

IDOL PHI Package

Software Version 12.7

Technical Note



Document Release Date: October 2020
Software Release Date: October 2020

Legal notices

Copyright notice

© Copyright 2020 Micro Focus or one of its affiliates.

The only warranties for products and services of Micro Focus and its affiliates and licensors (“Micro Focus”) are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Micro Focus shall not be liable for technical or editorial errors or omissions contained herein. The information contained herein is subject to change without notice.

Documentation updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for updated documentation, visit <https://www.microfocus.com/support-and-services/documentation/>.

Support

Visit the [MySupport portal](#) to access contact information and details about the products, services, and support that Micro Focus offers.

This portal also provides customer self-solve capabilities. It gives you a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the MySupport portal to:

- Search for knowledge documents of interest
- Access product documentation
- View software vulnerability alerts
- Enter into discussions with other software customers
- Download software patches
- Manage software licenses, downloads, and support contracts
- Submit and track service requests
- Contact customer support
- View information about all services that Support offers

Many areas of the portal require you to sign in. If you need an account, you can create one when prompted to sign in. To learn about the different access levels the portal uses, see the [Access Levels descriptions](#).

Contents

Introduction	5
Data Sources	5
Names	5
Age	6
Dates	6
Postal Codes	6
Addresses and Locations	6
Telephone Number	7
Email Address	7
IP Address/URL	8
National Identification Number	8
Vehicle Identifiers	8
Medical	8
Profession	8
Unique Device Identifiers	9
Health Plan and Medical Record Numbers	9
Birth Certificate Numbers	9
Laboratory Numbers	9
DEA	9
Accounts	9
New in this Release	10
Resolved Issues	10
IDOL Eduction Grammars	11
Configure Post Processing	11
Configure Pre-Filtering	12
Entity Context	12
Balance Precision and Recall	13
Configure Tangible Characters	13
Customize Stop Lists	14
Eduction Grammar Reference	14
account.ecr	15
address.ecr	15
age.ecr	16
certificate.ecr	17
date.ecr	17

dea.ecr	18
device.ecr	18
healthplan.ecr	18
internet.ecr	19
laboratory.ecr	20
license.ecr	20
location.ecr	20
name.ecr	21
national_id.ecr	21
medical_terms.ecr	22
telephone.ecr	22
vehicle.ecr	22
Components	23
User Grammar Extensions	30
Medical Record Numbers	30
Generic License Numbers	31
Generic Certificate Numbers	31
PHI Grammar Customization	31
Example 1: New Street Address	32
Example 2: New Known City	33
Example 3: New Name and Custom Separator	34
Compile Custom Grammars	35
Modify Other Grammars and Entities	35
Validated ID Numbers	35
IDOL AgentBoolean IDX	37
Send documentation feedback	39

Introduction

The IDOL PHI Package contains tools that allow you to locate Protected Healthcare Information (PHI) in your data, to ensure compliance with regulations such as the *Standards of Privacy of Individually Identifiable Health Information* implemented as part of the Health Insurance Portability and Accountability Act (HIPAA).

The IDOL PHI Package has two types of tools:

- [IDOL Education Grammars](#) (.ecr and files). IDOL Education is a tool for finding entities (small pieces of information such as names and phone numbers) in text. Education grammars contain descriptions of the entities. In some cases, this might be a list of fixed values (such as names), and in others it might be pattern matching tools that find data of a particular type (such as a set of digits that make up a phone number). The Education grammars included in the IDOL PHI Package describe different kinds of personally identifiable information, so that you can find these in your data.
- [IDOL AgentBoolean IDX](#). IDOL AgentBoolean is a method of storing entities and querying for them that uses the IDOL Agentstore component (a specially configured IDOL Content component), rather than Education. The IDX files are index files that contain the details of the entities, which you can index into the IDOL Agentstore component.

Data Sources

The IDOL PHI Package contains a variety of different kinds of entities to describe healthcare information that is protected by regulations such as HIPAA. The following sections provide some information about how this information is compiled.

For each entity type, extensive testing is performed to ensure the precision and recall metrics are optimized. Many millions of examples are run through the package to test full coverage of the patterns and algorithms involved.

Names

An international database containing over 100 million individuals is analyzed to identify the structure and characteristics of names in each country. In doing so, extensive lists of the frequencies of occurrence of given names and family names are used to generate strong identification grammars for names.

In addition, rules are included to handle linguistic information, such as transliteration (for example, from the Cyrillic or Greek alphabets), or the use or removal of diacritic marks.

Age

The linguistic patterns of usage of both unstructured and semi-structured text are analyzed in all supported languages to determine the range of formats used to refer to a patient's age or age demographic. The resulting grammar establishes a confidence measure to distinguish references to age as opposed to other information, and includes all elements of dates that allow the determination of age.

Dates

A large corpus of documents from public sources is processed to analyze the occurrence and format of dates. In this way, coverage of all common and less-common formats is built up, while enabling a *likelihood* measure to indicate the confidence that the characters identified are a date of birth, rather than an unrelated date or other alphanumeric string.

Dates of any type that relate to an individual's healthcare, other than a single year, are covered by PHI regulation. The IDOL PHI Package allows determination of all such dates from analysis of linguistic patterns in all supported languages. In addition, the package can identify dates of particular types, such as date of death, and hospital admission and discharge dates.

Postal Codes

For each country, the publications of the national Postal Services are used as the authoritative source on the postal code.

In addition, testing against widely-gathered examples allows the identification and inclusion of non-standard formats and common errors (such as mixing the letter O with the digit 0), with an appropriately adjusted likelihood measure.

Addresses and Locations

The identification of addresses consists of a number of steps, each of which is used as additional evidence that a piece of text represents a postal address. These are:

1. The format of the text.
2. The house number / street-name portion.
3. The village / town / county / region portion.
4. The postal code.

These components are not necessarily always present for a particular address, but each is taken as evidence that the text does indeed contain an address, combining to form an overall likelihood.

- Few countries have prescribed formats for addresses, while most have conventions defined by the national Postal Service that is generally adhered to, but also frequently ignored.

The IDOL Web Connector is used to gather many millions of web documents to identify candidate addresses in each applicable country. From there, the variety of formats that are used in practice are identified. In addition, any recommendations published by the national Postal Services are also used.

- For the street-address portion, the extensive OpenStreetMap project is used, and a database of every named street in each of the supported countries is obtained and analyzed. From this database, rules are derived to allow the identification of the vast majority of street-address strings.
- The de facto authority for geographical place names is the GeoNames database, with 11 million locations identified by data including country, population and type. In particular the *type* field is used to generate complete lists of populated settlements and administrative regions (such as county / department / region) for the countries that frequently use those in addresses. In addition, the names are available in different character sets and transliteration schemes to ensure internationalization.
- The patterns derived for matching Postal Codes are also used here (see [Postal Codes, on the previous page](#)).

For locations, the IDOL PHI Package identifies any address portion smaller than a state.

Telephone Number

The general schemes for the creation of telephone numbers and fax numbers are readily available from the appropriate government department of each country. However, the formats of such numbers when written down varies considerably within a country, and even more so when numbers are referred to in a foreign document.

The strategy for creating comprehensive phone number matching grammars is centered on several key methods:

- Knowledge of the national scheme for assigning numbers.
- Databases of international and area codes in each country, obtained from authoritative sources.
- Analysis of many millions of examples of the usage of telephone numbers, obtained from a wide variety of public sources.

This final point is the most important. Only through examination of real-world usage of such numbers is the full range of formats obtained for each country.

The proximity of keywords indicating that the digits represent a telephone or fax number is used to strengthen the likelihood of the match.

Email Address

The IETF publications RFC 5321 and RFC 5322 define the standards of validity of email addresses, and so the IDOL PHI Package uses these for this purpose. In addition, it uses metrics of likelihood

derived from the analysis of the most common email domains, to allow the grammars to differentiate between likely email addresses and those that are unlikely but still valid (for example, example@example.test).

IP Address/URL

The formats of IP Addresses are defined by the IETF in RFC 791 (IPv4) and RFC 4291 (IPv6) with later modifications. These allow the location of potentially identifiable information in candidate text by the IDOL PHI Package.

In the same way, Uniform Resource Locators (URL) were defined in RFC 1738.

National Identification Number

Each country has a different scheme for the use of National Identification. For countries with National ID cards, the format of the number is derived from governmental sources. In other countries, the formats of National Health, National Social Security, or National Insurance numbers are obtained from governmental sites, with the exception of a few cases in which other sources are used.

Vehicle Identifiers

Each country has a different scheme for the identification of vehicles by license plates. In each case, the national Vehicle Licensing Authority is used as the authoritative source of such information.

In each type, there are often standard and non-standard formats, with the former following a prescribed system more tightly. In the identification of such plates, a likelihood metric is used to take into account such formats and give a confidence that an identifier is actually a vehicle license.

Medical

Documents that contain mention of medical procedures or conditions are identified with the Medical categories, available in each of the supported languages. The categories are generated from the Medical Subject Headings (MeSH) taxonomy published by the United States National Library of Medicine using the C hierarchy (diseases and conditions).

Profession

Documents that contain mention of an individual's occupation or profession are identified by the IDOL PHI Package in each supported language. The items are generated from an international database of over 60 million items.

Unique Device Identifiers

The IDOL PHI Package identifies Unique Device Identifiers (UDI) for medical devices. The formats match the standard formats issued by the three agencies accredited by the US Food and Drug Administration (FDA): GS1, HIBCC, and ICCBA.

Health Plan and Medical Record Numbers

The IDOL PHI Package identifies health plan numbers and Medical Record Numbers (MRN).

Health plan numbers have a standard format, which is readily available from governmental sources.

MRN formats differ for different healthcare provider. In this case, example formats are used to create as broad an identifier as possible, and landmark text is used to locate likely numbers. A grammar extension is also used to allow you to restrict the MRN detection to known formats, when you have specific formats you would like to detect. See [Medical Record Numbers, on page 30](#).

Birth Certificate Numbers

Birth certificate numbers have standard formats, which are readily available from governmental sources, such as the Social Security Administration (SSA).

Laboratory Numbers

The IDOL PHI Package identifies Clinical Laboratory Improvement Amendments (CLIA) laboratory numbers. The formats of these numbers is sourced from governmental sources, such as the Centres for Disease Control (CDC).

DEA

US Drug Enforcement Agency (DEA) numbers have a standard format. Official US Department of Justice sources, as well as secondary public sources, are used to define the format in the entities.

Accounts

The bank account and swift code formats have been compiled in existing Micro Focus Education grammars, and tested extensively against appropriate data.

New in this Release

This section describes the enhancements to the IDOL PHI Package in version 12.7.

- The address grammar now returns the PO_BOX component for all countries (previously, only UK and USA had this component). In addition, the following countries now detect the post office name, with the POST_OFFICE component: Canada, Italy, Lithuania, Norway, New Zealand, Portugal, Taiwan, and Japan.
- The address grammar has been improved to reduce false positive matches. In particular, in the recommended configuration, the grammar no longer matches an unknown street with an unknown city. Either the street name or the city must belong to one of the known lists.
- The name and address grammars now have additional landmark entities for the full name or address.
- Synonyms have been added to street components of the address entities. In these cases, when a synonym is detected, the main headword is returned in the normalized text. For example, Oxford St normalizes to Oxford Street in address entity matches.

Resolved Issues

This section lists the resolved issues in the IDOL PHI Package version 12.7.

- In the `phi_postprocessing.lua` scripts, `print()` was used instead of `error()` or `assert()` in some cases, which could result in errors not being correctly returned when using the Education SDK.

IDOL Education Grammars

The following section describes the Education grammars available in the IDOL PHI Package.

You can use these grammars with IDOL Education, by using Education Server, the `edktool` command-line utility, or the Education SDK. For more information, refer to the *IDOL Education User Guide* and the *Education SDK Programming Guide*.

IMPORTANT: To use the Education grammars in the IDOL PHI Package, you must have a license that enables them. To obtain a license, contact Micro Focus Support.

The IDOL PHI Package includes a default configuration file, which includes the basic required settings that you need to use the PHI grammars.

NOTE: If you create your own configuration file, you must include some of the settings in the default configuration file, such as post-processing and Education *components* (see [Configure Post Processing, below](#)).

Configure Post Processing

When you use the IDOL PHI Package Education grammars it is essential to configure a Lua post-processing task to run the script `phi_postprocessing.lua`. This script contains post-processing to improve results for various entities, such as stop list filtering, and checksum validation (see [Validated ID Numbers, on page 35](#)).

IMPORTANT: If you do not run this script, you might encounter unexpected behavior.

The default configuration file provided in the IDOL PHI Package includes a suitable post-processing task. If you use a different configuration, you must add the post-processing task to your Education configuration. For example:

```
[Education]
PostProcessingTask0=MyPostProcessingSection

[MyPostProcessingSection]
Type=Lua
Script=scripts/phi_postprocessing.lua
Entities=phi/*
```

IMPORTANT: The post-processing script requires Education components (see [Components, on page 23](#)). The default PHI configuration file enables components. If you use a custom configuration file you must set the `EnableComponents` parameter to `True` to return components.

For more information about configuring post-processing tasks, refer to the *Education User and Programming Guide*.

Configure Pre-Filtering

Pre-filtering allows the IDOL PHI Package to run a quick initial check to find potential matches in your input text. It then selects match windows around these potential matches, reducing the amount of text that it must match against your grammars. This process can improve the performance in certain cases.

Micro Focus recommends that you use the following pre-filtering configuration with the `address.ecr` grammar.

```
[Eduction]
PrefilterTask0=AddressPrefilter
```

```
[AddressPrefilter]
Regex=\d{1,7}
WindowCharsBeforeMatch=100
WindowCharsAfterMatch=100
```

NOTE: Pre-filter tasks run for all configured entities, so you must configure it only for the appropriate entities to ensure that it does not affect the results for other entities.

For more information about pre-filtering, refer to the *Eduction User and Programming Guide*.

Entity Context

Some of the entities are available in two versions, with and without context. The context-based entities match the entity when it occurs in an easily identifiable location in text. For example, it might match a telephone number that occurs next to the prefix **Phone**:

The entities that do not have context attempt to match the entity wherever it occurs. This version might over-match significantly (that is, it is likely to return values that are similar to the entity patterns, such a number that is not a telephone number). However, it also reduces the number of false negatives (that is, it misses fewer matches).

You can configure Eduction to use both versions of an entity; matches located with context are given a higher score in the results.

When you have data in tables, the context for an entity might not occur next to the entity value. For example, you might have a table with columns titled **name** and **date of birth**, but the values themselves do not occur next to these headers.

In this case, you can use Eduction table extraction to extract entities according to the landmarks detected in the table headers. For example, you can configure Eduction so that if it finds a table heading that matches the landmark **date of birth**, it extracts dates from that column.

For more information about how to configure table extraction, refer to the *Eduction User and Programming Guide*.

Balance Precision and Recall

In many cases, Education is able to locate entities that are ambiguous, such as a postal code which is simply a five-digit number. In some situations it is desirable to match as many entities as possible ("high recall") and in others only entities with a high likelihood of being a useful match ("high precision"). Each match is given a score value so that you can filter the results.

As described in [Entity Context, on the previous page](#), matches located by an entity that requires context are assigned higher scores than matches located by the corresponding entity without context. Most matches extracted without context have a score of 0.4. For example, a context-free date ("January 18, 1998") might be returned by a Date Of Birth entity with a score of 0.4. But with context to suggest that it is indeed a date of birth ("DOB: January 18, 1998"), the score should be above 0.5.

The PHI post-processing script (see [Configure Post Processing, on page 11](#)) includes a step to validate matches (for example, it can validate some ID numbers by calculating a checksum). The script increases the score of matches that have valid checksums, because this is an indication that the match is more likely to be genuine. Any match that has an invalid checksum is immediately discarded because it cannot be genuine.

When you configure Education, use the parameters `MinScore` and `PostProcessThreshold` to achieve the desired balance between precision and recall. Education discards any match with a score lower than `MinScore`. Matches with scores that meet or exceed `MinScore` are then processed by post-processing tasks. After post-processing has finished, Education discards any match with a score lower than `PostProcessThreshold`.

In the example configuration that is included with the IDOL PHI Package, `MinScore` is set to 0.4 and `PostProcessThreshold` is set to 0.5. These values have been chosen to return results only if they have a relatively high likelihood of being a useful match. Any match that is located without context can proceed to post-processing, but, unless its score is increased through successful validation, it is then discarded. If you prefer to maximize recall rather than precision, you can reduce or remove these thresholds.

For more information about Education configuration parameters, refer to the *Education User and Programming Guide*.

Configure Tangible Characters

`TangibleCharacters` is a configuration parameter that you can set when using the Education SDK, the Education Server, or the Education command-line utility (`edktool`). It specifies a list of characters to treat as part of a word, rather than as word boundaries.

Some of the entities in the IDOL PHI Package Education Grammars require tangible characters to be set in order to perform correctly (see the descriptions of the entities in [Education Grammar Reference, on the next page](#)).

When you use Education to search for matches, `TangibleCharacters` applies across all of your chosen entities. If you use multiple entities that have different recommended tangible character sets, you might need to take some extra steps. For example:

- If you are using the Eduction SDK, create a separate EDK engine for each distinct set of tangible characters, and configure the tangible characters for the engine using the appropriate API call:

C	<code>EdkSetTangibleCharacters</code>
Java	<code>EDKEngine.setTangibleCharacters</code>

After configuring an engine with the correct tangible characters, you can add the relevant entities. You will need to create a session from each engine to process your input text.

- If you are using an Eduction Server, send a separate action (`EduceFromText` or `EduceFromFile`) for each distinct set of tangible characters. In each action, set the `TangibleCharacters` and `Entities` action parameters to specify which set of tangible characters and which entities to use.
- If you are using the command line `edktool`, create a separate configuration file for each distinct set of tangible characters and associated entities, and process your input text once with each configuration file.

For more information about the `TangibleCharacters` configuration parameter, refer to the *Eduction User Guide*.

Customize Stop Lists

The IDOL PHI Package post-processing script (see [Configure Post Processing, on page 11](#)) uses stop lists to discard matches that are likely to be false positives. You can add entries to the stop lists, or remove entries, by modifying the following files.

- `scripts/address_stoplist.lua` contains a list of common words that are likely to indicate a false positive when returned as the `STREET` or `CITY` component of an address match.
- `scripts/names_stoplist.lua` contains two stop lists to discard names. In the first stop list, each component is plausible but the entire match is likely to be a false positive, for example "Christian Church" or "Norman Conquest". The second stop list contains common words that are likely to indicate a false positive when returned as either the `FORENAME` or `SURNAME` component of a name match. The stop lists in this file can be customized such that a name can be considered a false positive in one country but not another.

Eduction Grammar Reference

The following table describes the grammar files that are available in the IDOL PHI Package, and the entities that each provides.

account.ecr

Entity	Description
phi/account/bank/context/us	A US bank account number, with context.
phi/account/bank/nocontext/us	A US bank account number, without context.
phi/account/bank/landmark/us	A bank account landmark, such as "Bank Account Number".
phi/account/swiftcode/context/us	A SWIFT code, with context.
phi/account/swiftcode/nocontext/us	A SWIFT code, without context.
phi/account/swiftcode/landmark/us	A SWIFT code landmark, such as "SWIFT Code".

address.ecr

Entity	Description
phi/address/us	<p>A postal address.</p> <p>In general, a score of one is given to an address that includes a numbered, common format street address (for example "23 North Road"), a known city (for example "London"), and a postal code in a viable format for the country (for example "SW1A 2AA"). Deviations from this form lead to score penalties. The ordering of these elements varies by country.</p> <p>Micro Focus recommends that you use pre-filtering to improve the performance for this grammar. See Configure Pre-Filtering, on page 12.</p> <p>Example matches: "Schlosshoferstrasse 20, 1210 Vienna", "Avenida Juan Xxiii 20, 41006, Sevilla", "Abidei Hurriyet Cd Taner Palas Han 9 Kat:7 Dayre 9, 34437 Istanbul", "162-168 Regent Street, London, W1B 5TG".</p> <p>This entity returns the addresses in a normalized format by default. The normalized form standardizes apartment and house numbers, removes additional punctuation, and converts the text to uppercase. For example ABIDEI HURRIYET CD TANER PALAS APT 9, KAT:7, D:9, 34437 ISTANBUL. The exact order depends on the country.</p> <p>You can turn off normalization by setting <code>normalize_</code></p>

Entity	Description
	<p>addresses=false in the address_stoplist.lua script. This option can improve performance when you do not need normalization.</p> <p>This entity returns components. See Components, on page 23.</p>
phi/address/landmark/us	A postal address landmark. For example "Address".
phi/address/streetlocation/context/us	A street location (house number and street name), with context. For example "Address: 123, Mill Road".
phi/address/streetlocation/nocontext/us	A street location (house number and street name), without context. For example "123, Mill Road".
phi/address/streetlocation/landmark/us	A street location landmark. For example "Address"
phi/address/city/context/us	A city or town, with context. For example "City: London".
phi/address/city/nocontext/us	A city or town, without context. For example "London".
phi/address/city/landmark/us	A city or town landmark. For example "City".
phi/address/postcode/context/us	A postal code, with context. For example "Postcode: CB4 0WZ".
phi/address/postcode/nocontext/us	A postal code, without context. For example "CB4 0WZ".
phi/address/postcode/landmark/us	A postal code landmark. For example "Postcode".
phi/address/country/context/us	A country, with context. For example "Country: United Kingdom".
phi/address/country/nocontext/us	A country, without context. For example "United Kingdom".
phi/address/country/landmark/us	A country landmark. For example "Country".

age.ecr

Entity	Description
phi/age/over89/context/us	A US age statement with context. This entity finds ages over 89 years old. For example "Age: 99".
phi/age/over89/nocontext/us	A US age statement without context. This entity finds ages over 89 years old. For example "99 years old".
phi/age/landmark/us	A US age landmark. For example "Age".

certificate.ecr

Entity	Description
phi/certificate/birth/context/us	A US birth certificate number with context. For example "Birth Certificate: 160 99 123456".
phi/certificate/birth/nocontext/us	A US birth certificate number without context. For example "160 99 123456".
phi/certificate/birth/landmark/us	A US birth certificate landmark. For example "Birth Certificate".
phi/certificate/generic/context/us	A US generic certificate number with context. For example "Certificate number "MX-123-456/78".
phi/certificate/generic/nocontext/us	A US generic certificate number without context. This option is available only for the certificate user extension. See Generic Certificate Numbers, on page 31 .
phi/certificate/generic/landmark/us	A US generic certificate landmark. For example "Certificate number".

date.ecr

Entity	Description
phi/date/nocontext/eng	An English date, without context. For example, "01/13/1981".
phi/date/noyear/nocontext/eng	An English date without the year, without context. For example, "01/13"
phi/date/dob/context/eng	An English date of birth, with context. For example, "DOB: 1/13/1981".
phi/date/dob/noyear/context/eng	An English date of birth without the year, with context. For example, "DOB: 01/13".
phi/date/dob/landmark/eng	An English date of birth landmark. For example, "DOB".
phi/date/dod/context/eng	An English date of death, with context. For example, "Died on 01/13/1981".
phi/date/dod/noyear/context/eng	An English date of death without the year, with context. For example, "Died on 01/13".
phi/date/dod/landmark/eng	An English date of death landmark. For example, "Died on".

Entity	Description
phi/date/medical/context/eng	An English medical date with context. For example, "Admission date: 01/13/1981".
phi/date/medical/noyear/context/eng	An English medical date without the year, with context. For example, "Admission date: 01/13".
phi/date/medical/landmark/eng	An English medical date landmark. For example, "Admission date".

dea.ecr

Entity	Description
phi/dea/context/us	A US DEA (Drug Enforcement Agency) registration number with context. For example, "DEA Registration Number: BE1234563".
phi/dea/nocontext/us	A US DEA registration number without context. For example, "BE1234563".
phi/dea/landmark/us	A US DEA registration number landmark. For example, "DEA Registration Number".

device.ecr

Entity	Description
phi/device/udi/nocontext	<p>A Unique Device Identifier (UDI) required by the United States FDA, as issued by the three accredited agencies, GS1, HIBCC, and ICCBA. For example: "+X999123ABC0/\$\$31905151234AB/S5678EDFG/16D20151001J"</p> <p>NOTE: To ensure that this entity performs correctly, set your TangibleCharacters configuration to include the following characters: (+=). For more information, see Configure Tangible Characters, on page 13.</p>

healthplan.ecr

Entity	Description
phi/healthplan/context/us	A PHI health plan beneficiary number. These numbers consist of fourteen alphanumeric characters, preceded by a

Entity	Description
	suitable landmark. For example, "Plan ID 12345678LMNOP9".
phi/healthplan/nocontext/us	A PHI health plan beneficiary number without context. For example, "12345678LMNOP9".
phi/healthplan/landmark/us	A health plan number landmark. For example, "Health Plan Number".
phi/mrn/context/us	A Medical Record Number. See Medical Record Numbers, on page 30 .
phi/mrn/nocontext/us	A Medical Record Number without context. This option is available only for the healthplan user extension. See Medical Record Numbers, on page 30 .
phi/mrn/landmark/us	A Medical Record Number landmark, such as "MRN".

internet.ecr

Entity	Description
phi/inet/email/context	An email address with context. For example, "e-mail: jsmith@mailserver.com".
phi/inet/email/nocontext	An email address without context. For example, "jsmith@mailserver.com".
phi/inet/email/landmark	An email address landmark. For example, "e-mail".
phi/inet/ip/context	An IP address with context. For example, "ip address: 5.5.5.5".
phi/inet/ip/nocontext	An IP address without context. For example, "10.12.14.16".
phi/inet/ip/landmark	An IP address landmark. For example, "ip address".
phi/inet/url/context	A URL with context. For example, "uri: https://www.example.com".
phi/inet/url/nocontext	A URL without context. For example, "www.example.com".
phi/inet/url/landmark	A URL landmark. For example, "url".

laboratory.ecr

Entity	Description
phi/laboratory/context/us	A US laboratory number with context. For example "CLIA No: 01D1234567".
phi/laboratory/nocontext/us	A US laboratory number without context. For example "01D1234567".
phi/laboratory/landmark/us	A US laboratory number landmark. For example, "CLIA No".

license.ecr

Entity	Description
phi/license/driving/context/us	A US driving license number with context. For example, "Driving license: 012AB3456".
phi/license/driving/nocontext/us	A US driving license number without context. For example, "012AB3456".
phi/license/driving/landmark/us	A US driving license landmark. For example, "Driving license".
phi/license/generic/context/us	A US generic license number with context. For example, "License number: MX-123-456/78".
phi/license/generic/nocontext/us	A US generic license number without context. This option is available only for the license user extension. See Generic License Numbers, on page 31 .
phi/license/generic/landmark/us	A US generic license landmark. For example, "License number".

location.ecr

Entity	Description
phi/location/us	<p>A US subdivision smaller than a state, such as towns, cities, and counties. For example, "Houston".</p> <p>Scores are boosted by the presence of a state, a zipcode, or a nearby landmark value. For example, "Houston" scores</p>

Entity	Description
	0.4, while "Houston, Texas" scores 0.65 and "city of Houston, Texas" scores 1.

name.ecr

Entity	Description
phi/name/us	A full personal name, in title case or upper case. This entity returns the names in a normalized format, in the form <i>GIVEN NAME SURNAME</i> , for example JOHN SMITH. You can turn off normalization by setting <code>normalize_names=false</code> in the <code>name_stoplist.lua</code> script. You can also turn off score adjustment, by setting <code>rescore_names=false</code> in the <code>name_stoplist.lua</code> script. This option can improve performance when you do not need the normalization or score refinement.
phi/name/landmark/us	A full name landmark. For example "name".
phi/name/given_name/context/us	A given name, with context. For example "Forename: John".
phi/name/given_name/nocontext/us	A given name, without context. For example "John".
phi/name/given_name/landmark/us	A given name landmark. For example "Forename".
phi/name/surname/context/us	A surname with context. For example "Surname: Smith".
phi/name/surname/nocontext/us	A surname without context. For example "Smith".
phi/name/surname/landmark/us	A surname landmark. For example "Surname".
phi/name/pre_title/us	A title that precedes a name. For example "Ms".
phi/name/post_title/us	A title that follows a name. For example "Esq".

national_id.ecr

Entity	Description
phi/id/context/us	A national identity number (US Social Security Number) with context.
phi/id/nocontext/us	A national identity number (US Social Security Number) without context.
phi/id/landmark/us	A national identity number landmark, such as "Social security number".

medical_terms.ecr

Entity	Description
phi/medical_terms/eng	A medical condition or procedure. For example "abdominal hernia".

telephone.ecr

Entity	Description
phi/telephone/context/us	<p>A telephone number with context. For example "Telephone: (204)-243-9955".</p> <p>NOTE: To ensure that this entity performs correctly, set your <code>TangibleCharacters</code> configuration to include the following characters: ()+- . For more information, see Configure Tangible Characters, on page 13.</p> <p>This entity returns the telephone number in the normalized format +NNNNN, starting with the country code. For example +12042439955.</p>
phi/telephone/nocontext/us	<p>A telephone number without context. For example: "(204)-243-9955".</p> <p>NOTE: To ensure that this entity performs correctly, set your <code>TangibleCharacters</code> configuration to include the following characters: ()+- . For more information, see Configure Tangible Characters, on page 13.</p> <p>This entity returns the telephone number in the normalized format +NNNNN, starting with the country code. For example +12042439955.</p>
phi/telephone/landmark/us	A telephone number landmark, such as "Tel:".

vehicle.ecr

Entity	Description
phi/vehicle/licenseplate/context/us	A vehicle license place number with context. For example, "License Plate Number: ABC 123".

Entity	Description
phi/vehicle/licenseplate/nocontext/us	A vehicle license place number without context. For example, "ABC 123".
phi/vehicle/licenseplate/landmark/us	A vehicle license plate number landmark. For example, "License Plate Number".
phi/vehicle/vin/context/us	A vehicle identification number with context. For example, "VIN: LJPCBLCX11000237".
phi/vehicle/vin/nocontext/us	A vehicle identification number without context. For example, "LJPCBLCX11000237".
phi/vehicle/vin/landmark/us	A vehicle identification number landmark. For example, "VIN".

Components

Some of the PHI entities extract *components* as well as whole matches. Components are parts of a match that can provide useful information. For example, the entity for an address can return components for the street and postcode.

NOTE: The post-processing script requires components. The default PHI configuration file enables components. If you use a custom configuration file you must set the `EnableComponents` parameter to `True` to return components.

The following tables list the components available for particular entities.

address.ecr - address entity

Component Name	Notes
APARTMENT	
APARTMENT_BUILDING	tr and za
APARTMENT_PREFIX	ca only
BLOCK	br only
CITY	
COUNTRY	
COUNTY	gb only
FLOOR	ca, us, tr, and za
INTERSECTING_STREET	za only

address.ecr - address entity, continued

Component Name	Notes
MUNICIPALITY	za only
NUMBER	
PO_BOX	
POSTCODE	
POST_OFFICE	ca, it, lt, no, nz, pt, and za
PRIVATE_BAG_AGENCY	za only
PRIVATE_BAG_BOX_NUMBER	za only
PRIVATE_BAG_NUMBER	za only
PROVINCE	ca only
QUADRANT	br only
REGION	br and za
SECTOR	br only
SITE_ID	za only
SITE_TYPE	za only
STATE	us and au
STREET	
SUBURB	za only
TOWN	za only
TOWNSHIP	za only
VILLAGE_NAME	za only

The following examples demonstrate the use of these components.

- SHS Quadra 4 Bloco D, 70314-000, Brasília, Brazil

SECTOR: SHS
 QUADRANT: QUADRA 4
 BLOCK: BLOCO D
 CITY: BRASÍLIA
 POSTCODE: 70314000
 COUNTRY: BRAZIL

- Avenida João Jorge, 112, apto. 31, Campinas - SP, 13035-680, Brazil
NUMBER: 112
APARTMENT: APT 31
STREET: AVENIDA JOÃO JORGE
CITY: CAMPINAS
REGION: SP
POSTCODE: 13035680
COUNTRY: BRAZIL
- 2-400 Steeprock Dr Toronto M3J 2X1, Canada
APARTMENT_PREFIX: 2-
NUMBER: 400
STREET: STEEPROCK DR
CITY: TORONTO
POSTCODE: M3J2X1
COUNTRY: CANADA
- Çınar mahallesi. Basefendi sok. Küçükyalı Maltepe Asli apt. No:15 Kat 3, Maltepe, 34841 Istanbul, Turkey
APARTMENT_BUILDING: KÜÇÜKYALI MALTEPE ASLI APT
STREET: ÇINAR MAHALLESİ BASEFENDİ SOK
NUMBER: 15
FLOOR: KAT:3
POSTCODE: 34841
CITY: ISTANBUL
COUNTRY: TURKEY
- PO Box 34, DULUTH MN 55803-0034
PO_BOX: 34
CITY: DULUTH
STATE: MN
POSTCODE: 558030034
- PO BOX 90242, AUCKLAND MAIL CENTRE, AUCKLAND, 1142
PO_BOX: 90242
POST_OFFICE: AUCKLAND MAIL CENTRE
CITY: AUCKLAND
POSTCODE: 1142
- 70 Park Street West, Hatfield, City of Tshwane, Gauteng
NUMBER: 70
STREET: PARK STREET WEST
SUBURB: HATFIELD
MUNICIPALITY: CITY OF TSHWANE
REGION: GAUTENG
- Commission House, corner of Church Avenue and Hill Streets, Pretoria

APARTMENT_BUILDING: COMMISSION HOUSE
INTERSECTING_STREET: CHURCH AVENUE
STREET: HILL STREET
CITY: PRETORIA

- Farm 938 Rietfontein 341-JR

SITE_TYPE: FARM
NUMBER: 938
SUBURB: RIETFONTEIN
SITE_ID: 341-JR

- Postnet suite 8839, Private Bag X09, WELTEVREDENPARK, 1715

PRIVATE_BAG_AGENCY: POSTNET SUITE
PRIVATE_BAG_BOX_NUMBER: 8839
PRIVATE_BAG_NUMBER: X09
SUBURB: WELTEVREDENPARK
POSTCODE: 1715

- 110101 Corana, Umtata, 5100

NUMBER: 110101
VILLAGE_NAME: CORANA
TOWN: UMTATA
POSTCODE: 5100

- SA Post Office Ltd., PO Box 10 000, PROTEA PARK, 0305

POST_OFFICE: SA POST OFFICE LTD
PO_BOX: 10000
TOWNSHIP: PROTEA PARK
POSTCODE: 0305

- 123 London Street, Cambridge, Cambridgeshire, CB42AA

NUMBER: 123
STREET: LONDON STREET
CITY: CAMBRIDGE
COUNTY: CAMBRIDGESHIRE
POSTCODE: CB42AA

- 100 Boulevard Alexis-Nihon, Montreal (Quebec) H4M 2N7

NUMBER: 100
STREET: BOULEVARD ALEXIS NIHON
CITY: MONTREAL
PROVINCE: QUEBEC
POSTCODE: H4M 2N7

address_cjkvt.ecr - address entity

Component Name	Notes
ALLEY	tw only
APARTMENT	
BLOCK	jp only
BU_BLOCK	jp only
COUNTRY	
CITY	
CITY_BLOCK	jp only
CITY_BLOCK_DIRECTION	jp only
CITY_BLOCK_LANDMARK	jp only
CITY_BLOCK_NUMBER	jp only
DISTRICT	tw only
DISTRICT_DIRECTION	jp only
DISTRICT_NAME	jp only
DISTRICT_NUMBER	jp only
DISTRICT_NUMBER_JIKKAN	jp only
FLOOR	tw only
LANE	tw only
NEIGHBORHOOD	tw only
NUMBER	
PO_BOX	
POSTCODE	
POST_OFFICE	
REGION	
RURAL_AREA	jp only
RURAL_SUBUNIT	jp only
SECTION	tw only

address_cjkvt.ecr - address entity, continued

Component Name	Notes
STREET	tw only
TOWN	jp only
TOWNSHIP	tw only
VILLAGE	
WARD	jp only

The following examples demonstrate the use of these components:

- 日本、〒123-4567神奈川県津島市城南区月形町八重洲四丁目1番2-34号

COUNTRY: 日本
POSTCODE: 1234567
REGION: 神奈川県
CITY: 津島市
WARD: 城南区
TOWN: 月形町
DISTRICT_NAME: 八重洲
DISTRICT_NUMBER: 四
BLOCK: 1
NUMBER: 2
APARTMENT: 34

- 京都府東筑摩郡音威子府村大字滝沢字住吉6番地の6-22

REGION: 京都府
RURAL_AREA: 東筑摩郡
VILLAGE: 音威子府村
RURAL_SUBUNIT: 大字滝沢字住吉
BLOCK: 6
NUMBER: 6
APARTMENT: 22

- 札幌市中央区南二十二条西十六丁目2番地

CITY: 札幌市
WARD: 中央区
CITY_BLOCK_DIRECTION: 南
CITY_BLOCK_NUMBER: 二十二
DISTRICT_DIRECTION: 西
DISTRICT_NUMBER: 十六
BLOCK: 2

- 北海道札幌市厚別区厚別中央二条十丁目3番6号

REGION: 北海道
CITY: 札幌市
WARD: 厚別区
CITY_BLOCK_LANDMARK: 厚別中央
CITY_BLOCK_NUMBER: 二
DISTRICT_NUMBER: 十
BLOCK: 3
NUMBER: 6

- 札幌市中央区大通西二十丁目2番18号

CITY: 札幌市
WARD: 中央区
CITY_BLOCK: 大通
DISTRICT_DIRECTION: 西
DISTRICT_NUMBER: 二十
BLOCK: 2
NUMBER: 18

- 新潟県新潟市江南区楚川甲180-1

REGION: 新潟県
CITY: 新潟市
WARD: 江南区
DISTRICT_NAME: 楚川
DISTRICT_NUMBER_JIKKAN: 甲
BLOCK: 180
NUMBER: 1

- 鹿島郡中能登町良川卜部32番地の5

RURAL_AREA: 鹿島郡
TOWN: 中能登町
DISTRICT_NAME: 良川
BU_BLOCK: 卜
BLOCK: 32
NUMBER: 5

- 〒963-8642郡山郵便局私書箱12号

POSTCODE: 9638642
POST_OFFICE: 郡山郵便局
PO_BOX: 12

- 10603台北市大安區金山南路2段55號

POSTCODE: 10603
REGION: 台北市
DISTRICT: 大安區
STREET: 金山南路
SECTION: 2
NUMBER: 55

- 臺東縣關東河鄉都蘭村42鄰431之2號

REGION: 臺東縣
TOWNSHIP: 關東河鄉
VILLAGE: 都蘭村
NEIGHBORHOOD: 42
NUMBER: 431
APARTMENT: 2

- 臺北市大安區延吉街70巷5弄6號1樓

REGION: 臺北市
DISTRICT: 大安區
STREET: 延吉街
LANE: 70
ALLEY: 5
NUMBER: 6
FLOOR: 1

User Grammar Extensions

In some cases, the PHI grammar files provide entities for values that are very broad, to allow you to find values that do not have very well-defined formats. To reduce the number of false positives in these cases, only the context form of the entity is available by default (that is, with a suitable landmark).

You can use a user extension to expand the entity to include more specific formats, which enables the nocontext entity.

Medical Record Numbers

Medical Record Numbers (MRN) are provider-specific, and have a large number of possible formats. Therefore, the provided MRN entities in the `healthplan.ecr` grammar are generic, and matches a string of 7-15 alphanumeric characters, with optional - and / separators.

Because the MRN pattern is so broad, it can match a lot of values that are not actually MRNs. To reduce the number of false positives, only the context form of the entity is available by default (that is, with a suitable landmark).

If you have a specific set of MRN values with well-defined patterns that you want to match, you can use the `healthplan_user.xml` extension grammar. This XML file allows you to expand the MRN grammar with more specific patterns, and compile them into a grammar. In this case, the original context entity is available, with your additional entities, and it also enables the nocontext entity.

For details of how to expand and compile the user XML grammar, refer to the *Eduction User and Programming Guide*.

Generic License Numbers

The generic entities in the `license.ecr` grammar attempt to match a variety of different license numbers. The `phi/license/generic` entity matches a string of 5-20 alphanumeric characters (which must include at least one number), with optional - and / separators.

If you have a specific set of license values with well-defined patterns that you want to match, you can use the `license_user.xml` extension grammar. This XML file allows you to expand the grammar with more specific patterns, and compile them into a grammar. In this case, the original `context` entity is available, with your additional entities, and it also enables the `nocontext` entity.

For details of how to expand and compile the user XML grammar, refer to the *Education User and Programming Guide*.

Generic Certificate Numbers

The generic entities in the `certificate.ecr` grammar attempt to match a variety of different license numbers. The `phi/certificate/generic` entity matches a string of 5-20 alphanumeric characters (which must include at least one number), with optional - and / separators.

If you have a specific set of certificate values with well-defined patterns that you want to match, you can use the `certificate_user.xml` extension grammar. This XML file allows you to expand the grammar with more specific patterns, and compile them into a grammar. In this case, the original `context` entity is available, with your additional entities, and it also enables the `nocontext` entity.

For details of how to expand and compile the user XML grammar, refer to the *Education User and Programming Guide*.

PHI Grammar Customization

In cases where you find that the PHI grammars miss particular matches in your input, you can customize them. This section describes the possible customizations.

The following grammars support customization:

- `address.ecr`
- `name.ecr`

NOTE: It is technically possible to extend any public entity in a PHI grammar, but it can involve a lot of work. If you want to extend an entity that is not listed in the following list, see [Modify Other Grammars and Entities, on page 35](#).

For each grammar that supports customization, you can customize the following entities:

- address
 - phi/address/knowncity_headwords/us
 - phi/address/knownstreet/us
- name
 - phi/name/surname/nocontext/us
 - phi/name/given_name/nocontext/us

You can use customizations to add entries that the existing entities do not match (such as unusual names). You might also use it if your data uses unusual separators and punctuation. The following sections provide examples of these changes.

TIP: When you customize an entity, you can either replace or extend the definition. For PHI grammars, Micro Focus recommends that you only extend the entity definitions.

If you replace an entity, you are likely to miss matches or reduce performance. In addition, existing definitions cover many match cases that you might not consider, so there is a lot of value in using these definitions as a base.

Example 1: New Street Address

The following grammar definition below shows an example for extending address.ecr.

address_extended.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE grammars SYSTEM "../published/edk.dtd">
<grammars version="4.0">
  <include path="address.ecr"/>
  <grammar name="phi/address">

    <entity name="suffixes/us" type="private">
      <entry headword="Cury"/>
      <entry headword="CURY"/>
    </entity>

    <entity name="knownstreet/us" extend="append" type="private">
      <pattern>[A-Z][a-z]+ (?A:suffixes/us)</pattern>
    </entity>

    <entity name="streetlocation/nocontext/us" extend="append">
      <pattern score="0.75">(A=STREET:(?A:knownstreet/us))</pattern>
    </entity>

  </grammar>
</grammars>
```

This definition extends the `knownstreet/us` and `streetlocation/nocontext/us` entities in the PHI address grammar:

- It adds *Cury* as a known street suffix.
- It extends the `knownstreet` entity to accept any two word street name that ends with the new *Cury* suffix.
- It extends the `streetlocation/nocontext/us` entity to use the extended `knownstreet` entity, so that these changes take effect.

The result of these changes is that *Petty Cury* matches as a street location with a score of 0.75. Previously, it would not have matched at all.

TIP: You do not need to redeclare the full address entity to use the extended `knownstreet` entity.

For example, with these changes *123 Petty Cury, Cambridge MA 02140* now matches `pii/address/us` with a score of 1. Previously, this address would have matched, but with a lower score.

When you add known street names or patterns for your country of interest, it improves scores for matches that contain these customizations.

Example 2: New Known City

The following grammar definition adds more known cities to `address.ecr`.

`address_extended.xml`

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE grammars SYSTEM "../published/edk.dtd">
<grammars version="4.0">
  <include path="address.ecr"/>
  <grammar name="phi/address">
    <entity name="knowncity_headwords/us" extend="append" type="private">
      <entry headword="Chesterton"/>
    </entity>
    <entity name="city/nocontext/us" extend="append">
      <pattern>(?A=CITY:(?A^knowncity_headwords/us))</pattern>
    </entity>
  </grammar>
</grammars>
```

This example definition:

- adds *Chesterton* to the `knowncity_headwords/us` entity.
- extends the `city/nocontext/us` entity to use the extended `knowncity` entity, so that the change takes effect.

The result of these changes is that *Chesterton* matches as a city with a score of 1. Previously, it would have matched as a speculative city name, with a lower score.

Again, you do not need to change the full address entity to pick up this new declaration. For example, *123 Main Street, Chesterton MA 02140* now matches `pii/address/us` with a score of 1, which is an improved score. Previously, it would have matched with a lower score, because the city was a speculative match.

TIP: The definition for `city/nocontext/us` uses the dynamic reference syntax when using the `knowncity_headwords/us`; that is, `(?A^`. Micro Focus recommends this syntax for performance reasons when you refer to that entity, because the version of this entity for each country often contains several thousand entries.

To make both sets of changes for known streets and cities, merge the declarations in examples 1 and 2 into a single XML file.

Example 3: New Name and Custom Separator

Another way to use entity customizations is to declare patterns with custom separators. For example, if your input data contains unusual spacing or characters between entities, you can declare these in your entity extensions.

The following grammar definition extends `name.ecr`.

name_extended.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE grammars SYSTEM "../published/edk.dtd">
<grammars version="4.0">
  <include path="name.ecr"/>
  <grammar name="phi/name">

    <entity name="given_name/nocontext/us" extend="append" case="insensitive">
      <entry headword="Fobo" score="2"/>
    </entity>

    <entity name="surname/nocontext/us" extend="append" case="insensitive">
      <entry headword="Jobo" score="2"/>
    </entity>

    <entity name="gb" extend="append">
      <pattern>(A=SURNAME:(A:surname/nocontext/us))@@(A=FORENAME:(A:given_
name/nocontext/us))</pattern>
    </entity>

  </grammar>
</grammars>
```

This declaration makes two changes:

- It adds new entries for `given_name` and `surname`. This change allows *Fobo Jobo* to match as a name.
- It declares a new pattern for the `us` entity, to match a name in reverse order, with the elements separated by a custom separator (two `@` symbols). This change allows *Jobo@@Fobo* to match as a name.

TIP: The grammar already handles hyphenated known names. For example, after this definition change, `Eduction` matches *Fobo-Fobo Jobo* with a score of 1, with no further changes required. You do not need to add hyphenated entries to the `given_name/nocontext` or `surname/nocontext` entities.

Compile Custom Grammars

As with any `Eduction` grammar, Micro Focus recommends that you compile your grammar extensions before using them. You can use the `edktool` command-line tool to compile the XML file that contains your extension declarations into an ECR file.

For more information about compiling custom grammars, refer to the *Eduction User and Programming Guide*.

Modify Other Grammars and Entities

It is possible to extend any public entity in a PHI grammar. However, you cannot use the various private entities that the public ones use in their definitions.

For entities in the simpler grammars such as driving or national ID, this might be less of a problem, as long as you know the format for the data portion of this entity. For example, you might want to add new landmarks to these entities, for example.

However, be aware that existing definitions account for factors such as varying spaces, and additional words between the landmark and the data. In this case, you must emulate this behavior in your extensions, which might take a lot of work.

In practice, Micro Focus recommends that you make a support request to make these changes to the official PHI grammars, unless you need to add support in a very short time frame. The existing definitions provide a lot of value because they cover so many match cases, and you might miss these cases when you extend the public entities where these definitions are not available.

Validated ID Numbers

The script `phi_postprocessing.lua` (see [Configure Post Processing, on page 11](#)) includes steps to validate ID numbers that are found by `Eduction`. This improves accuracy by discarding results that match the pattern for a valid ID number, but cannot be genuine because they have an invalid checksum. The script increases the score for matches that have a valid checksum, because this is an indication that the match is more likely to be genuine.

The following tables list the entities that are validated.

Unique Device Identifiers (device.ecr)

phi/device/udi/nocontext

IDOL AgentBoolean IDX

IDOL AgentBoolean provides another way of finding pieces of information in text. In this case, you index the entities that you want to find into an IDOL Agentstore component.

The IDOL Agentstore component is a specially configured IDOL Content component. It uses IDOL AgentBoolean queries for entity matching.

When you use AgentBoolean for entity matching, each entity becomes a document in Agentstore. You then send a piece of text as a query to Agentstore, and it returns the entity documents that match the text.

The IDOL PHI Package contains IDX documents that describe entities for professions and occupations. You can use these IDX documents as another tool to find data that is protected by PHI regulations.

The package also contains an example Agentstore configuration file to allow you to set up your Agentstore component more easily.

NOTE: This example file does not contain all the required settings, such as license details, for your Agentstore. You might also want to add authorization roles and SSL settings for your system.

After you configure and set up your Agentstore, you can index the IDX documents and use Agentstore for entity matching. The following example index action indexes one of these IDX files:

```
http://localhost:5502/DREADD?C:\MicroFocus\PHIPackage\agentstore\idx\professions.idx
```

NOTE: The provided IDX files include a database field, so that Agentstore automatically indexes them into the correct database.

If you do not use the provided Agentstore configuration file, you must ensure that the professions database exists in your Agentstore. You must also configure the appropriate DatabaseType fields, or supply the DREDbName parameter in your DREADD index action.

For more information about how to set up and use IDOL querying, refer to the *IDOL Server Administration Guide* and the *IDOL Content Component Reference*.

Send documentation feedback

If you have comments about this document, you can [contact the documentation team](#) by email. If an email client is configured on this system, click the link above and an email window opens with the following information in the subject line:

Feedback on Technical Note (Micro Focus IDOL PHI Package 12.7)

Add your feedback to the email and click **Send**.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to swpdl.idoldocsfeedback@microfocus.com.

We appreciate your feedback!