values match each other by calculating the Word Edit Distance between two Strings and taking into account the length of the longer or the shorter of the two values, by word count.	The WMC is like WED, with the difference being that WMC gives the number of matches between 2 words and WED gives the number of words that did not match between 2 words. If the source attribute is “John smith” and target attribute is “Jon smith”, then the WMC is 2 as two words have matched (allowing for the character tolerance).
Exact String Match	Considers two String values and determines whether they match exactly.	
Abbreviation	Checks if the first character matches with the first character of source and target values.	
Starts With	Compares two values and determines whether either value starts with the whole of the other value. It therefore matches both exact matches and matches where one of the values starts the same as the other but contains extra information.	
Jaro Winkler or Reverse Jaro Winkler	The Jaro Winkler similarity is the measure of the edit distance between two strings. Click here for more information. In the Reverse Jaro Winkler, matches are generated even if the string is reversed. For example, if the source string is Mohammed Ali and the target string is Ali Mohammed, then the similarity = 1.	If the source string is Mohammed Ali and the target string is Mohammed Ali, then the similarity = 1.
Levenshtein	The Levenshtein Distance (LD) or edit distance provides the distance, or the number of edits (deletions, insertions, or substitutions) needed to transform the source string into the target string. Click here for more information.	For example, if the source string is Mohamed and the target string is Mohammed, then the LD = 1, because there is one edit (insertion) required to match the source and target strings.

2.2 How It Works

This section explains how matching works.

Consider the following match details:

- Evaluation Logic-> 46,4,26
- Evaluation score -> 0.88

- Word Match count (WMC_1) >= 2
- Abbreviated and CMP >=66

If the first name in **Mapping Source Attribute** column and **Target Attribute** column matched with a score of 0.88 and the Set threshold value is 0.75. The first name score is greater than the threshold value. Then 0.88x100x weightage (0.8) is provided in matching rule which gives you the score -> 70.4.

If **Mapping Source Attribute** column and **Target Attribute** column has the city data and the score value does not cross the provided threshold then it will not contribute to the score. Score is still 70.4

If city data cross the provided threshold, then it will contribute to the score as 100 (exact match). Then 100x0.05x weightage (0.05) => 5 is provided in matching rule which gives you the score 75.8.

If **Mapping Source Attribute** column and **Target Attribute** column has no city data, then score will cross the provided threshold. Then 50x0.05 => 2.5 is provided in matching rule which gives you the score 72.9.

Similarly, if other column data matches and score crosses the threshold, then the score will be added.

2.3 Date Matching

This section date matching details.

The following steps occurs in Date Matching:

- Derive Year of Birth (v_yob) from date of births attributes.
- Find minimum (v_min_yob) and maximum (v_max_yob) values from v_yob.
- Replace null/empty for v_min_yob with 9999 and v_max_yob with 1111.
- Match customer records against watchlist records within +/- 5 years range.
- For SAN records score is boosted if a match is within range of (v_min_yob - 5) to (v_max_yob + 5) years.
- For PEP/EDD records matches are eliminated if it is outside the range of (v_min_yob - 5) to (v_max_yob + 5) years.

Table 2-2 Example of Processing of Watchlist Records

V_DATE_OF_BIRTHS	V_YOB	V_MIN_YOB	V_MAX_YOB
1953-02-16	1953	1953	1953
1973-07	1973	1973	1973
1975-01-30	1975	1975	1975
(null)	(null)	9999	1111
1958-01-01;1952-12-31; 1956-10-28	1958;1952;1956	1952	1958

Table 2-3 Example of Date Matches

Source DOB	Target DOB	Match
1953-02-16	1953-01-01;1952-12-31;1956-10-28	True
1953-02-16	1958;1959	True
1953-02-16; 1954-02-16	1961-01-02	False
1953-02-16	Null	True
Null	1958-01-01	True
Null	Null	True

2.4 Processing Null Values

This section date matching details.

Null values are considered a way of hiding information. Whenever there is a Null value in any column in the Source or Target, the Matching Engine will give half the weightage for those attributes.

3
Scoring Methods

The Scoring Methods used in Customer Screening are as follows:

- Jaro Winkler
- Levenshtein
- Individual Name
- Individual SAN
- Individual PEP
- Individual EDD
- Entity Name
- Entity SAN
- Entity PEP
- Entity EDD

3.1 Jaro Winkler

This algorithm gives high scores for the following strings:

- The strings contain the same characters but within a certain distance from one another.
- The order of the matching characters is the same.

To be precise, the distance of finding a similar Character is one Character less than half of the length of the longest string. So if the longest string has a length of five, a character at the start of string 1 must be found before or on ((5/2)–1) ~ 2nd position in the string 2. This is considered a good match. Hence, the algorithm is directional and gives a high score if matching is from the beginning of the strings.

For example:

- `textdistance.jaro_winkler("mes", "messi") 0.86`
- `textdistance.jaro_winkler("crate", "crat") 0.96`
- `textdistance.jaro_winkler("crate", "atcr") 0.0`

In the first case, as the strings match from the beginning, a high score is given. Similarly, in the second case, only one Character was missing, and that is also at the end of string 2, a very high score is given. In the third case, the last two Characters of string 2 are rearranged by bringing them at the front, resulting in 0% similarity.

3.2 Levenshtein

The Levenshtein Distance (LD) or edit distance provides the distance or the number of edits (deletions, insertions, or substitutions) needed to transform the source string into the target string. For example, if the source string is Mohamed and the target string is Mohammed, the

LD = 1 because one edit (insertion) must match the source and target strings. For more information, see the Website.

3.3 Individual Name

This is a bespoke rules-based algorithm that has been optimized for determining individual name matches. Generation of the final matching score for Individual Name is based on the combination of scores generated by the following feature vector in the scoring method:

- **Abbreviation**: Checks if the first Character matches with the first Character of source and target tokens.

  Example:

  - **String 1**: "S Turner"
  - **String 2**: "Steve Turner"

  String1 is an abbreviated string for string2.

- **Character Edit Distance (CED)**: The Character Edit Distance comparison compares two String tokens and determines how closely they match each other by calculating the minimum, and the maximum number of character edits (deletions, insertions, and substitutions) needed to transform one value into the other.

  It compares each token in a string with each token of another string and finds the minimum edits we need to convert one token to another. Maximum CED score that is required to convert one token to another token in a string.

  The stopwords for Individual Names are Mr., Mrs, and Ms.

  Example:

  - **String1**: “Mr Jerrod Benito Carrera”
  - **String2**: “JOSE BENITO CABRERA”
  - **Stopword**: Mr.

  Jerrod matches with JOSE à CED: 5 (CED_MAX))

  Benito matches with BENITO à CED: 0 (CED_MIN)

  Carrera matches with CABRERA à CED: 1

- **Character Match Percentage (CMP)**: The Character Match Percentage comparison determines how closely two values match each other by calculating the Character Edit Distance between two String tokens and taking into account the shorter length of the two tokens by character count.

  CMP = (MCL – CED) * 100 / MCL

  - **CMP** = Character match percentage
  - **MCL** = Maximum character Length

- **Exact String Match**: The Exact String Match comparison is a simple comparison that determines whether or not two String values match. Checks if all the tokens are exact to each other. It can be in any order.

  Example:

  - **String 1**: Ram Lakshaman
  - **String 2**: Lakshaman Ram
  - **String 3**: Ram Lakshaman

String 1, String 2, and string 3 are exact.

- **Starts With**: The Starts With comparison compares two values and determines whether either value starts with the whole of the other value. It, therefore, matches both exact matches and matches where one of the values starts the same as the other but contains extra information.

  Checks if all the tokens start with the respective token in the target tokens.

  > **Note**
  >
  > Whichever token is smaller (either in source or target), that token will be considered compared with the other token (of longer length). It should be in order.

- **Starts With First**: It is similar to Starts With. Starts with the first token only.
- **Metaphone**: Checks if strings sound like. It is similar for hearing, but spelling may be different. It encodes a string into a double Metaphone value.
- **Tokenize Jaro**: Checks similarity is the measure between two strings. It tokenizes source and target strings, then uses the Jaro Winkler algorithm to calculate the score between tokens, and then consolidates the scores to a single score by taking the average.
- **Word Edit Distance (WED)**: The Word Edit Distance comparison determines how well multi-word String values match each other by calculating the minimum number of word edits (word insertions, deletions, and substitutions) required to transform one value to another.

  So WED is similar to CED, where instead of character edits, we find the word edits.

  In WED, we have an additional parameter called "character tolerance." Character tolerance allows the user to have a character tolerance in words, i.e. how many Character edits can it allow in each token for one token to match another one.

  WED in simple words is: Number of words that did not match with the target words (after allowing the character tolerance)

  Example:

  – **String 1**: "Yohan Russel Smith"
  – **String 2**: "Smith Johaan Rusel"

  Yohan matches with Johann - CED: 2

  Russel matches with Rusel - CED: 1

  Smith matches with Smith - CED: 0

  – If we have a character tolerance of "1,". The number of WED will be: 1
    * Russel, with a character tolerance of 1 matches with Rusel.
    * Smith with character tolerance of 0 matches with Smith.
    * Yohan with character tolerance of 2 does not match with Johann as character tolerance is 1.

    One token did not match. WED = 1.
  – If we have a character tolerance of "2,". The number of WED will be : 0
    * Russel, with a character tolerance of 1 matches with Rusel.
    * Smith with character tolerance of 0 matches with Smith.
    * Yohan, with a character tolerance of 2, does match with Johann.

All tokens matched. WED = 0.

- **Word Match Percentage (WMP)**: The Word Match Percentage comparison determines how closely two multi-word values match each other by calculating the Word Edit Distance between two Strings and considering the length of the longer or longer or longer, the shorter of the two values, by word count.

  In mathematical terms, the Word Match Percentage comparison uses the following formula to calculate its results as the Number of tokens matched:

  WMP = (WMC / WL) *100

  – **WMP**: Word Match Percentage

  – **MWL**: Maximum Word Length (i.e., the maximum number of words in the two values being compared)

  – **WED**: Word Edit Distance between two String values, and

  – **WL**: Minimum Word Length, relating to shorter input option

- **Word Match Count (WMC)**: The Word Match Percentage comparison determines how closely two multi-word values match each other by calculating the Word Edit Distance between two Strings and taking into account the length of the longer or, the longer shorter of the two values, by word count.

  WMC, in simple words, is the Number of words that did match with the target words (after allowing the character tolerance).

  WMC compliments WED, as WMC gives the number of matches between 2 words and WED gives the number of mismatches between 2 words.

  > **Note**
  >
  > Based on the Year of birth, if the match falls beyond +/- 5 years, then the match is eliminated. This is applicable for the Individual PEP and the EDD records.

  Example 1:

  – **String 1**: "Yohan Russel Smith"

  – **String 2**: "Smith Johaan Rusel"

    Yohan matches with Johann - CED: 2

    Russel matches with Rusel - CED: 1

    Smith matches with Smith - CED: 0

  – If we have a character tolerance of "1,". The number of WED will be: 2

    * Russel, with a character tolerance of 1 matches with Rusel.

    * Smith with character tolerance of 0 matches with Smith.

    * Yohan with character tolerance of 2 does not match with Johann as character tolerance is 1.

    One word did not match. So WMC = 2.

  – If we have a character tolerance of "2,". The number of WMC will be : 3

    * Russel, with a character tolerance of 1 matches with Rusel.

    * Smith with character tolerance of 0 matches with Smith.

* Yohan, with a character tolerance of 2, does match with Johann.

All tokens matched. WMC = 3.

–

Example 2:

– **String 1**: “Mr Jerrod Benito Carrera”

– **String 2**: “JOSE BENITO CABRERA”

– **Stopword**: Mr. (It will not be considered for calculating the scrore)

The individual feature vector scores with final scrore for the match:


```
{"ced_list": [5, 0, 1], "ced_min": 0, "ced_max": 5, "cmp": 68.42105,
"wed_1": 1, "wed_2": 1, "wmc_1": 2, "wmc_2": 2, "wmp_1": 66.0, "wmp_2":
66.666664, "metaphone": 1, "starts_with": 0, "abbreviation": 1,
"tokenize_jaro": 0.8301587, "exact": 0, "inorderMaxPos": 0, "score": 0.88}
```


Figure 3-1 Individual Name Score

	Mr Jerrod Benito Carrera	JOSE BENITO CABRERA		
CED	Jerrod	Jose	5	
	Benito	BENITO	0	
	Carrera	CABRERA	1	
	Benito	BENITO	CED_MIN	0
	Jerrod	Jose	CED_MAX	5
			CMP	68.42105
			WED, tolerance=1	1
			WED, tolerance=2	1
			WMC_1	2
			WMC_2	2
			WMP_1	66
			WMP_2	66.66666
	Benito	Benito	Metaphone:	1
	JBC	JBC	Abbreviation	1
			Starts With	0
			Exact	0
			Tokenize_jaro	0.830159
			Individual Name score:	**0.88**

3.4 Individual SAN

This is a bespoke rules-based algorithm that has been optimized for determining Individual Sanctions name matches. The process and methods to generate feature vector are similar to Individual Name. For more details, see Individual Name.

3.5 Individual PEP

This is a bespoke rules-based algorithm that has been optimized for determining Individual PEP name matches. The process and methods to generate feature vector are similar to Individual Name. For more details, see Individual Name.

3.6 Individual EDD

This is a bespoke rules-based algorithm that has been optimized for determining Individual EDD name matches. The process and methods to generate feature vector are similar to Individual Name. For more details, see Individual Name.

3.7 Entity Name

This is a bespoke rules-based algorithm that has been optimized for determining organization name matches. For example, BLUE SKYE COLLECTIONS LTD.

The process and methods to generate feature vector are similar to **Individual Name**. For more details, see **Individual Name**.

The process to derive the final score from the feature vector is different from Individual Name. The stopwords are not considered for computing the score. The stopwords for Entity Name are different from Individual Name.

Example:

- **String 1**: "BLUE SKYE COLLECTIONS LTD"
- **String 2**: "BLUE SKY THREE LTD"
- **Stopword**: LTD (It will not be considered for calculating the scrore)

The individual feature vector scores with overall scrore for the match:

```
{"ced_list": [0, 1, 9, 0], "ced_min": 0, "ced_max": 9, "cmp": 54.545456,
"wed_1": 1, "wed_2": 1, "wmc_1": 3, "wmc_2": 3, "wmp_1": 75.0, "wmp_2": 75.0,
"metaphone": 3, "starts_with": 0, "abbreviation": 0, "tokenize_jaro":
0.8026768, "exact": 0, "inorderMaxPos": 0, "score": 0.95}
```

Figure 3-2 Individual Name Score

	Mr Jerrod Benito Carrera	JOSE BENITO CABRERA		
CED	Jerrod	Jose	5	
	Benito	BENITO	0	
	Carrera	CABRERA	1	
	Benito	BENITO	CED_MIN	0
	Jerrod	Jose	CED_MAX	5
			CMP	68.42105
			WED, tolerance=1	1
			WED, tolerance=2	1
			WMC_1	2
			WMC_2	2
			WMP_1	66
			WMP_2	66.66666
	Benito	Benito	Metaphone:	1
	JBC	JBC	Abbreviation	1
			Starts With	0
			Exact	0
			Tokenize_jaro	0.830159
			Individual Name score:	**0.88**

3.8 Entity SAN

This is a bespoke rules-based algorithm that has been optimized for determining Sanctions organization name matches. The process and methods to generate feature vector are similar to Individual Name. For more details, see Individual Name.

3.9 Entity PEP

This is a bespoke rules-based algorithm that has been optimized for determining PEP organization name matches. The process and methods to generate feature vector are similar to Individual Name. For more details, see Individual Name.

3.10 Entity EDD

This is a bespoke rules-based algorithm that has been optimized for determining Entity EDD organization name matches. The process and methods to generate feature vector are similar to Individual Name. For more details, see Individual Name.

4

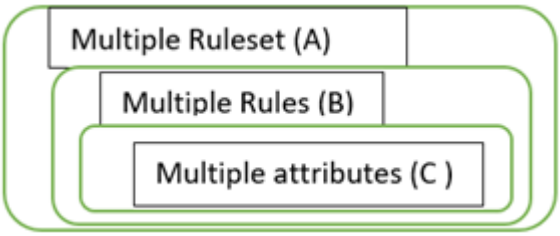
Rulesets

The Ruleset facilitates identifying the similarity between two entities (customer, account, and so on) and derives a match. A Ruleset is a set of rules applied to the defined source and target entities and compares the entities' attributes to derive a match. OFS Cloud Service provides ready-to-use rulesets; however, you can modify these rulesets or create your own rulesets.

The process of creating Match Rule (s) is as follows:

> **Note**
>
> The sum of weightage for mappings and groups for each rule must be 1.

Each Ruleset comprises multiple rules. The Ruleset compares the attributes defined in the rules for the source entity with the target entity and applies one of the match scoring mechanisms described in this document. The threshold value is defined in each level.



- Ruleset (A): All the matches from the rules should cross this threshold.
- Rule (B): All the attribute thresholds should cross this threshold.
- Attribute (C): Based on the match, it checks across the threshold. Once it crosses the threshold, the record will be considered.

4.1 Out of the Box Rulesets

This topic provides the details of Out of the Box (OOB) match rulesets.

4.1.1 Match Rules

Each Ruleset contains the pre-defined source and target node types. Each Ruleset compares the parameters/attributes of the source and target node types to obtain a match.

Table 4-1 List of Rulesets

Pipeline Name	Ruleset Name
Individual Batch Screening	Customer to SAN Watchlist Individual Match Customer to PEP Watchlist Individual Match Customer to EDD Watchlist Individual Match Prohibited Country for Individual
Entity Batch Screening	Customer to SAN Watchlist Entity Match Customer to PEP Watchlist Entity Match Customer to EDD Watchlist Entity Match Prohibited Country for Entity
Individual Real Time Screening	Candidate to SAN Watchlist Individual Match Candidate to EDD Watchlist Individual Match Candidate to PEP Watchlist Individual Match Country Prohibition Matching Rule for Individual
Entity Real Time Screening	Customer to SAN Watchlist Entity Match Customer to EDD Watchlist Entity Match Customer to PEP Watchlist Entity Match Country Prohibition Matching Rule for Entity
Individual 314 A Batch Screening	Customer to 314a Watchlist Individual Match
Entity 314 A Batch Screening	Customer to 314a Watchlist Entity Match
External Entities 314A Transactions Batch Screening	External Entity to 314a Watchlist Match

> **Note**
>
> Out of the Box Rulesets are not customizable.

4.2 Modifying the Ruleset

Each Ruleset comprises multiple rules. The Ruleset compares the attributes defined in the rules for the source with the target. For example, First Name to Given Name, Last Name to Family Name, Date of Birth to Date of Birth, and so on.

Every Entity has attributes such as date of birth, jurisdiction, and so on. To derive a match you must apply the conditions for these attributes, such as match type, scoring method, threshold score, and weightage.

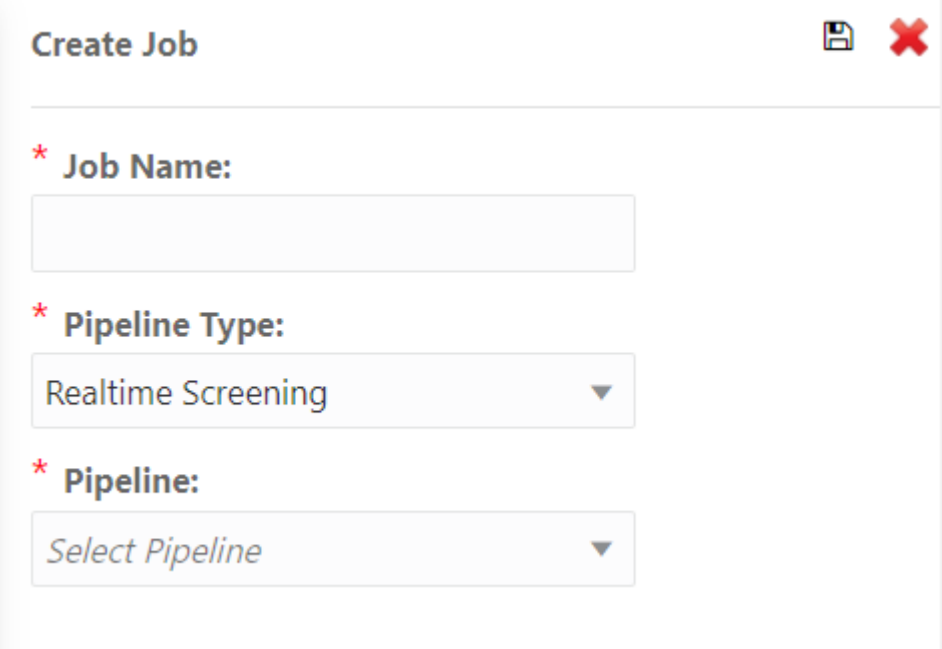
Follow these steps to create a customized ruleset:

1. To customize a ruleset, copy the pipeline using the Copy icon and give it a new name.

Figure 4-1 Copy icon

2. Once you have created a new pipeline, you can customize the ruleset and save it. Save the pipeline.
3. From the navigation list, go to Batch Administration → Jobs.
4. Create a new Job and give a Job Name. Enter the Pipeline type: Individual Screening/ Batch Screening. Once you have entered the Pipeline type, mention the new Pipeline Name in Pipeline.

Figure 4-2 Creating a new Job for using a modified Pipeline for Matching



5. Save the Job.

5
Data Processing

This chapter describes the components of Data Processing.

5.1 Transliteration

Transliterating a word does not tell you the meaning of the word. It tells you how the how the word is pronounced in a foreign language. This makes the language a little more accessible to people who are unfamiliar with the alphabet of the foreign language. This is opposed to translation, which, put in simple terms, gives you the meaning of a word that's written in another language. For example, the greeting in Arabic is translated as greeting in Arabic in the Arabic script but is transliterated in the Latin script as *shukraan*.

General transforms provide a general-purpose package for processing Unicode text. They are a powerful and flexible mechanism for handling a variety of different tasks, including:

• Uppercase, lowercase, or title-case conversions

• Normalization

• Hex and character name conversions

• Script-to-script conversion

The reference data sources supported by Customer Screening are all provided in the Latin character set, and some in the original scripts. The screening process can also be used with non-Latin data. Non-Latin data can be screened against the Latin reference data sources which are supported by performing transliteration of data from the non-Latin character set to the Latin character set.

Non-Latin customer data can be screened against non-Latin reference data without any changes to the product, although certain fuzzy text matching algorithms may not be as effective when used to match data with the non-Latin character set. Text is processed on a left-to-right basis.

5.1.1 Original Script Matching

To match the original script data against reference data, prepare customer and external entity data such that non-Latin names are populated in the Original Script Name fields.

5.1.2 Input Fields for Individual Screening

This section lists the REST input fields used when screening individuals via the real-time process. The following input attributes are available for the individual screening process. They are available for any additional inputs required by your screening process.

Table 5-1 Input Fields for Individual Screening

Field Name	Expected Data Format
v_given_name	String

Table 5-1 (Cont.) Input Fields for Individual Screening

Field Name	Expected Data Format
v_family_name	String
v_full_nm	String
v_aliases_family_name	String
v_aliases_given_name	String
v_aliases	String

> **Note**
>
> The Individual Matching Process is based primarily on the name supplied for the individual.

5.1.3 Input fields for Entity Screening

This section lists the REST input fields used when screening entities via the real-time process. The following input attributes are available for the Entity Screening Process. They are available for any additional inputs required by your screening process.

Table 5-2 Input fields for Entity Screening

Field Name	Expected Data Format
v_org_nm_bus_strip	String
v_last_nm	String
v_full_nm	String
v_org_nm	String
v_alias_nm	String
v_aliases	String
v_first_nm	String

5.2 Analyzers

The Analyzer assists compliance teams and technical users in understanding how specific input data—such as names or addresses—is tokenized, normalized, and matched against watchlist entries. It enables detailed examination of synonym expansion, token classification, and match scoring mechanisms that are essential for fine-tuning the screening engine and ensuring accurate detection of true positives while minimizing false positives.

Configure Analyzer on Index Management

1. From the **Navigation List**, click **Watchlist Management**>**Index Management**. The index Management page displays a list of available watchlists.
2. Click Index JSON

Figure 5-1 Edit icon

to view the JSON file details associated with this watch list.
The **Edit Index JSON** pop-up appears.

3. You can edit the JSON in this window.
For example: If we want to use Country Synonyms for Country Name under Prohibited Country Watchlist, then edit the Country List's index json and change the `analyzerType` of particular attribute (`"name" : "v_country_name"`) to address.

```
{  "schemaName" : "",  "runSkey" : 68,  "batchRunId" :
"WLDJWLoad_2024-03-28_1711619630679_1",  "tableName" : "FCC_WL_DJW_V",
"deletedProfilesTableName" : null,  "filterCondition" : "1=1",
"indexName" : "fcc_idx_djw",  "indexAlias" : "idx_djw",
"disasterRecovery" : false,  "indexLogicalName" : "Watchlist",
"indexBusinessName" : "Dow Jones",  "indexKeyAttribute" : "n_uid",
"loadType" : "FullLoad",  "shards" : 3,  "replicas" : 4,  "attributes" :
[ {    "name" : "v_given_name", -- Each of these blocks is used to define
     pre-processing of the fields.    "type" : "text",    "similarity" :
"boolean",    "analyzerType" : "name", -- Here, the user can enter any
analyser mentioned in the
     list.    "searchAnalyzerType" : null,    "fields" : [ ],
"termVector" : null  }, {    "name" : "v_family_name",    "type" :
"text",    "similarity" : "boolean",    "analyzerType" : "namestop",
"searchAnalyzerType" : null,    "fields" : [ ],    "termVector" : null  },
{    "name" : "v_full_name",    "type" : "text",    "similarity" :
"boolean",    "analyzerType" : "namestop",    "searchAnalyzerType" :
null,    "fields" : [ ],    "termVector" : null  }, {    "name" :
"v_aliases_given_name",    "type" : "text",    "similarity" :
"boolean",    "analyzerType" : "namestop",    "searchAnalyzerType" :
null,    "fields" : [ ],    "termVector" : null  }, {    "name" :
"v_aliases_family_name",    "type" : "text",    "similarity" :
"boolean",    "analyzerType" : "name",    "searchAnalyzerType" : null,
"fields" : [ ],    "termVector" : null  }, {    "name" :
"v_entity_name",    "type" : "text",    "similarity" : "boolean",
"analyzerType" : "organization",    "searchAnalyzerType" : null,
"fields" : [ ],    "termVector" : null  }, {    "name" :
"v_entity_name_bus_strip",    "type" : "text",    "similarity" :
"boolean",    "analyzerType" : "organization",    "searchAnalyzerType" :
null,    "fields" : [ ],    "termVector" : null  }, {    "name" :
"v_original_script_name",    "type" : "text",    "similarity" :
"boolean",    "analyzerType" : "name",    "searchAnalyzerType" : null,
"fields" : [ ],    "termVector" : null  }, {    "name" :
"v_date_of_births",    "type" : "text",    "similarity" : "boolean",
"analyzerType" : "date",    "searchAnalyzerType" : null,    "fields" :
[ ],    "termVector" : null  }, {    "name" : "v_passports",    "type" :
"text",    "similarity" : "boolean",    "analyzerType" : "name",
"searchAnalyzerType" : null,    "fields" : [ ],    "termVector" : null  },
{    "name" : "v_ssn",    "type" : "text",    "similarity" : "boolean",
"analyzerType" : "name",    "searchAnalyzerType" : null,    "fields" :
[ ],    "termVector" : null  }, {    "name" :
```

```
"v_identification_numbers",    "type" : "text",    "similarity" :
"boolean",    "analyzerType" : "name",    "searchAnalyzerType" : null,
"fields" : [ ],    "termVector" : null  }, {    "name" : "v_city",
"type" : "text",    "similarity" : "boolean",    "analyzerType" :
"address",    "searchAnalyzerType" : null,    "fields" : [ ],
"termVector" : null  }, {    "name" : "v_country",    "type" : "text",
"similarity" : "boolean",    "analyzerType" : "address",
"searchAnalyzerType" : null,    "fields" : [ ],    "termVector" : null  },
{    "name" : "v_nationality",    "type" : "text",    "similarity" :
"boolean",    "analyzerType" : "address",    "searchAnalyzerType" :
null,    "fields" : [ ],    "termVector" : null  }, {    "name" :
"v_residence",    "type" : "text",    "similarity" : "boolean",
"analyzerType" : "address",    "searchAnalyzerType" : null,    "fields" :
[ ],    "termVector" : null  }, {    "name" : "v_yob",    "type" :
"text",    "similarity" : "boolean",    "analyzerType" : "name",
"searchAnalyzerType" : null,    "fields" : [ ],    "termVector" : null  },
{    "name" : "v_min_yob",    "type" : "integer",    "similarity" :
"boolean",    "analyzerType" : null,    "searchAnalyzerType" : null,
"fields" : [ ],    "termVector" : null  }, {    "name" : "v_max_yob",
"type" : "integer",    "similarity" : "boolean",    "analyzerType" :
null,    "searchAnalyzerType" : null,    "fields" : [ ],    "termVector" :
null  }, {    "name" : "v_address",    "type" : "text",    "similarity" :
"boolean",    "analyzerType" : "address",    "searchAnalyzerType" :
null,    "fields" : [ ],    "termVector" : null  }, {    "name" :
"v_aliases",    "type" : "text",    "similarity" : "boolean",
"analyzerType" : "namestop",    "searchAnalyzerType" : null,    "fields" :
[ ],    "termVector" : null  }, {    "name" : "v_gender",    "type" :
"text",    "similarity" : "boolean",    "analyzerType" : "gender",
"searchAnalyzerType" : null,    "fields" : [ ],    "termVector" : null  },
{    "name" : "v_place_of_birth",    "type" : "text",    "similarity" :
"boolean",    "analyzerType" : "address",    "searchAnalyzerType" :
null,    "fields" : [ ],    "termVector" : null  }, {    "name" :
"v_title",    "type" : "text",    "similarity" : "boolean",
"analyzerType" : "gender",    "searchAnalyzerType" : null,    "fields" :
[ ],    "termVector" : null  } ],  "customAnalyzer" : [ ],
"customFilter" : [ ],  "customCharFilter" : [ ],  "customTokenizer" :
[ ],  "others" : [ "n_wl_skey", "n_run_skey", "v_wl_sub_type", "v_wl_type",
     "v_entity_type", "n_uid" ],  "replaceEmptyFields" : [ ],
"replaceCharFields" : [ {    "name" : "v_full_name",    "charArray" :
[ "-.", "'" ],    "replaceWith" : [ " ", "" ]  }, {    "name" :
"v_family_name",    "charArray" : [ "-.", "'" ],    "replaceWith" :
[ " ", "" ]  }, {    "name" : "v_given_name",    "charArray" : [ "-.",
"'" ],    "replaceWith" : [ " ", "" ]  }, {    "name" :
"v_entity_name",    "charArray" : [ "-.", "'", "&", "()" ],
"replaceWith" : [ " ", "", "and", " " ]  }, {    "name" :
"v_entity_name_bus_strip",    "charArray" : [ "-.", "'", "&", "()" ],
"replaceWith" : [ "", "", "and", " " ]  }, {    "name" :
"v_original_script_name",    "charArray" : [ "-.", "'" ],
"replaceWith" : [ " ", "" ]  }, {    "name" : "v_country",
"charArray" : [ "-.", "'" ],    "replaceWith" : [ " ", "" ]  },
{    "name" : "v_address",    "charArray" : [ "-.", "'" ],
"replaceWith" : [ " ", "" ]  }, {    "name" : "v_aliases_given_name",
"charArray" : [ "-.", "'" ],    "replaceWith" : [ " ", "" ]  },
{    "name" : "v_aliases_family_name",    "charArray" : [ "-.", "'" ],
"replaceWith" : [ " ", "" ]  }, {    "name" : "v_aliases",
"charArray" : [ "-.", "'" ],    "replaceWith" : [ " ", "" ]  } ],
```

```
"translateFields" : [ "v_family_name", "v_given_name", "v_full_name",
      "v_aliases_family_name", "v_aliases_given_name", "v_aliases" ]}}
```

4. Click **Validate** to verify that your edits to the JSON are valid.
5. Click **Save** to update the JSON or click **Cancel** to close without saving your changes.

Analyzer Types

Table 5-3 Analyzer Types

Analyzer Type	Supported Filters	Type	Description
Name	Individual Name Synonyms Individual Title	Synonym Stop Word	The Name analyzer processes person names by applying standardization rules such as name synonyms (e.g., Will - William) and removal of non-essential titles (e.g., Mr., Dr.). This ensures better match results during screening.
Address	Country Synonyms	Synonym	The Address analyzer processes address components by resolving country-specific synonyms (e.g., USA - United States, , UK - United Kingdom)). This enhances consistency and accuracy in location-based matching. **Example**: Input: `123 Main St, NY,USA` Normalized to: `123 Main Street, New York, United States`
Phone	No Token Filters	-	The Phone Analyzer tokenizes and indexes phone numbers without applying any additional filters. It enables straightforward and direct matching of phone number values during screening.
Email	No Token Filters	-	The Email Analyzer processes email addresses as exact tokens without applying any transformations or filters. It is designed for direct string matching, ensuring that the full email address is preserved for accurate comparison.

Table 5-3 (Cont.) Analyzer Types

Analyzer Type	Supported Filters	Type	Description
Organization	Organization Numbers Organization Suffix Organization Strip Words	Synonym Stop Word Stop Word	The Organization analyzer standardizes organization names by removing common suffixes (e.g., Inc, Ltd), normalizing common terms, and ignoring non-distinct or generic words. This improves matching for corporate entity names.
Gender	Individual Gender	Synonym	The Gender analyzer handles gender-related fields by resolving known synonyms (e.g., F - Female, M - Male) for consistent identity matching.
Date	No Token Filters	-	Dates are indexed as-is with no transformation or filtering.
Name Stop	Individual Name Synonyms Individual Title Individual Name Strip Words	Synonym Stop Word Stop Word	The Name Stop Analyzer clean and normalize names by removing non-essential tokens and standardizing known variations. This helps improve match accuracy during screening. **Example:** **Name** : `Mr. John A. Smith` The Name Stop Analyzer: • Removes the title “Mr.” using the Individual Title filter. • Removes initials like “A.” if listed in *Strip Words*. • Recognizes that “John” is a synonym for “Jon” if defined in *Individual Name Synonyms*. `John Smith` - normalized for better matching

Table 5-3 (Cont.) Analyzer Types

Analyzer Type	Supported Filters	Type	Description
TF Analyzer	No Token Filters	-	The TF Analyzer is used in Transaction Filtering to tokenize and normalize input data by applying filters like lower casing, stop word removal, and synonym resolution for improved match accuracy.
Document ID	No Token Filters	-	A custom analyzer that tokenizes text using delimiters (comma, semicolon, Tilde, Parentheses, and space), converts tokens to lowercase, and removes duplicates. This analyzer is intended for indexing document identifiers such as national ID numbers or other personal identification numbers.
Organization Strip	Organization Numbers Organization Stop Words	Synonym Stop Word	The Organization Strip Analyzer is designed to strip common suffixes or terms from organization names that don't add unique value to the name, improving match consistency across similar entries. **Example**: Consider the organization name: `ABC Technologies Pvt. Ltd`. The Organization Strip Analyzer: • Removes stop words like “Pvt.” and “Ltd.” • May replace common synonyms like “Technologies” with “Tech” if configured. **Processed Result**:`ABC Tech` This helps normalize similar entries like ABC Technologies Limited and ABC Tech Ltd. to a comparable form for accurate screening.

Table 5-3 (Cont.) Analyzer Types

Analyzer Type	Supported Filters	Type	Description
Alphanumeric Keyword	No Token Filters	-	A custom analyzer that splits text on any sequence of non-letter and non-digit characters, converts all characters to lowercase, and applies ASCII folding to normalize accented or special characters to their ASCII equivalents.

5.2.1 Synonyms

There can be multiple ways to coin the spelling of a name of an entity or an individual. Even country names have synonyms.

Follow these steps to add a Synonym:

1. From the Navigation List go to Watchlist Management.
2. Go to Synonym Management.
3. Select the Synonym type from the available categories.
4. Click on the Plus "+" button.
5. Enter the values separated by commas (,).

Table 5-4 Examples of Synonyms

Names	Synonyms
Ayssa	Eisa, Isa, Issa
Mohammad	Mohammed, Muhammed
Iran	Persia

> **Note**
>
> All the synonyms are treated equal and that means if you try to match Ayssa from the source with Issa from the target, you will get an exact match.

5.2.2 Stopwords

The stopwords are not considered for computing the score. The stopwords for Entity Name are different from Individual Name.

For example: Mr. Ian Smith (here the word Mr. can affect your match score).

5.3 Treating Special and Accented Characters

The source and watchlist may have some names with special characters based on the geography.

For example: Nsĩã´, Dúkũ, kwán, Nyamékyɛ, Obím`pɛ́.

> **Note**
>
> The hyphen (-), dot (.), apostrophe ('), ampersand (&), tilde (~), comma (,), semicolon (;), parentheses (()) along with all accented characters are the special characters that are supported as part of the matching Engine. These characters are replaced by white spaces.

The system supports the following accented characters to ensure proper matching and processing of international names and terms:

a, á, b, c, č, d, ď, e, é, ě, f, g, h, ch, i, í, j, k, l, m, n, ň, o, ó, p, (q), r, ř, s, š, t, ť, u, ú, ů, v, (w), (x), y, ý, z, ž.