IDOL PCI Package

Software Version 12.9

Technical Note

MICRO FOCUS

Document Release Date: June 2021
Software Release Date: June 2021

Legal notices

Copyright 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

The IDOL PCI Package contains tools that allow you to locate Payment Card Industry (PCI) in your data, to ensure compliance with financial regulations.

The IDOL PCI Package uses (.ecr files).

IDOL Eduction is a tool for finding entities (small pieces of information such as names and phone numbers) in text. Eduction 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 Eduction grammars included in the IDOL PCI Package describe different kinds of personally identifiable information, so that you can find these in your data.

Data Sources

The IDOL PCI Package contains a variety of different kinds of entities to describe payment card information that is protected by payment card industry regulations. The following sections provide some information about how this information is compiled.

For all of these types of information, as much test data is acquired as possible to test the recall metric of the algorithms. Many millions of examples are run through the grammars to ensure that all patterns in usage are covered.

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.

Other sources are also included for some countries, such as census data and lists of popular baby names. The list is also checked by performing Eduction over a large corpus of public data to find forenames and surnames that result in too many false positives, and add them to a name stop list.

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.

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 payment card date, rather than an unrelated date or other alphanumeric string.

PCI Numbers

The formats of the PCI numbers entities are sourced from the PCI Security Standards Council, and other public sources where appropriate.

New in this Release

This section describes the enhancements to the IDOL PCI Package in version 12.9.

- Packaged post-processing Lua scripts have been updated so that you are not required to modify the LUA_PATH environment variable to run post-processing.
- The name grammar has been updated for the USA using 2010 US Census data and Social Security Administration data. This change improves matching, particularly for prefixed names, mixed case names, and very short names.

Resolved Issues

There were no resolved issues in the IDOL PCI Package version 12.9.

Country and Language Support

The IDOL PCI Package contains grammars that apply to data from many countries and languages.

Country Codes

For data that corresponds to a particular country, the Eduction grammars identify each country by using the ISO 3166-1 alpha-2 country codes. The following countries are supported:

Country Code	Country
at	Austria
au	Australia
be	Belgium
bg	Bulgaria
br	Brazil
ca	Canada
ch	Switzerland
cn[1]	China
cy	Cyprus
cz	Czech Republic
de	Germany
dk	Denmark
ee	Estonia
es	Spain
fi	Finland
fr	France
gb	United Kingdom (England, Wales, Scotland, and Northern Ireland)
gr	Greece

[1]This country is available only in CJKVT grammars.

Country Code	Country
hr	Croatia
hu	Hungary
ie	Ireland
in[1]	India
is	Iceland
it	Italy
jp[2]	Japan
kr[3]	South Korea
li	Liechtenstein
lt	Lithuania
lu	Luxembourg
lv	Latvia
mt	Malta
nl	Netherlands
no	Norway
nz	New Zealand
pl	Poland
pt	Portugal
ro	Romania
se	Sweden
si	Slovenia
sk	Slovakia
tr	Turkey
tw[4]	Taiwan

[1]This country has entities only for names (in Latin script, and with English landmarks).
[2]This country is available only in CJKVT grammars.
[3]This country is available only in CJKVT grammars.
[4]This country is available only in CJKVT grammars.

Country Code	Country
us	United States of America
za	South Africa

Languages

For data that corresponds to a particular language, the Eduction grammars identify each language by using the ISO 639-2/B language codes. The following languages are supported:

Language Code	Language
eng	English

IDOL Eduction Grammars

The following section describes the Eduction grammars available in the IDOL PCI Package.

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

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

The IDOL PCI Package includes a default configuration file, which includes the basic required settings that you need to use the PCI 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 Eduction *components* (see Configure Post Processing, below).

Configure Post Processing

When you use the IDOL PCI Package Eduction grammars it is essential to configure a Lua post-processing task to run the script `pci_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 22).

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

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

```
[Eduction]
PostProcessingTask0=MyPostProcessingSection

[MyPostProcessingSection]
Type=Lua
Script=scripts/pci_postprocessing.lua
Entities=pci/*
```

IMPORTANT: The post-processing script requires Eduction components (see Components, on page 18). The default PCI 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 *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, Eduction 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, above, 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 PCI post-processing script (see Configure Post Processing, on the previous page) 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 Eduction, use the parameters `MinScore` and `PostProcessThreshold` to achieve the desired balance between precision and recall. Eduction 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, Eduction discards any match with a score lower than `PostProcessThreshold`.

In the example configuration that is included with the IDOL PCI 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 Eduction configuration parameters, refer to the *Eduction User and Programming Guide*.

Configure Tangible Characters

`TangibleCharacters` is a configuration parameter that you can set when using the Eduction SDK, the Eduction Server, or the Eduction 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 PCI Package Eduction Grammars require tangible characters to be set in order to perform correctly (see the descriptions of the entities in Eduction Grammar Reference, on the next page).

When you use Eduction 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	`EdkSetTangibleCharacters`
Java	`EDKEngine.setTangibleCharacters`

  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 PCI Package post-processing script (see Configure Post Processing, on page 10) 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/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 tables describe the grammar files that are available in the IDOL PCI Package, and the entities that each provides.

In the entity names, the abbreviation CC refers to a two-letter country code. For a list of available country codes, see Country Codes, on page 7.

TIP: You can use the Eduction parameter `EntityN` to specify which entities you want to extract. This parameter accepts wildcards, so you can extract entities of a specific type for all supported countries or languages. For example, to match names for all countries specify a value of `pci/name/??`.

NOTE: Many entities return components, in addition to the full match. For more information, and examples, see Components, on page 18.

date.ecr

Entity	Description
pci/date/nocontext/eng	A calendar date, written numerically or using words, without context. For example "01.03.1918", or "01/01/2020". This entity returns dates in the normalized ISO-8601 format YYYY-MM-DD. Partial dates without a day are formatted YYYY-MM. You can turn off normalization by setting `normalize_dates=false` in the `pci_postprocessing.lua` script. This option can improve performance when you do not need normalization.
pci/date/paymentcard/context/eng	A card date, with context. For example "Expires end: 01/20".
pci/date/paymentcard/nocontext/eng	A card date without context. For example "01/20".
pci/date/paymentcard/landmark/eng	A card date landmark. For example "Expires end".

name.ecr

Entity	Description
pci/name/CC	A full personal name, in title case or upper case.

Entity	Description
	This entity returns the names in a normalized format, in the form *GIVEN NAME SURNAME*, for example JOHN SMITH. You can turn off normalization by setting `normalize_names=false` in the `name_stoplist.lua` script. You can also turn off score adjustment, by setting `rescore_names=false` in the `name_stoplist.lua` script. This option can improve performance when you do not need the normalization or score refinement. This entity returns components. See Components, on page 18.
pci/name/landmark/CC	A full name landmark. For example "name".
pci/name/given_name/context/CC	A given name, with context. For example "Forename: John".
pci/name/given_name/nocontext/CC	A given name, without context. For example "John".
pci/name/given_name/landmark/CC	A given name landmark. For example "Forename".
pci/name/surname/context/CC	A surname with context. For example "Surname: Smith".
pci/name/surname/nocontext/CC	A surname without context. For example "Smith".
pci/name/surname/landmark/CC	A surname landmark. For example "Surname".
pci/name/pre_title/CC	A title that precedes a name. For example "Ms".
pci/name/post_title/CC	A title that follows a name. For example "Esq".
pci/name/title_surname/CC	A title and surname. For example "Mr. Smith".

name_cjkvt.ecr

Entity	Description
pci/name/CC	A full personal name, in romanized text or CJKVT native script. Romanized names can be in title case or upper case, and can be in the order *given name surname* or *surname given name*. CJKVT native script names must be *surname given name*. For Japanese, either form can include honorifics. This entity returns the names in a normalized format, in the form *GIVEN NAME SURNAME*, for example KEIKO NAKAMURA. You can turn off normalization by setting `normalize_names=false` in the `name_stoplist.lua` script. You can also turn off score adjustment, by setting `rescore_`

Entity	Description
	`names=false` in the `name_stoplist.lua` script. This option can improve performance when you do not need the normalization or score refinement. This entity returns components. See Components, on page 18.
pci/name/cjkvt/CC	A full personal name in CJKVT native script. For example "山田恵". This entity returns components. See Components, on page 18.
pci/name/latin/CC	A romanized full personal name. For example "Shinzo Abe". This entity returns components. See Components, on page 18.
pci/name/landmark/CC	A full name landmark. For example "名前".
pci/name/given_name/context/cjkvt/CC	A given name in CJKVT native script, with context. For example "名前: 直樹 ".
pci/name/given_name/nocontext/cjkvt/CC	A given name in CJKVT native script, without context. For example "直樹 ".
pci/name/given_name/context/latin/CC	A romanized given name, with context. For example "Given Name: Keiko".
pci/name/given_name/nocontext/latin/CC	A romanized given name, without context. For example "Keiko".
pci/name/given_name/context/CC	A given name in romanized text or CJKVT native script, with context. For example "名前: 直樹 ".
pci/name/given_name/nocontext/CC	A given name in romanized text or CJKVT native script, without context. For example "直樹 ".
pci/name/given_name/landmark/CC	A given name landmark in CJKVT native script. For example: "名前"
pci/name/surname/context/cjkvt/CC	A surname in CJKVT native script, with context. For example "名字: 山田".
pci/name/surname/nocontext/cjkvt/CC	A surname in CJKVT native script, without context. For example "山田".
pci/name/surname/context/latin/CC	A romanized surname, with context. For example "Surname: Nakamura".
pci/name/surname/nocontext/latin/CC	A romanized surname, without context. For example

Entity	Description
	"Nakamura".
pci/name/surname/context/CC	A surname in romanized text or CJKVT native script, with context. For example "名字: 山田".
pci/name/surname/nocontext/CC	A surname in romanized text or CJKVT native script, without context. For example "山田".
pci/name/surname/landmark/CC	A surname landmark in CJKVT native script. For example "名字".
pci/name/pre_title/nocontext/jp pci/name/pre_title/kr pci/name/pre_title/tw	A title that precedes a name in romanized text. For example "Ms".
pci/name/post_title/nocontext/latin/jp	A title that follows a name in romanized text. For example "Esq".
pci/name/post_title/nocontext/cjkvt/jp	A title that follows a name in Japanese script. For example "さん".
pci/name/post_title/nocontext/jp pci/name/post_title/kr pci/name/post_title/tw	A title that follows a name in romanized text, or Japanese script for Japan (jp). For example "Esq" or "さん".
pci/name/title_surname/latin/CC	A title and surname in romanized text. For example "Mr Chin-yi". This entity is not available for China (cn).
pci/name/title_surname/cjkvt/jp	A title and surname in Japanese script. For example "山田様".
pci/name/title_surname/CC	A title and surname in romanized text, or Japanese script for Japan (jp). For example "Mr Chin-yi" or "山田様". This entity is not available for China (cn).

pci_numbers.ecr

Entity	Description
pci/magstripe/context/magstripe	Magnetic stripe data with context. For example "Magstripe: %B5641821234567890122^SMITH/JOHN A. ^20111260000000000000000000?c". **NOTE:** To ensure that the entities in this grammar perform correctly, set your `TangibleCharacters` configuration to include the following characters: `%;`. For more information, see Configure Tangible Characters, on page 12.
pci/magstripe/nocontext/magstripe	Magnetic stripe data without context. For example

Entity	Description
	"%B5641821234567890122^SMITH/JOHN A. ^2011126000000000000000000000?c". **NOTE:** To ensure that the entities in this grammar perform correctly, set your `TangibleCharacters` configuration to include the following characters: `%;`. For more information, see Configure Tangible Characters, on page 12.
pci/magstripe/landmark/magstripe	A magnetic stripe landmark. For example "Magstripe".
pci/pan/context/pan	A Primary Account Number with context. For example "PAN: 4485221211756505".
pci/pan/nocontext/pan	A Primary Account Number without context. For example "4485 2212 1175 6505".
pci/pan/landmark/pan	A Primary Account Number landmark. For example "PAN".
pci/pin/context	A card Personal Identification Number with context. For example "PIN: 1234".
pci/pin/nocontext	A card Personal Identification Number without context. For example "1234".
pci/pin/landmark	A card Personal Identification Number landmark. For example "PIN".
pci/pin_block/context	An encrypted or unencrypted PIN block with context (either base-64, base-16 or base-2). For example "PIN block: BABCDEFGHIJ=".
pci/pin_block/nocontext	An encrypted or unencrypted PIN block without context (either base-64, base-16 or base-2). For example "BABCDEFGHIJ=".
pci/pin_block/landmark	A PIN block landmark. For example "PIN block".
pci/printed_security_code/context/cav2	A CAV2 security code with context. For example "CAV2: 123".
pci/printed_security_code/landmark/cav2	A CAV2 security code landmark. For example "CAV2".
pci/printed_security_code/context/cid	A CID security code with context. For example "CID: 1234".
pci/printed_security_code/landmark/cid	A CID security code landmark. For example "CID".
pci/printed_security_code/context/cvc2	A CVC2 security code with context. For example "CVC2: 123".

Entity	Description
pci/printed_security_code/landmark/cvc2	A CVC2 security code landmark. For example "CVC2".
pci/printed_security_code/context/cvv2	A CVV2 security code with context. For example "CVV2: 123".
pci/printed_security_code/landmark/cvv2	A CVV2 security code landmark. For example "CVV".
pci/printed_security_code/nocontext	Any of CAV2, CID, CVC2 or CVV2 security code without landmark. For example "123".
pci/service_code/context	A service code with context. For example "Service code: 123".
pci/service_code/nocontext	A service code without context. For example "123".
pci/service_code/landmark	A Service code landmark. For example "Service code".

Components

Some of the PCI entities extract *components* as well as whole matches. Components are parts of a match that can provide useful information.

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

The following sections list the components available for particular entities.

NOTE: The PCI grammars sometimes generate additional components, which it uses to eliminate false positives during post-processing. If you disable post-processing, you might see these additional components. However, Micro Focus recommends that you always enable post-processing for these grammars.

Name Components

name and name_cjkvt - name, name/latin, and name/cjkvt entities

Component Name	Notes
PRE_TITLE	

name and name_cjkvt - name, name/latin, and name/cjkvt entities, continued

Component Name	Notes
FORENAME	
INITIAL	
INITIAL_NODOT	This component is for internal use during post-processing.
SURNAMEPREFIX	
SURNAME	
POST_TITLE	
SURNAME_POSSESSIVE	jp only

The following examples demonstrate the use of these components.

- Dr Jane D O'Reilly Jr

  ```
  PRE_TITLE: DR
  FORENAME: JANE
  INITIAL: D
  INITIAL_NODOT
  SURNAMEPREFIX: O
  SURNAME: REILLY
  POST_TITLE: JR
  ```

- Watanabe no Tsuna

  ```
  FORENAME: TSUNA
  SURNAME_POSSESSIVE: NO
  SURNAME: WATANABE
  ```

- 山田 太郎

  ```
  FORENAME: 太郎
  SURNAME: 山田
  ```

PCI Numbers Components

pci_numbers- pci/magstripe context and nocontext entities

Component Name	Notes
EXPIRATION_DATE	
HOLDER_NAME	
PAN	

The following examples demonstrate the use of these components.

- %B4012123456789012^SMITH/ADAM ^2012126000000000000000000000?c

  EXPIRATION_DATE: 2012
  HOLDER_NAME: ADAM SMITH
  PAN: 4012123456789012

PCI Grammar Customization

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

The following grammars support customization:

- `name.ecr`
- `name_cjkvt.ecr`

> **NOTE:** It is technically possible to extend any public entity in a PCI 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 the next page.

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

- name
  - `pci/name/surname/nocontext/CC`
  - `pci/name/given_name/nocontext/CC`

In this list, CC means country code (for example: gb, us, nz). See Country and Language Support, on page 7.

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 section provides an example of these changes.

> **TIP:** When you customize an entity, you can either replace or extend the definition. For PCI 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: New Name and Custom Separator

The following example shows how to add a new given name and surname to an entity. It also shows how 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="pci/name">

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

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

      <entity name="gb" extend="append">
         <pattern>(?A=SURNAME:(?A:surname/nocontext/gb))@@(?A=FORENAME:(?A:given_
name/nocontext/gb))</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 for the `gb` entity.
- It declares a new pattern for the gb 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 PCI 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 PCI 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 `pci_postprocessing.lua` (see Configure Post Processing, on page 10) 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.

Magnetic Stripe Data (`magstripe.ecr`)		
pci/magstripe/context/magstripe	pci/magstripe/nocontext/magstripe	Validation implicitly validates the Primary Account Number (PAN) that is included in the magstripe data.

Primary Account Numbers (`pan.ecr`)	
pci/pan/context/pan	pci/pan/nocontext/pan

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 Micro Focus IDOL PCI Package 12.9 Technical Note

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!