Informatica

Informatica® Address Verification (On-Premises)

6.5.0

On-Premises Developer Guide

Informatica Address Verification (On-Premises) On-Premises Developer Guide
6.5.0
June 2024

Publication Date: 2024-06-03

Table of Contents

Preface

Read the Informatica Address Verification Developer (On-Premises) Guide to learn how you can use Informatica Address Verification (On-Premises) to analyze, parse, validate, and enrich postal address data. This guide also describes the result output fields and process status codes that Address Verification (On-Premises) returns.

Informatica Resources

Informatica provides you with a range of product resources through the Informatica Network and other online portals. Use the resources to get the most from your Informatica products and solutions and to learn from other Informatica users and subject matter experts.

Informatica Network

The Informatica Network is the gateway to many resources, including the Informatica Knowledge Base and Informatica Global Customer Support. To enter the Informatica Network, visit https://network.informatica.com.

As an Informatica Network member, you have the following options:

- Search the Knowledge Base for product resources.
- View product availability information.
- Create and review your support cases.
- Find your local Informatica User Group Network and collaborate with your peers.

Informatica Knowledge Base

Use the Informatica Knowledge Base to find product resources such as how-to articles, best practices, video tutorials, and answers to frequently asked questions.

To search the Knowledge Base, visit https://search.informatica.com. If you have questions, comments, or ideas about the Knowledge Base, contact the Informatica Knowledge Base team at KB_Feedback@informatica.com.

Informatica Documentation

Use the Informatica Documentation Portal to explore an extensive library of documentation for current and recent product releases. To explore the Documentation Portal, visit https://docs.informatica.com.

If you have questions, comments, or ideas about the product documentation, contact the Informatica Documentation team at infa_documentation@informatica.com.

Informatica Product Availability Matrices

Product Availability Matrices (PAMs) indicate the versions of the operating systems, databases, and types of data sources and targets that a product release supports. You can browse the Informatica PAMs at https://network.informatica.com/community/informatica-network/product-availability-matrices.

Informatica Velocity

Informatica Velocity is a collection of tips and best practices developed by Informatica Professional Services and based on real-world experiences from hundreds of data management projects. Informatica Velocity represents the collective knowledge of Informatica consultants who work with organizations around the world to plan, develop, deploy, and maintain successful data management solutions.

You can find Informatica Velocity resources at http://velocity.informatica.com. If you have questions, comments, or ideas about Informatica Velocity, contact Informatica Professional Services at ips@informatica.com.

Informatica Marketplace

The Informatica Marketplace is a forum where you can find solutions that extend and enhance your Informatica implementations. Leverage any of the hundreds of solutions from Informatica developers and partners on the Marketplace to improve your productivity and speed up time to implementation on your projects. You can find the Informatica Marketplace at https://marketplace.informatica.com.

Informatica Global Customer Support

You can contact a Global Support Center by telephone or through the Informatica Network.

To find your local Informatica Global Customer Support telephone number, visit the Informatica website at the following link:
https://www.informatica.com/services-and-training/customer-success-services/contact-us.html.

To find online support resources on the Informatica Network, visit https://network.informatica.com and select the eSupport option.

Part I: On-Prem Operations

This part contains the following chapters:

CHAPTER 1

Introduction

This chapter includes the following topics:

Informatica Address Verification Overview

Informatica Address Verification (On-Premises) verifies addresses from more than 240 countries and territories quickly, efficiently, and effectively.

The address verification workflow includes steps to validate, format, enrich, and certify addresses. Address Verification standardizes and formats the address data to meet the requirements that the local postal authorities specify. It uses integrated transliteration capabilities to validate addresses that you capture in different writing systems and languages. Postal certification improves the quality of addresses and ensures that address verification services meet postal authority requirements.

Address Verification complies with the following postal certifications:

- Address Matching Approval System (AMAS) certification for Australia Post
- Coding Accuracy Support System (CASS) certification for the United States Postal Services
- SendRight certification for New Zealand Post
- Service National de L'Adresse (SNA) certification for La Poste of France
- Software Evaluation and Recognition Program (SERP) certification for Canada Post

Additionally, Address Verification is an accredited Eircode encoder in Ireland.

Address Verification Modes

Informatica Address Verification validates your address records against one or more reference databases and returns results that can include corrected, completed, and enhanced versions of each address.

Address Verification returns the best available matches for an input address. Address Verification also returns detailed status data that describes the type, accuracy, and deliverability of each address.

Choose the address validation mode that suits your requirements:

- To submit one or more addresses for validation and receive a single verified result for each address, choose Batch mode.
- To validate addresses to the certification standards in Australia, Canada, France, New Zealand, or the United States, choose Certified mode. Certified mode requires additional certified reference address databases.
- To receive a list of address suggestions from which you can choose the most appropriate record, choose Interactive mode.
- To receive address suggestions as you enter an address, choose QuickCapture mode. In QuickCapture, Address Verification returns suggestions even for incomplete or truncated addresses.
- To receive the address or addresses that match a pair of latitude and longitude coordinates that you enter, choose GeocodeToAddress mode.

For more information about address validation modes, see "Process Modes" on page 32.

Verification Modes and Input Address Formats

The verification mode that you select for your input address data can depend on the structure of your data set. You can select batch, interactive, and certified modes for all address formats. If you select QuickCapture mode, your complete address must occupy a single field.

Your organization might store data in one of several formats. For example, your data set might use a discrete field for each address element, or it might store multiple address elements in each field.

Example: United States Address

Your data set might contain the following United States address:

123 MAIN ST UNIT 4567, KRYTON TN 38101

You might store the address in a database table in one of the following formats:

Partially-Fielded Address Format

Your data set might store multiple address elements in each field.

The following table shows how you can map the fields in your data set to fields in Address Verification:

PostalFormattedAddressLine	PostalFormattedAddressLine
123 MAIN ST UNIT 4567 KRYTON TN 38101	KRYTON TN 38101

Hybrid Address Format

Your data set might store multiple address elements in some fields and unique elements in others.

The following table shows how you can map the fields in your data set to fields in Address Verification:

PostalDeliveryAddressLine	City	State	ZIP
123 MAIN ST UNIT 4567	KRYTON	TN	38101

Discrete Address Format

Your data set might store address elements in discrete fields.

The following table shows how you can map the fields in your data set to fields in Address Verification:

HouseNumber	Street	Sub-Building	City	State	ZIP
123	MAIN ST	UNIT 4567	KRYTON	TN	38101

Single Address Format

Your data set might store the address in a single field.

The following table shows how you can map the field in your data set to an Address Verification field:

SingleAddressLine
123 MAIN ST UNIT 4567 KRYTON TN 38101

Note: Choose QuickCapture mode for the SingleAddressLine entry of your complete address. Do not include country information in the SingleAddressLine field. Add a Country field for country information.

The following table lists the address fields that you can select for each format and the verification modes that support the formats:

Input Address Format	Field Examples	Supported Verification Modes
Partially fielded	PostalFormattedAddressLines, PostalDeliveryAddressLines	Batch, Interactive, Certified
Hybrid (discrete and multi-element)	AddressElements, PostalFormattedAddressLines, PostalDeliveryAddressLines	Batch, Interactive, Certified
Discrete fields for address elements	AddressElements	Batch, Interactive, Certified
Unfielded/single-line input	SingleAddressLine	QuickCapture, Batch, Interactive

Parallel Processing Options for Efficient Multiple Address Execution

To maximize the performance of multiple address outputs with bulk addresses, use the parallel processing option. You can either use individual threads for sequential address processing, or distribute addresses across multiple threads at the same time.

There are two ways in which Address Verification can parallelly process multiple addresses:

- You can set up multiple threads and enter one address at a time through each thread. This option utilizes all available function servers and improves the average processing speed. The number of available function servers is determined by the NumFunctionServer parameter value.
- You can enter bulk addresses in Batch or Certified mode, and simultaneously process multiple addresses (up to 1000 addresses) through one or more threads. Address Verification distributes multiple addresses across all available function servers. If all the available function servers are not utilized at the same time, then some function servers may remain idle while others run the processing job. As a result, the average processing speed reduces.

Address Verification Process

The address verification process includes multiple steps for validation, formatting, enrichment, character set mapping, and transliteration. Based on the parameters you set, Informatica Address Verification completes the verification process and returns the result.

Address Formatting and Standardization

After validating an address, Informatica Address Verification formats the address according to the standardization parameters that you configure.

The standardization properties that you can configure include the following:

- Whether to include the country information in the output.
- The language in which to return the address.
- The script in which to return the address.
- Whether to return alias versions of an element.
- The case of the output.
- Whether to use element abbreviations.

You can configure standardization properties at the global level and at the country-specific level.

You can find the standardization properties in the `AVJob.schema.json` file under **CountrySets**.

Address Enrichments

You can retrieve address enrichments with the validated result output. Address enrichments provide additional information that helps you better understand and use the address data.

You can retrieve enrichments for validated addresses in batch and interactive modes. Address Verification can also return enrichments for addresses that fail validation in some cases.

You can retrieve the following address enrichments:

General Enrichments

You can specify enrichments at a global level for all countries that you verify, or you can specify enrichments at a country-by-country level.

Address Verification can provide information such as geographic and administrative codes and identifiers that are specific to countries or territories. Examples for country-specific enrichments include postal address codes for Austria addresses, INSEE codes for France addresses, and time zone codes for United States addresses.

Geocoding

Geocoding finds the physical location of an address. Geocodes are latitude and longitude coordinates.

Character Set Mapping

Character set mapping provides a mapping between source and destination character sets and thus enables conversion between character sets. Informatica Address Verification internally uses Unicode and externally supports multiple character sets, including UTF-8, ISO 8859-1, GBK, BIG5, JIS, and EBCDIC.

Character sets use a numeric representation for each of the supported alphabets or characters. Typically, character sets use the same numeric representation for common alphabets or characters. However, some of the language-specific characters have different numeric representations across character sets.

For example, the letter A has the same numeric representation, 65, in both Unicode and Latin character sets. However, the letter Å has different representations in Unicode and Latin character sets. Å is represented by 143 in Unicode and 197 in Latin character sets.

To render character sets, Address Verification first converts the input character strings to Unicode. Then it uses the corresponding mapping of the destination character set to render the data with near perfection. If no representation is available for a character in the destination character set, Address Verification maps that character to an underscore character.

Note: Submit addresses to Address Verification in a single character set.

Transliteration

Transliteration converts data between non-Roman characters and Latin characters. Transliteration can also replace diacritical and extended characters with plain text equivalents.

Transliteration helps nonnative speakers find an approximate pronunciation of a word based on the pronunciation rules of their own language. Transliterations of ISO character sets use invertible mapping so that the transliteration can be reversed without any information loss. However, for other character sets, such as BGN, transliteration is not reversible.

Informatica Address Verification can transliterate to and from the following writing systems:

- Chinese Pinyin (Mandarin, Cantonese)
- Cyrillic (BGN/PCGN 1947, ISO 9 – 1995)

  You can perform Cyrillic transliteration for Belarus, Bulgaria, Kazakhstan, Macedonia, Russia, and Ukraine.
- Greek (BGN/PCGN 1962, ISO 843 – 1997)
- Hebrew
- Japanese Katakana, Hiragana, and Kanji

Transliteration goes beyond character set mapping, which is a mapping between different numeric representations of a character. A language such as Japanese, with Katakana, Hiragana, and Kanji characters, has sounds with no direct representation in the English language. However, each Japanese character has an associated sound that can be approximated phonetically in Latin characters.

The following table shows transliteration of sample characters from different character sets:

Source Character Set	Input	Destination Character Set	Output
Latin	Ä	ASCII	AE
Latin	ĝ	ASCII	g

Source Character Set	Input	Destination Character Set	Output
Kanji (Japanese)	市	Latin	shi
Cyrillic	Ж	Latin	ZH

Transliteration Limitations

If the source language has fewer syllables and does not have exact matches in other languages, transliteration becomes difficult. Japanese is an example for languages that are difficult to transliterate.

Most languages use a subset of the sounds a person could produce. Different languages use different subsets. If a sound used by one language cannot be represented correctly in a non-native script, then it needs to be approximated. This approximation might be inaccurate if the sounds in the source and destination languages are significantly different.

The following table shows how transliteration can be inaccurate when the words are transliterated from English (Latin) to Japanese (Katakana) and back to English (Latin):

Original Latin (English)	Katakana (Japanese)	Transliterated Latin
Philippines	フィリピン	Firipin
Düsseldorf	デュッセルドルフ	Dyusserudorufu
Beethoven	ベートーベン	Betoben

Some Kanji characters in the first name of the contact in Japan addresses are incorrectly transliterated into Arabic numerals instead of Latin alphabets.

The following table shows the Kanji numerals that Informatica Address Verification might incorrectly translate into Arabic numerals instead of Latin alphabets:

Kanji Numeral	Arabic Equivalent	Latin Transliteration
一	1	ichi
二	2	ni
三	3	san
四	4	yon
五	5	go
六	6	roku
七	7	nana
八	8	hachi

Kanji Numeral	Arabic Equivalent	Latin Transliteration
九	9	kyū
十	10	jū

CHAPTER 2

Verifying Addresses

This chapter includes the following topics:

Verification Operations Overview

Informatica Address Verification runs address jobs in the Intelligent Data Verification engine (IDVE).

The main address verification functionality is encapsulated in one or more *function servers*. A function server is a running instance of the Informatica data verification engine.

Each function server runs as a separate process, isolating the main process from any issues related to address processing. If a function server fails, the event is reported back to the caller as an error, the function server process is terminated, and a new function server is started.

The `IDVE.h` header file contains the functions that you use to define and run the address jobs.

Verification Tasks

To verify an address, you perform the following tasks:

1. Create the job objects.
2. Set the parameters.
3. Submit the address set for processing. The address set can be one or more records.
4. Retrieve the results.

5. Clear the input and output data.
6. Delete the job objects.

Note: If you process more than one address set with the same parameters, you can perform steps 3 through 5 in a loop. You can reuse the job object that you create for multiple address calls, so long as the job object is appropriate to the verification process that you want to perform.

Creating a Job

A job object is a passive data structure that contains parameters, input data, and result data based on the `AVJob.schema.json` specification. Each job has its own Uniform Resource Identifier (URI), which the function returns when creating the job.

Before you run a job request, use the IDVE_Post() or IDVE_PostW() function to create a job object. Use the IDVE_Post() function to retrieve the job URI in the UTF-8 encoding. Use the IDVE_Post()W function to retrieve the job URI in the UTF-16 encoding.

Each function allows you to append one or more commands to the URI to specify the type of action that the function must perform. To create a job, append `create` to the URI string. The `create` command is not a part of the schema.

The functions also provide arguments to specify input and output data. The input and output content depends on the commands that you use.

Note: All parameters, elements, and values in Address Verification JSON schemas are case-sensitive. Take care to follow the character case that the AVJob.schema.json file specifies.

Creating a Job: UTF-8 Variant

The following sample code shows the structure of the IDVE_Post() function:

```
IDVE_EXPORTCALL1 IDVE_StatusCode IDVE_EXPORTCALL2 IDVE_Post(
   const char* const kpksCustomerID,
   const char* const kpksURI,
   const char* const kpksInput,
   const IDVE_U64 ku64InputLength,
   char* const kpsOutput,
   const IDVE_U64 ku64OutputBufferSize,
   IDVE_U64* const kpu64SizeWritten,
   char* const kpsExtStatusMsg
);
```

The following table shows the parameter definitions for the IDVE_Post() function:

Parameter	Operation	Comment
const char* const kpksCustomerID	[in]	Pointer to the zero-terminated 7-bit ASCII-encoded customer ID. The value might not be NULL.
const char* const kpksURI	[in]	Pointer to the zero-terminated 7-bit ASCII URI. The value might not be NULL.
const char* const kpksInput	[in]	Pointer to the UTF-8-encoded input string. The value can be NULL if there is no input string. The content of the input string depends on the command. For the create command, the string content must be NULL.

Parameter	Operation	Comment
const IDVE_U64 ku64InputLength	[in]	Length of the input string in number of code units excluding a possibly terminating zero. The value can be string length or IDVE_AUTOLEN.
char* const kpsOutput	[out]	Pointer to an UTF-8-encoded output string buffer for the output. The value can be NULL. For the create command, the output string contains the job ID and the full URI to the root of the job.
const IDVE_U64 ku64OutputBufferSize	[in]	Size of the output string buffer in code units, including the terminating zero.
IDVE_U64* const kpu64SizeWritten	[out]	Pointer to an unsigned 64-bit integer value that receives the size of the output written in code units, excluding the terminating zero. The value can be NULL.
char* const kpsExtStatusMsg	[out]	Pointer to a buffer of size IDVE_EXT_STATUS_MSG_BUFFER_SIZE for an optional extended status message. The value can be NULL.

Creating a Job: UTF-16 Variant

The following sample code shows the structure of the IDVE_PostW() function:

```
IDVE_EXPORTCALL1 IDVE_StatusCode IDVE_EXPORTCALL2 IDVE_PostW(
const char* const kpksCustomerID,
const char* const kpksURI,
const IDVE_WChar* const kpksInput,
const IDVE_U64 ku64InputLength,
IDVE_WChar* const kpsOutput,
const IDVE_U64 ku64OutputBufferSize,
IDVE_U64* const kpu64SizeWritten,
char* const kpsExtStatusMsg
);
```

The following table shows the parameter definitions for the IDVE_PostW() function:

Parameter	Operation	Comment
const char* const kpksCustomerID	[in]	Pointer to the zero-terminated 7-bit ASCII-encoded customer ID. The value might not be NULL.
const char* const kpksURI	[in]	Pointer to the zero-terminated 7-bit ASCII URI. The value might not be NULL.
const IDVE_WChar* const kpksInput	[in]	Pointer to the UTF-16-encoded input string. The value can be NULL if there is no input string. The content of the input string depends on the command. For the create command, the string content must be NULL.
const IDVE_U64 ku64InputLength	[in]	Length of the input string in number of code units excluding a possibly terminating zero. The value can be string length or IDVE_AUTOLEN.
IDVE_WChar* const kpsOutput	[out]	Pointer to an UTF-16-encoded output string buffer for the output. The value can be NULL. For the create command, the output string contains the job ID and the full URI to the root of the job.

Parameter	Operation	Comment
const IDVE_U64 ku64OutputBufferSize	[in]	Size of the output string buffer in code units, including the terminating zero.
IDVE_U64* const kpu64SizeWritten	[out]	Pointer to an unsigned 64-bit integer value that receives the size of the output written in code units, excluding the terminating zero. The value can be NULL.
char* const kpsExtStatusMsg	[out]	Pointer to a buffer of size IDVE_EXT_STATUS_MSG_BUFFER_SIZE for an optional extended status message. The value can be NULL.

Example: Creating a Job

The following example shows an IDVE_Post() function call to create a job:

```
char sJobURI[ IDVE_POST_OUTPUT_BUFFER_SIZE ];
char sExtStatusMsg[ IDVE_EXT_STATUS_MSG_BUFFER_SIZE ];
IDVE_StatusCode i32StatusCode = IDVE_Post( "", "AV/v1/Jobs/create", NULL, 0, sJobURI,
sizeof( sJobURI ), NULL, sExtStatusMsg );
```

The parameters in the IDVE_Post call have the following meanings:

- The first parameter is an empty string. You can specify a customer ID.
- The second parameter is the URI.
- The third parameter is an unused input string.
- The fourth parameter is the length of the input string.
- The fifth parameter is a pointer to an output string buffer.
- The sixth parameter is the length of the output string buffer.
- The seventh parameter is NULL, because you do not want to receive the number of chars of the output string.
- The final parameter is a pointer to an output buffer for an extended error message.

The call creates a new job object and, in case of success, returns its URI in the string buffer sJobURI. In the URI string "`AV/v1/Jobs/0`", the final element "0" might be arbitrary, for example `E9FC48AF9`. The value is a handle to a unique job.

After you create a job object, set or verify the parameters.

Setting the Parameters

After you create a job, you set the parameters.

The `AVJob.schema.json` file defines the parameters and specifies default values for many of the parameters. To set the parameters, use the IDVE_PutJSON() or IDVE_PutJSONW() function.

Note: All parameters, address elements, and values that you use in the functions are case-sensitive.

You set or verify the parameters once for each bulk processing job.

Note on Get and Put Function calls

The steps in this chapter docs focus on the use of IDVE_PutJSON to set the address data and IDVE_GetJSON to get the result. However, in principle you can use any of the following functions to set the address data: PutJSON, PutString, PutInt32, PutInt64, and PutBool.

Likewise, you can use any of the following functions to get the address result: GetJSON, GetString, GetInt32, GetInt64, and GetBool.

You can decide how to set or get the data, either in a single activity (with a JSON function), or based on individual values (with string, int, or bool functions), or with any combination of the functions. You can also use the functions to set or get the parameters of an object.

Submitting a Job Request

After you create a job, submit one or more addresses for verification.

The process to submit an address has the following steps:

- Enter an input address.
- Request processing for the address.

Entering the Input Address Data

To enter address data for verification, use the IDVE_PutJSON() or IDVE_PutJSONW() function. The IDVE_PutJSON() function passes the data in a UTF-8-encoded string. The IDVE_PutJSONW() function passes the data in a UTF-16-encoded string. The functions otherwise operate identically.

The JSON functions operate on JSON values. The values written and read must be a valid JSON document that matches the sub-schema of the given URI.

Entering the UTF-8 Address Data

The following sample code shows the structure of the IDVE_PutJSON() function:

```
IDVE_EXPORTCALL1 IDVE_StatusCode IDVE_EXPORTCALL2 IDVE_PutJSON(
const char* const kpksCustomerID,
const char* const kpksURI,
const char* const kpksValue,
const IDVE_U64 ku64ValueLength,
const char* const kpksValueFilePath,
char* const kpsExtStatusMsg
);
```

The following table shows the parameter definitions for the IDVE_PutJSON() function:

Parameter	Operation	Comment
const char* const kpksCustomerID	[in]	Pointer to the zero-terminated 7-bit ASCII-encoded customer ID. The value might not be NULL.
const char* const kpksURI	[in]	Pointer to the zero-terminated 7-bit ASCII URI, for example " `AV/v1/Jobs/0/IO/Inputs/0`". The value might not be NULL.

Parameter	Operation	Comment
const char* const kpksValue	[in]	Pointer to the UTF-8-encoded JSON value. For the process command, the string can contain an input address in JSON format or can be NULL. The value must be NULL if the kpksValueFilePath is not NULL.
const IDVE_U64 ku64ValueLength	[in]	Length of the JSON value in number of code units, excluding any terminating zero. The value can be string length or IDVE_AUTOLEN. Ignored if kpksValue is NULL.
const char* const kpksValueFilePath	[in]	Pointer to the file path which refers to a JSON-formatted file to use as the value, encoded as UTF-8 or UTF-16. For the process command, the path identifies a file that contains the set of addresses that the job will process. The value must be NULL if kpksValue is not NULL.
char* const kpsExtStatusMsg	[out]	Pointer to a buffer of size IDVE_EXT_STATUS_MSG_BUFFER_SIZE for an optional extended status message. The value can be NULL.

Entering the UTF-16 Address Data

The following sample code shows the structure of the IDVE_PutJSONW() function:

```
IDVE_EXPORTCALL1 IDVE_StatusCode IDVE_EXPORTCALL2 IDVE_PutJSONW(
const char* const kpksCustomerID,
const char* const kpksURI,
const IDVE_WChar* const kpksValue,
const IDVE_U64 ku64ValueLength,
const char* const kpksValueFilePath,
char* const kpsExtStatusMsg
);
```

The following table shows the parameter definitions for the IDVE_PutJSONW() function:

Parameter	Operation	Comment
const char* const kpksCustomerID	[in]	Pointer to the zero-terminated 7-bit ASCII-encoded customer ID. The value might not be NULL.
const char* const kpksURI	[in]	Pointer to the zero-terminated 7-bit ASCII URI, for example " `AV/v1/Jobs/0/IO/Inputs/0`". The value might not be NULL.
const IDVE_WChar* const kpksValue	[in]	Pointer to the UTF-16-encoded JSON value. For the process command, the string can contain an input address in JSON format or can be NULL. The value must be NULL if the kpksValueFilePath is not NULL.
const IDVE_U64 ku64ValueLength	[in]	Length of the JSON value in number of code units, excluding any terminating zero. The value can be string length or IDVE_AUTOLEN. Ignored if kpksValue is NULL.
const char* const kpksValueFilePath	[in]	Pointer to the file path which refers to a JSON-formatted file to use as the value, encoded as UTF-8 or UTF-16. For the process command, the path identifies a file that contains the set of addresses that the job will process. The value must be NULL if kpksValue is not NULL.
char* const kpsExtStatusMsg	[out]	Pointer to a buffer of size IDVE_EXT_STATUS_MSG_BUFFER_SIZE for an optional extended status message. The value can be NULL.

Requesting the Process

When you submit an address, use the IDVE_Post() or IDVE_PostW() function to trigger the verification of the address.

Append the `process` command to the URI of the corresponding job object string to verify the address:

```
IDVE_StatusCode i32StatusCode = IDVE_Post( "", "AV/v1/Jobs/0/process", NULL, 0, NULL,
sExtStatusMsg );
```

The element "0" in the URI string indicates the value of the sJobURI parameter returned during the job creation operation. The value is the UID of the job.

Address Verification uses the process command to find a function server instance and assign the job to it. The function server verifies the address and writes the result back to the job data. The method blocks the call until the result is available.

If a function server is not available, the call waits until one becomes available and then performs the verification.

Note: The steps to create and run a job both use the Post function. Refer to "Creating a Job" on page 20 for information on the structure of the IDVE_Post() or IDVE_PostW() function.

Retrieving the Job Results

The operations to retrieve the output data and to submit the input data are similar. To fetch the verified addresses, use the IDVE_GetJSON() or IDVE_GetJSONW() function. The IDVE_GetJSON() function returns the data in a UTF-8-encoded string, whereas the IDVE_GetJSONW() function returns the data in a UTF-16-encoded string. The functions otherwise operate identically. The JSON functions operate on JSON values.

Retrieving the Job Results in the UTF-8 Encoding

The following sample code shows the structure of the IDVE_GetJSON() function:

```
IDVE_EXPORTCALL1 IDVE_StatusCode IDVE_EXPORTCALL2 IDVE_GetJSON(
   const char* const kpksCustomerID,
   const char* const kpksURI,
   char* const kpsValueBuffer,
   const IDVE_U64 ku64ValueBufferSize,
   IDVE_U64* const kpu64SizeWritten,
   IDVE_Bool* const kpbIsValuePresent,
   char* const kpsExtStatusMsg
);
```

The following table shows the parameter definitions of the IDVE_GetJSON() function:

Parameter	Operation	Comment
const char* const kpksCustomerID	[in]	Pointer to the zero-terminated 7-bit ASCII-encoded customer ID. The value might not be NULL.
const char* const kpksURI	[in]	Pointer to the zero-terminated 7-bit ASCII URI, for example " `AV/v1/Jobs/0/IO/Outputs/0/Results/0/Variants/0`". The value might not be NULL.

Parameter	Operation	Comment
char* const kpsValueBuffer	[out]	Pointer to the output buffer that receives the UTF-8-encoded JSON value and the terminating zero. The JSON value may be an output address, or it may be the path to the file that contains the set of output addresses.
const IDVE_U64 ku64ValueBufferSize	[in]	Size of the output buffer in code units, including the terminating zero.
IDVE_U64* const kpu64SizeWritten	[out]	Pointer to an unsigned 64-bit integer value that receives the size of the output written in code units, excluding the terminating zero. The value can be NULL.
IDVE_Bool* const kpbIsValuePresent	[out]	If you provide a valid pointer for kpbIsValuePresent, the function will indicate if the value at the given URI is present by setting kpbIsValuePresent to true or false. **Note:** If you provide NULL for kpbIsValuePresent and the JSON value that you want to retrieve does not exist, the function call reports an error. If you provide NULL for kpbIsValuePresent and the JSON value exists, it is retrieved normally.
char* const kpsExtStatusMsg	[out]	Pointer to a buffer of size IDVE_EXT_STATUS_MSG_BUFFER_SIZE for an optional extended status message. The value can be NULL.

Retrieving the Job Results in the UTF-16 Encoding

The following sample code shows the structure of the IDVE_GetJSONW() function:

```
IDVE_EXPORTCALL1 IDVE_StatusCode IDVE_EXPORTCALL2 IDVE_GetJSONW(
const char* const kpksCustomerID,
const char* const kpksURI,
IDVE_WChar* const kpsValueBuffer,
const IDVE_U64 ku64ValueBufferSize,
IDVE_U64* const kpu64SizeWritten,
IDVE_Bool* const kpbIsValuePresent,
char* const kpsExtStatusMsg
);
```

The following table shows the parameter definitions of the IDVE_GetJSONW() function:

Parameter	Operation	Comment
const char* const kpksCustomerID	[in]	Pointer to the zero-terminated 7-bit ASCII-encoded customer ID. The value might not be NULL.
const char* const kpksURI	[in]	Pointer to the zero-terminated 7-bit ASCII URI, for example " `AV/v1/Jobs/0/IO/Outputs/0/Results/0/Variants/0`". The value might not be NULL.
IDVE_WChar* const kpsValueBuffer	[out]	Pointer to the output buffer that receives the UTF-16-encoded JSON value and the terminating zero. The JSON value may be an output address, or it may be the path to the file that contains the set of output addresses.
const IDVE_U64 ku64ValueBufferSize	[in]	Size of the output buffer in code units, including the terminating zero.

Parameter	Operation	Comment
IDVE_U64* const kpu64SizeWritten	[out]	Pointer to an unsigned 64-bit integer value that receives the size of the output written in code units, excluding the terminating zero. The value can be NULL.
IDVE_Bool* const kpbIsValuePresent	[out]	If you provide a valid pointer for kpbIsValuePresent, the function will indicate if the value at the given URI is present by setting kpbIsValuePresent to true or false. **Note:** If you provide NULL for kpbIsValuePresent and the JSON value that you want to retrieve does not exist, the function call reports an error. If you provide NULL for kpbIsValuePresent and the JSON value exists, it is retrieved normally.
char* const kpsExtStatusMsg	[out]	Pointer to a buffer of size IDVE_EXT_STATUS_MSG_BUFFER_SIZE for an optional extended status message. The value can be NULL.

Example: Retrieving the Job Results

The following example shows an IDVE_GetJSON() function call to retrieve a job result:

```
char sJSON[ 50 * 1024 ];
char sExtStatusMsg[ IDVE_EXT_STATUS_MSG_BUFFER_SIZE ];
IDVE_StatusCode i32StatusCode = IDVE_GetJSON( "", "AV/v1/Jobs/0/IO/Outputs/0", sJSON,
sizeof( sJSON ), NULL, NULL, sExtStatusMsg );
```

The URI value that you pass to the IDVE_GetJSON() function is the job URI returned by the IDVE_Post call and the sub-URI pointing to the first output of the job. An implementation typically concatenates the URI fragments dynamically.

The output is not a URI, but rather a complete JSON document in UTF-8 format.

Retrieving String Data Values

You can use the IDVE_GetString() and IDVE_GetStringW() functions to retrieve string data values in the address as string data types. Use the IDVE_GetString() to retrieve the string data in the UTF-8 encoding. Use the IDVE_GetStringW() function to retrieve the string data in the UTF-16 encoding.

Consider the following rules and guidelines when you use IDVE_GetString() or IDVE_GetStringW():

- The IDVE_GetString() function uses the same structure as the IDVE_GetJSON() function. To use the IDVE_GetString() function, replace IDVE_GetJSON with IDVE_GetString in the structure.
- The IDVE_GetStringW() function uses the same structure as the IDVE_GetJSONW() function. To use the IDVE_GetStringW() function, replace IDVE_GetJSONW with IDVE_GetStringW in the structure.
- For IDVE_GetString() and IDVE_GetStringW(), the URI must point to a terminal property of type string, in contrast to IDVE_GetJSON, where the URI may point to a terminal or non-terminal property of any type.

Retrieving Boolean and Integer Values

To retrieve individual integer or boolean values, for example ResultQuality values, use the IDVE_GetInt32(), IDVE_GetInt64(), or IDVE_GetBool() function.

Example: The IDVE_GetBool() function

The following sample code shows the structure of the IDVE_GetBool() function:

```
IDVE_EXPORTCALL1 IDVE_StatusCode IDVE_EXPORTCALL2 IDVE_GetBool(
   const char* const kpksCustomerID,
```

```
    const char* const kpksURI,
    IDVE_Bool* const kbValue,
    IDVE_Bool* const kpbIsValuePresent,
    char* const kpsExtStatusMsg
);
```

The following table shows the parameter definitions for the IDVE_GetBool() function:

Parameter	Operation	Comment
const char* const kpksCustomerID	[in]	Pointer to the zero-terminated 7-bit ASCII-encoded customer ID. The value might not be NULL.
const char* const kpksURI	[in]	Pointer to the zero-terminated 7-bit ASCII URI, for example " `AV/v1/Jobs/0/IO/Outputs/0/ResultInfo/ResultCountOverflow`". The value might not be NULL.
IDVE_Bool* const kbValue	[out]	Pointer to a boolean value that the user provides. The boolean receives a copy of the boolean value present at the given URI. The pointer must not be NULL.
IDVE_Bool* const kpbIsValuePresent	[out]	If you provide a valid pointer for kpbIsValuePresent, the function will indicate if the value at the given URI is present by setting kpbIsValuePresent to true or false. **Note:** If you provide NULL for kpbIsValuePresent and the boolean value that you want to retrieve does not exist, the function call will report an error. If you provide NULL for kpbIsValuePresent and the boolean value exists, it is retrieved normally.
char* const kpsExtStatusMsg	[out]	Pointer to a buffer of size IDVE_EXT_STATUS_MSG_BUFFER_SIZE for an optional extended status message. The value can be NULL.

Note: To use the IDVE_GetInt32() or IDVE_GetInt64() function, replace IDVE_GetBool with IDVE_GetInt32 or IDVE_GetInt64 in the sample structure.

Also, change the parameter name `IDVE_Bool* const kbValue` to `IDVE_I32* const kpi32Value` or `IDVE_I64* const kpi64Value` to point to 32-bit or 64-bit integer values respectively.

Retrieving Array Sizes

The IDVE_GetArraySize() function identifies the current size of an array in a JSON document. The function returns the size in integer format and can be used with URI strings that point to JSON arrays. For example, you can call IDVE_GetArraySize() on the URI "State/FileSets/FileSetA" to find out how many data files the engine has loaded from the FileSetA directory.

Example: The IDVE_GetArraySize() function

The following sample code shows the structure of the IDVE_GetArraySize() function:

```
IDVE_EXPORTCALL1 IDVE_StatusCode IDVE_EXPORTCALL2 IDVE_GetArraySize(
    const char* const kpksCustomerID,
    const char* const kpksURI,
    IDVE_U64* const kpu64NumItems,
    IDVE_Bool* const kpbIsValuePresent,
    char* const kpsExtStatusMsg
);
```

The following table shows the parameter definitions for the IDVE_GetArraySize() function:

Parameter	Operation	Comment
const char* const kpksCustomerID	[in]	Pointer to the zero-terminated 7-bit ASCII-encoded customer ID. The value might not be NULL.
const char* const kpksURI	[in]	Pointer to the zero-terminated 7-bit ASCII URI, for example "`AV/v1/Jobs/###/IO/Outputs/0/Results`". The value might not be NULL.
IDVE_U64* const kpu64NumItems	[out]	Pointer to an unsigned 64-bit integer value that receives the value of the output. The value might not be NULL.
IDVE_Bool* const kpbIsValuePresent	[out]	If you provide a valid pointer for kpbIsValuePresent, the function will indicate if the value at the given URI is present by setting kpbIsValuePresent to true or false. **Note:** If you provide NULL for kpbIsValuePresent and the array size that you want to retrieve does not exist, the function call will report an error. If you provide NULL for kpbIsValuePresent and the array size exists, it is retrieved normally.
char* const kpsExtStatusMsg	[out]	Pointer to a buffer of size IDVE_EXT_STATUS_MSG_BUFFER_SIZE for an optional extended status message. The value can be NULL.

Retrieving Optional Properties

The IDVE_IsValuePresent() function identifies any optional JSON property that has no default value in a JSON schema document. For example, if you want to know if the error log has been disabled because of an error, you can call IDVE_IsValuePresent() on the URI "`State/IDVEInformation/ErrorLog/ErrorCode`". You can use this function call to check for the presence of optional properties in a JSON document.

If you call the IDVE_IsValuePresent() function on a non-optional property, the function returns *true* as output.

Example: The IDVE_IsValuePresent() function

The following sample code shows the structure of the IDVE_IsValuePresent() function:

```
IDVE_EXPORTCALL1 IDVE_StatusCode IDVE_EXPORTCALL2 IDVE_IsValuePresent(
   const char* const kpksCustomerID,
   const char* const kpksURI,
   IDVE_Bool* const kbValue,
   char* const kpsExtStatusMsg
);
```

The following table shows the parameter definitions for the IDVE_IsValuePresent() function:

Parameter	Operation	Comment
const char* const kpksCustomerID	[in]	Pointer to the zero-terminated 7-bit ASCII-encoded customer ID. The value might not be NULL.
const char* const kpksURI	[in]	Pointer to the zero-terminated 7-bit ASCII URI, for example "`AV/v1/Jobs/###/IO/Outputs/0/Results/0/Variants/0/AddressElements/AdministrativeDivision`". The value might not be NULL.

Parameter	Operation	Comment
IDVE_Bool* const kbValue	[out]	Pointer to a boolean value to be retrieved by the function call. If the pointer is not NULL, it indicates if the value at the given URI is present or not.
char* const kpsExtStatusMsg	[out]	Pointer to a buffer of size IDVE_EXT_STATUS_MSG_BUFFER_SIZE for an optional extended status message. The value can be NULL.

Retrieving the State of the Engine

To retrieve the current state of the data verification engine, use an IDVE_GetJSON function call and pass the URI value in the function call as *State*.

The following sample code shows an IDVE_GetJSON() call that can retrieve the engine state:

```
char sJSON[ 50*1024];
        char sExtStatusMsg[ IDVE_EXT_STATUS_MSG_BUFFER_SIZE];
        IDVE_StatusCode i32StatusCode= IDVE_GetJSON("", "State", sJSON, sizeo
        f(sJSON), NULL, NULL, sExtStatusMsg);
```

The function call returns the current values on the properties in the `IDVEState.schema.json` file.

The elements and properties in the *IDVEState.json* file are organized in the following groups:

- DateTime.
- SystemInformation.
- IDVEInformation.
- Config.
- Licenses.
- Functions.
- FileSets.

You can return the elements and properties within a single group by calling the IDVE_GetJSON function and appending the group name to the URI path. For example, to return the current Config information, call IDVE_GetJSON with the URI value *State/Config*.

Note: The Config group elements and properties map to the elements and properties in the `IDVEConfig.json` file. For more information about the engine configuration properties, see the *Installation and Getting Started Guide*.

Clearing the Input and Output Data

To clear the input and output data, call the IDVE_Delete() function. Supply a URI that points to the "IO" part of the JSON schema of the corresponding job object, for example "AV/v1/Jobs/0/IO". Deleting IO does not affect the parameters of the job.

The following sample code shows the structure of the IDVE_Delete() function:

```
IDVE_EXPORTCALL1 IDVE_StatusCode IDVE_EXPORTCALL2 IDVE_Delete(
const char* const kpksCustomerID,
```

```
const char* const kpksURI,
char* const kpsExtStatusMsg
);
```

The following table shows the parameter definitions of the IDVE_Delete() function:

Parameter	Operation	Comment
const char* const kpksCustomerID	[in]	Pointer to the zero-terminated 7-bit ASCII-encoded customer ID. The value might not be NULL.
const char* const kpksURI	[in]	Pointer to the zero-terminated 7-bit ASCII URI, for example "`AV/v1/Jobs/0/IO`". The value might not be NULL.
char* const kpsExtStatusMsg	[out]	Pointer to a buffer of size IDVE_EXT_STATUS_MSG_BUFFER_SIZE for an optional extended status message. The value can be NULL.

After you clear the input and output data, you can feed in the next request with the same job object.

Deleting a Job Object

When you no longer need a job object, delete the job with the IDVE_Delete() function. Pass the URI that the engine returned with the IDVE_Post() or IDVE_PostW() function when you created the job object.

CHAPTER 3

Address Structure and Elements

This chapter includes the following topics:

Address Structure and Elements Overview

Informatica Address Verification makes use of a range of input and output address elements. Select the appropriate input fields based on the input address structure. Select the output fields based on the result that you want to achieve. Use the `AVJob.schema.json` file as a guide to the address elements and properties you can use.

You choose the verification modes and the database files that you need to verify an address. You can assign the default country for an input address and identify the country information if the information is absent in an address.

Process Modes

The process mode defines the type of address processing that Informatica Address Verification performs.

You can specify one of the following process modes:

- AddressCodeLookup
- Batch
- Certified
- GeocodeToAddress
- Interactive

- QuickCapture

Note: The engine handles the process mode values in a case-sensitive manner. When you configure an address job, enter the process mode values as they appear in the list above. For example, do not enter BATCH as a process mode.

AddressCodeLookup Mode

The Address Verification reference data includes unique values that represent part of an address or the complete address. To return the value for an address, or to retrieve the address information that a value represents, configure Informatica Address Verification in address code lookup mode. You must also install the corresponding address code lookup (`ACL`) database with valid license keys.

In address code lookup mode, you can retrieve address information for the following countries:

- Austria
- Germany
- Japan
- Serbia
- South Africa
- United Kingdom

To configure Informatica Address Verification in address code lookup mode, specify the address code for the information that you require in the input. Find the AddressCode element of the LookupKeys element in the `AVJob.schema.json` file.

The AddresssCode element includes the following properties:

- Value. The address code value.
- Type. The type of address code.

When you submit the address code lookup values for processing, Address Verification returns the partial or complete addresses that correspond to the values.

Address Codes

The following table lists the address codes that you can add:

Country	Address Code	Description
Austria	AUT_PACID	An Austria address has a PAC ID value when the address receives mail at another address at the same location. The PAC ID value is the Postal Address Code (PAC) value of the corresponding address that receives the mail. For example, a PAC ID value of 100004254 returns the following address: Neue-Welt-Gasse 2 8010 Graz AUT **Note:** The address that gives access to the mailbox is the Ident address.
Germany	DEU_AGS	The Amtliche Gemeindeschlüssel (AGS) is a variable-length value that uniquely identifies a municipality in Germany. An AGS value might return more than one level of municipality information. See also DEU_StreetID.
Germany	DEU_LocalityID	The Locality ID is a variable-length value that uniquely identifies a German locality. See also DEU_StreetID.
Germany	DEU_StreetID	The Street ID is a variable-length value that uniquely identifies a German street address. For example, when you validate the following address: Röntgenstr. 9 67133 Maxdorf Germany Address Verification returns the following additional information in the address output: AGS: 07338018 LocalityID: 68015519 StreetID: 100560690
Japan	JPN_ChoumeiAzaCode	The Choumei Aza code is an 11-digit value that corresponds to a delivery point address in Japan. For example, a JPN_ChoumeiAzaCode of 28201160001 fetches the following result: 〒670-0081 兵庫県姫路市田寺東１丁目 or 01 Chome Taderahiga-shi Himeji-shi Hyogo-ken 670-0081 Japan You can use the Choumei Aza code in address code lookup mode to find the current version or an historical version of an address.

Country	Address Code	Description
Japan	JPN_ChoumeiAzaGaikuCode	The Choumei Aza code is an 11-digit value that defines a unique delivery point for Japan addresses. The Gaiku code is a four-digit value that identifies a city block (ban) in Japan. The combined Choumei Aza code and Gaiku code helps you to retrieve more accurate addresses from Address Verification.
Serbia	SRB_PAK	The Postal Address Code (PAK) is a six-digit value that defines a unique Serbian address to the street level. For example, a SRB_PAK value of 111411 fetches the following result: Braće Krsmanovića 13 11000 Beograd SRB
South Africa	ZAF_NADID	The National Address Database (NAD) ID is a unique numeric ID assigned to a South Africa street address. For example, a ZAF_NADID value of 2170232 fetches the following result: 4 Balmoral Road Vincent East London 5247 South Africa
United Kingdom	GBR_UDPRN	The Unique Delivery Point Reference Number (UDPRN) is an eight-character value that uniquely identifies a postal address in the Royal Mail PAF database. For example, a GBR_UDPRN value of 15511432 fetches the following result: Flat 16 Haden Court Lennox Road London N4 3HS United Kingdom
United Kingdom	GBR_UPRN	The Unique Property Reference Number (UPRN) is a numeric value that uniquely identifies a land or property unit in the United Kingdom. For example, a GBR_UPRN value of 151117706 fetches the following result: 2 Drumforber Cottage Laurencekirk AB30 1RS United Kingdom

Note: Address Verification returns a value of 0 for the ResultQuality and PostalRelevance status codes in address code lookup mode, because the lookup returns a partial address.

Address Verification returns a value of U as the Address Type in address code lookup mode.

Transaction Key and Record ID Values in Address Code Lookup Mode

If the address records in your organization contain transaction keys or record ID values, you can use the values in address code lookup mode. When you include the values in the input addresses, Address

Verification passes the values through to the output addresses. You can then enter a transaction key or record ID value in address code lookup mode to query the output addresses and to return the address that matches the value.

Use the SourceID to enter the values. You can find the input SourceID in the `AVJob.schema.json` file.

Batch Mode

Use batch mode to verify one or more addresses without any user intervention. In batch mode, Informatica Address Verification processes and verifies the addresses that you enter and returns the results when the processing is complete.

Choose batch mode when you want to enter multiple addresses for verification and receive one result for each input address. If you have multiple addresses to verify, you can create a text file that contains the addresses and provide the file as the input. You can submit 1000 addresses in a single call in batch mode.

In batch mode, Address Verification does not correct addresses that are ambiguous or difficult to correct without user intervention. If the reference address database for the country to which the input address belongs is not available, Address Verification does not verify the address.

Certified Mode

Use certified mode to verify addresses to the standards that the postal authority in a country specifies. You can use certified mode to verify addresses in Australia, Canada, France, New Zealand, and the United States.

The verification process in certified mode works in broadly the same way as verification in batch mode. However, certified mode applies additional rules and regulations to the verification process.

You can use certified mode to verify addresses in accordance with the following certification standards:

- Address Matching Approval System (AMAS) certification for Australia Post. Certified to AMAS Cycle 2020.
- Coding Accuracy Support System (CASS) certification for the United States Postal Service. Certified to CASS Cycle N.
  Informatica Address Verification also supports several requirements that the United States Postal Service proposes for CASS Cycle O.
- SendRight certification for New Zealand Post. Certified to SendRight Cycle 2020.
- Service National de L'Adresse (SNA) certification for La Poste of France. Certified to SNA Cycle 2020. Address Verification certifies addresses in France to the following levels:
  - CEDEX A. Organization level.
  - Hexacle. House-number level.
  - Hexaligne 3. Building level.
  - Hexaposte. Post code level.
  - Hexavia. Street level.
- Software Evaluation and Recognition Program (SERP) certification for Canada Post. Certified to SERP Cycle 2020.

Rules and Guidelines for Verification in Certified Mode

Consider the following rules and guidelines for address verification in certified mode:

- Certified mode applies all the certification standards that the postal authorities in Australia, Canada, France, New Zealand, and the United States define for address certification. The Address Verification 6.4 engine is formally certified for address verification by the postal authorities in Australia, New Zealand, and the United States.

- An address that you verify in certified mode must meet the formatting requirements that the postal authority of the country specifies. For example, to verify a Canada address in certified mode, the input address can include a maximum of two postal delivery address lines. To verify a France address in certified mode, the input address can include a maximum of six postal formatted address lines. Of the six lines in France addresses, three lines must contain delivery elements other than the post box and locality information.
- For certified mode, you must install the certified database files with valid license keys. You can submit 1000 addresses in a single call in certified mode.
- Certified mode falls back to batch mode if the address belongs to a country that Address Verification does not support in certified mode.
- If you enable certified mode and do not have the corresponding certified reference address database, Address Verification returns an N error status.

Certified Mode Verification for Australia Addresses

To verify Australia addresses in certified mode, install the `AUS_ADV_VRF_C01_000_6_1_0.MD6` database with valid license keys.

The `AUS_ADV_VRF_C01_000_6_1_0.MD6` database contains Postal Address File (PAF) data, which includes Delivery Point Identifiers (DPIDs) from Australia Post.

Certified Mode Verification for Canada Addresses

To verify Canada addresses in certified mode, install the `CAN_ADV_VRF_C01_000_6_1_0.MD6` database with valid license keys. The `CAN_ADV_VRF_C01_000_6_1_0.MD6` database contains the PoCAD (Point of Call Address Data).

Address Verification is SERP 2020 compliant. SERP 2020 compliance ensures that Address Verification adheres to the following changes to the postal rules and regulations set by Canada Post:

- If the input suite number is outside the only range available for the address in PoCAD, Address Verification marks that address as not valid. When the input postal code maps to a Large Volume Receiver (LVR), Address Verification copies the suite number from the input address to the output address even if the suite number does not match any database entry that contains the correct single suite-civic number combination.
- If the range-based PoCAD has only one address associated with a civic street and if the input address does not match the address in the database, Address Verification marks that address as not valid or noncorrectable.
- If the range-based PoCAD contains a Type 2 record that does not have a route identifier or a delivery mode identifier for a rural address, Address Verification handles that address in the same way that it handles Type 1 addresses. However, the following conditions apply to the handling of rural civic addresses:
  - If the input address does not have a match in the range-based PoCAD and the postal code of the input address has a corresponding Type 4 address in the range-based PoCAD, Address Verification marks the address as VQ (Valid but questionable) in the SERP category enrichment.
  - If the input address is a rural address with a street that has no civic street number, Address Verification adds the civic street number when a unique correction is possible. If no unique civic street can be added to the input address, Address Verification rejects the address.

Note that Post Office Box numbers from 99900 through 99905 in Canada denote Deliver to Post Office (DTPO) addresses for retail outlet locations. Addresses with Post Office Box numbers from 99900 through 99905 are specifically meant for parcel delivery and should not be used for other mail items.

Certified Mode Verification for New Zealand Addresses

To verify New Zealand addresses in certified mode, install the `NZL_ADV_VRF_C01_000_6_1_0.MD6` database with valid license keys.

Certified Mode Verification for France Addresses

To verify France addresses in certified mode, install the `FRA_ADV_VRF_BIA_000_6_1_0.MD6` database with valid license keys.

For SNA certified processing, enter addresses in the six-line PostalFormattedAddressLines format, including empty lines wherever a part of the address is missing.

The following table lists the values for the six-line PostalFormattedAddressLines format:

Postal Formatted Address Lines	Value
Line 1	ORGANIZATION IDENTIFICATION or IDENTITY OF THE ADDRESSEE
Line 2	INDIVIDUAL IDENTIFICATION (Company Contact) or DELIVERY POINT ACCESS INFORMATION (SubBuilding)
Line 3	DELIVERY POINT LOCATION (Building)
Line 4	STREET NUMBER or PLOT and THOROUGHFARE
Line 5	DELIVERY SERVICE or THOROUGHFARE COMPLEMENTARY IDENTIFICATION
Line 6	POSTCODE and LOCALITY or CEDEX POSTCODE and DISTRIBUTION AREA INDICATOR

SNA certification supports 7-bit ASCII data, and certified output does not include diacritics or mixed-case values.

Note: Informatica also provides the `FRA_ADV_VRF_BIA_001_6_1_0.MD6` file, which you can use to deliver addresses with diacritics and in mixed case in batch and interactive modes. If the 001 data file is not available, batch and interactive processing can succeed if the 000 data file is present.

The `FRA_ADV_VRF_BIA_000_6_1_0.MD6` file is the default file for France in the batch and interactive file data set.

Certified Mode Verification for United States Addresses

To verify United States addresses in certified mode, install the United States `Cxx` databases. The `Cxx` databases are numbered `USA_ADV_VRF_C01_000_6_1_0.MD6` through `USA_ADV_VRF_C07_000_6_1_0.MD6`. The databases enable the address verification process to analyze and update different aspects of the address according to the different USPS postal criteria. Add the licenses for the files.

Consider the following rules and guidelines for certified verification in the United States:

- Certified mode verification of United States addresses is available only to customers in the United States.
- The CASS certification files comply with the SHA-256 standard.
- The DPV database has information that helps you check whether a ZIP+4 coded address is in the USPS delivery file as a known address record. You can use the DPV product to confirm known USPS addresses and to identify potential addressing issues that might affect delivery.

Note: Address Verification can also add ZIP+4 Codes to addresses in batch and interactive mode. Address Verification reads the codes from the `USA_ADV_VRF_BIA_000_6_1_0.MD6` database in batch and interactive modes.

- Informatica Address Verification sends records with input suite data that does not match the ZIP+4 Code reference data thorough the SuiteLink process. If Address Verification finds a match for such addresses with SuiteLink, Address Verification retains the input suite data in the residue component and writes the correct suite data on the second postal delivery address line. The USPS requires that the address verification operation retains the non-matching input data.

GeocodeToAddress Mode

Use GeocodeToAddress mode to retrieve addresses at or near a set of geocoordinates that you submit.

Address Verification applies a search radius to the input geocoordinates to define the area in which to search for addresses. The default radius is 50 meters.

Address Verification returns the addresses closest to the geocoordinates within the search radius, up to the limit that the MaxResultCount property defines. Address Verification also returns the distance from the geocoordinates of each output address and the direction in which the address lies.

GeocodeToAddress mode requires arrival point reference data and returns addresses with arrival point precision.

For more information see "Reverse Geocoding" on page 69.

Interactive Mode

Use interactive mode to manually resolve ambiguities and select the best possible corrections. In interactive mode, Informatica Address Verification returns multiple address suggestions from which you can choose the most appropriate result.

For each address that you enter for verification in interactive mode, you can receive a maximum of 100 suggestions from Address Verification.

For information about how to specify the number of address suggestions for an address, see "Maximum Number of Results in an Address Job" on page 72.

QuickCapture Mode

You can configure QuickCapture mode to enable your end users to receive address suggestions as they type. When the end user enters an address or part of an address, the Address Verification engine returns all candidate matches from the reference data as suggestions to your application.

If address verification finds one or more unique matches for the input address, the end user can select a unique address in your application and the process ends.

If Address Verification finds non-unique matches, such as street address that supports a range of house numbers, the end user can drill down on the address or type additional information. The process continues until the end user selects a single unique address.

QuickCapture Configuration

QuickCapture mode supports single-line data input. Use the SingleAddressLine element and the Country element to retrieve address suggestions. You can receive up to 100 suggestions for an input address in QuickCapture mode.

You can specify the maximum number of suggestions that Informatica Address Verification returns for each partial or complete address that you enter in QuickCapture mode. For information about how to specify the number of address suggestions, see "Maximum Number of Results in an Address Job" on page 72

In QuickCapture mode, Address Verification ignores the input address delimiter, and returns address with a character space delimiter.

To achieve better output, use a character space as the delimiter in the input address. You don't need to update the single-line input properties for character spaces. Use the SingleAddressLineDelimiter element to set the output delimiter. The default output delimiter for elements in QuickCapture mode is the semicolon.

You can set the output delimiter to one of the following values:

- Comma
- Semicolon
- Tab

Address Selection Strategies

You can configure QuickCapture mode to return address suggestions in the following ways:

Return addresses based on user input

By default, QuickCapture mode returns addresses from the reference data that match the current data that the user enters.

If QuickCapture mode returns one or more unique addresses, the user can optionally select a unique address and the selection process ends. If the user enters additional data, the data entry triggers another call to the address verification engine, and the engine returns addresses that match the latest input. In this manner, the engine refines the suggestions that it returns based on the user data entry.

For example, if QuickCapture mode returns a house number range such as 11..20 MAIN STREET, the user might enter an address with a house number in that range. QuickCapture mode can then return additional suggestions or a complete address that the user can select.

Return addresses based on RangesToExpand and RangeExpansionType parameter selection

QuickCapture mode does not depend on the RangesToExpand or RangeExpansionType parameter default values.

If the RangesToExpand parameter is set to None, then QuickCapture mode uses its own smart expansion property and returns address ranges.

For example, QuickCapture mode returns a house number range such as 11..20 MAIN STREET.

If the RangesToExpand parameter is set to All or Only Valid and RangeExpansionType parameter is set to Full, then QuickCapture mode expands the addresses in each range to complete addresses up to the maximum permitted address count.

For example, QuickCapture mode returns every house number such as 11, 12, 13, 14, 15, 16, 17, 18, 19, 20 MAIN STREET.

If the RangesToExpand parameter is set to All or Only Valid and RangeExpansionType parameter is set to Flexible, then QuickCapture mode expands the addresses in each range to complete addresses up to the final address allowed by the address count properties. QuickCapture mode summarizes the remaining ranges for the input address in the space available for the final address.

For example, QuickCapture mode returns every house number such as 11, 12, 13, 14, 15, 16, 17..20 MAIN STREET.

Drill down on address ranges

QuickCapture mode can permit the end user to click on a non-unique address suggestion and obtain the list of addresses that the address identifies. The user needs to enter address data once. After the initial data entry, the user can select a non-unique address to drill down on the addresses that it represents.

The drill-down process can continue through multiple levels of address suggestions.

To enable the drill-down capability, add the QuickCaptureResultFilter property to your engine integration. Find the property under **Parameters** > **Country Sets** > **Result**.

If you do not implement the QuickCaptureResultFilter property, the engine uses the default strategy.

The QuickCaptureResultFilter property is an array of integers where every integer represents an index to a non-unique address suggestion. Configure your software application to read the integers on the property. When the end user selects a non-unique suggestion, the application returns the addresses that the integer represents.

Note: If the user enters additional data while clicking through the drill-down suggestions, the indexes will not match the results for the latest input. In this case, the application must clear the QuickCaptureResultFilter array values and restart the click-through process with the latest address results.

Sorting Criteria

At the highest level, Address Verification sorts address suggestions by the type of address element in each suggestion that most closely matches the input address values. The ResultGroup property in the `AVJob.schema.json` file identifies the elements.

Within each group, QuickCapture mode considers the following factors when it sorts and returns address suggestions:

- The similarity between the input address and the reference address, as indicated by the MatchPercentage value.
- The distance between the input address and the reference address, if the distance can be calculated.

If you provide an IP address or a set of geocoordinates in the input address, the locality that the IP address or the geocoordinates identify becomes the primary sorting criterion within each result group.

Address Verification returns a complete address in suggestions when the address type is S. For other address types, Address Verification may return a partial address that includes the information that links it to a result group.

For more information about the ResultGroup property, see "Result Group" on page 107.

Locality Order in QuickCapture Output

Address Verification can sort the address suggestion list according to the town or city to which the addresses belong, beginning with the addresses in the nearest town or city. Address Verification sorts the addresses by town or city when you include an IP address or a set of geocoordinates in the input address.

You can submit an IP address in the IPv4 format or the IPv6 format. When you submit IP address data with a partial address, Address Verification converts the IP address to a geocode and uses the geocode to select the town or district in which the address is located. Find the property under **Parameters** > **IO** > **Inputs** > **IPAddress** in the `AVJob.schema.json` file.

You can enter geocoordinates as a single input value, or you can enter the latitude and longitude coordinates as discrete input values. Find the properties under **Parameters** > **IO** > **Inputs** > **Geocode** in the `AVJob.schema.json` file.

If you enter both IP address and geocoordinate values for an address that you verify in QuickCapture mode, Address Verification gives precedence to the geocoordinates over the IP address.

Rules and Guidelines for QuickCapture Mode

Consider the following rules and guidelines when you submit an address for verification in QuickCapture mode:

- QuickCapture mode reads the SingleAddressLine element, the Country element, and optionally the SourceID element. Enter country information in the Country element and not in the SingleAddressLine element.
- Ensure that you fully preload the quick capture reference data files before you submit an address in QuickCapture mode.
- QuickCapture mode can process an incomplete address. For best results, submit a complete address.

  QuickCapture mode can tolerate a mis-spelling in an address element in a manner that batch and interactive modes cannot. For example, QuickCapture mode can recognize 1 FIFTH AVE NEW YO as 1 FIFTH AVE NEW YORK.
- When you submit an address to QuickCapture mode in a Chinese or Japanese character set, use a character space as a delimiter between the elements.
- Do not separate associated address values. For example, do not add a locality name between a street name and its associated descriptor.
- Do not enter a child address element without the corresponding parent element. For example, do not enter a house number without the corresponding street name.
- QuickCapture mode may return partial addresses when all elements are not provided on input or when there are ranges in the output that do not uniquely identify an address.
- QuickCapture mode supports ZIP+4 Codes in the United States.

  If Address Verification returns United States address suggestions that do not identify single or multiple addresses with the same four-digit ZIP code suffix, Address Verification returns ".." in place of the four-digit ZIP code suffix and records an AddressCount value greater than 1 or a different value.
- Address Verification expands any house number, building number, and postal code number ranges in the addresses that it returns in a suggestion list in QuickCapture mode. Address Verification performs range expansion from the first eligible address in the suggestion list and can continue to the limit that the MaxResultCount property defines.

  Address Verification applies preset values to the NumericRangeExpansion parameter properties in QuickCapture mode. For more information about range expansion, see "NumericRangeExpansion" on page 73.

Country Determination

The CountryDetermination element specifies how Informatica Address Verification handles the country information in the input address. Find the CountryDetermination element in the `AVJob.schema.json` file.

Address Verification can identify the country to which an address belongs from the information in your address.

You can specify the input fields that you want Address Verification to consider when it searches for country information. You add the input field names in the SearchInFields property.

You can specify one or more of the following fields on the SearchInFields property:

- Country
- PostalFormattedAddressLines

Note: The order in which you specify the fields on the SearchInFields property determines the order in which Address Verification searches for the country information.

You can also set a default country that Address Verification can apply to input addresses that do not contain country information. To set a default country, configure the DefaultCountries property in the CountryDetermination element.

If you leave the SearchInFields property empty, Address Verification does not identify the country information from the input address and returns the country that you specified in the DefaultCountries property.

Note: The country information in the country input field takes precedence when Address Verification can infer multiple countries from the input address.

Address Values in the AddressElements Object

Informatica Address Verification provides you with address elements that you can use to map an input address that contains each item of address information in a separate field. You can also use the address elements to configure the output structure of an address.

You can find the address input elements in the `AVJob.schema.json` file under **IO** > **Inputs**.

You can find the address output elements in the `AVJob.schema.json` file under **IO** > **Outputs** > **Results** > **Variants**.

Address Line Elements and Formatted Data

Informatica Address Verification provides you with address line elements that you can use to map an input address that contains multiple address information in a single field. You can also use the address line elements to configure the output structure of an address.

Informatica Address Verification supports the following address line elements:

PostalRecipientLines

Contains recipient information, such as contact or organization name. Address Verification supports up to six lines of recipient information.

PostalDeliveryAddressLines

Contains delivery information, such as street, house number, building, and delivery service information. Address Verification supports up to six lines of postal delivery address line information.

PostalFormattedAddressLines

Contains unfielded data such as recipient information, delivery address information, and locality information. Address Verification supports up to 19 lines of postal formatted address line information.

PostalLocalityLine

Contains information such as locality, postal code, province, and country details. Address Verification supports up to six lines of postal locality line information.

SingleAddressLine

Contains an address in a single line. You can submit fielded or fielded data in the SingleAddressLine element.

StreetWithNumber

Contains street and house number information in a single element. For example, 1 MAIN ST.

NumberAndSubBuilding

Contains number and sub-building information in a single element. For example, 1 #1.

Input Options for Single Address Line Data

By default, Informatica Address Verification does not recognize a delimiter in single-line input. To specify a delimiter, update the single-line input properties for the address.

Find the properties under **Parameters** > **CountrySets** > **Input** in the `AVJob.schema.json` file.

Note: When you submit address data on a single line, use a character space as a separator between the values.

The Input element has the following properties:

SingleAddressLineDataType

Determines whether Address Verification expects a delimiter in single-line data.

The property takes one of the following values:

- DelimitedMultiLine. Indicates that the single-line input uses delimiters to identify address fields. When you set the property to DelimitedMultiLine, Address Verification uses the SingleAddressLineDelimiter property value to identify the field delimiter.
- UndelimitedSingleLine. Indicates that the single-line input does not use delimiters. When you set the property to UndelimitedSingleLine, Address Verification ignores the SingleAddressLineDelimiter property value.

The default value is UndelimitedSingleLine.

SingleAddressLineDelimiter

Identifies the delimiter that Address Verification expects when you set the SingleAddressLineDataType property to DelimitedMultiLine.

The property takes one of the following values:

- Comma
- Semicolon
- Tab

The default value is Comma.

Example: Multiline Address in a Single-Line Field

The following United States address includes a semicolon as a delimiter between the street information and the city, state, and ZIP Code information:

1960 W CHELSEA AVE STE 2006R; ALLENTOWN PA 18104

If you submit the address with the delimiter, configure the SingleAddressLineDataType and SingleAddressLineDelimiter properties to expect the semicolon. If you omit a delimiter, you do not need to update the properties.

Reference Data Specification

When you run an address job, Informatica Address Verification uses the country information in the input addresses and the process mode that the job specifies to select the appropriate reference data file for each address. Address Verification selects the file from the files that are loaded into memory for the current job.

You can additionally use the DataSet array on the DataSetDetermination parameter to specify the reference data files that Address Verification can use for an address job.

Use a DataSet array when more than one reference data file meets the country and process mode requirements that the input address defines. If you configure a DataSet array, Address Verification uses only the files that the array identifies in the current address job.

Find the DataSetDetermination element in the `AVJob.schema.json` file.

Configuring the DataSet Array

Use a DataSet array to specify a single reference data file that Address Verification can use for a given country and process mode.

Populate the DataSet array with the following values:

- The three-character ISO code for the country to which each address belongs. Enter the ISO code on the *Country* property.
- The three-digit identifier of the reference data file. Enter the identifier on the *ID* property.
- The type of reference data that the file supports, for example BatchInteractive. Enter the type on the *SubType* property.

  You can enter the following SubType values:

  - AddressCodeLookup
  - BatchInteractive
  - CAMEO
  - Certified
  - GeocodeToAddress
  - GeocodingArrivalPoint
  - GeocodingRooftop
  - GeocodingStandard
  - QuickCapture
  - Supplementary

Rules and Guidelines for Dataset Arrays

Consider the following rules and guidelines when you configure the Dataset array:

- The DataSet array values are optional in the following cases:
  - Address Verification loaded a single data file into memory for the current job.

    If you do not populate any DataSet array and a single data file is loaded into memory, Address Verification uses the available file regardless of the file name.
  - Address Verification loaded multiple data file into memory, and a single file includes the default data set ID in the file name.

    If you do not populate any DataSet array, and if the loaded files include a single file with the default ID, Address Verification uses the file with the default ID.

- When you use a DataSet array, populate all of the values on the array.
- If a DataSet array specifies a reference data file for an address but the file is not loaded into memory, Address Verification does not process the address.
- You can add multiple arrays to DatasetDetermination element.
- A DataSet array may specify a data set that the address job does not use. In such cases, the job ignores the data set.

Defining an Address Job with a Single Data Set

The following code fragment shows a single data set in the DataSet array:

```
"DataSetDetermination": {
      "DataSet": [
         {
            "Country": "AUS",
            "ID": "002",
            "SubType": "BatchInteractive"
         }
      ]
   }
```

In this example, Address Verification will search the files in memory for an Australia reference data file with the sub type BatchInteractive and a data set ID of 002.

Defining an Address Job with Multiple Data Sets

The following code fragment shows a sample of multiple data sets in the DataSet array:

```
{
   "DataSet": [
      {
         "Country": "IRL",
         "ID": "001",
         "SubType": "BatchInteractive"
      },
      {
         "Country": "AUS",
         "ID": "001",
         "SubType": "BatchInteractive"
      }
   ]
}
```

In this example, Address Verification can search the files in memory for the following data files:

- A batch and interactive data file for Ireland with a data set ID of 001.
- A batch and interactive data file for Australia with a data set ID of 001.

JSON Output

You can configure the OutputDetail element to enable or disable certain output values in the output. Find the OutputDetail element in the `AVJob.schema.json` file.

Preformatted Data

You can set the following properties of the PreformattedData element to true or false:

- PostalFormattedAddressLines. The default value is true.
- PostalDeliveryAddressLines. The default value is true.

- PostalRecipientLines. The default value is true.
- PostalLocalityLine. The default value is true.
- StreetWithNumber. The default value is false.
- NumberAndSubBuilding. The default value is false.
- SingleAddressLine. The default value is false.
  For information about the SingleAddressLineDelimiter property of the PreformattedData element, see "QuickCapture Mode" on page 39.

SubItems and Status

You can set the SubItems, ElementStatus, and OutputStatusForEmptyElements properties of the OutputDetail element to true or false. The default value is false.

Part II: Parameter and Element Reference

This part contains the following chapters:

CHAPTER 4

Address Enrichments

This chapter includes the following topics:

Enrichments Overview

Address enrichments provide additional information that helps you better understand and use the address data. The enrichments include data that increases the precision of the address, provides additional information about the administrative or statistical regions to which the address belongs, and helps the postal service to find the destination mailbox more easily.

For example, Informatica Address Verification can return geocoordinates that identify the physical location of an address. You can determine the type of geocoordinates that Address Verification returns. For more information, see "Geocodes" on page 52.

Some enrichments are specific to addresses in individual countries, and some enrichments apply equally to every country. For more information, see "Country-Specific and Global Enrichments" on page 54.

CAMEO Consumer Segmentation

CAMEO profiles from the TransUnion Information Group provide consumer classification systems that indicate the socio-economic and geo-demographic profiles of neighborhoods across the world. You can retrieve CAMEO profiles as an address enrichment for many countries.

To receive CAMEO information in the validated addresses, you must install the CAMEO reference address databases.

Note: Informatica Address Verification adds CAMEO information only to validated addresses. To enable CAMEO enrichment, you must install the Batch and Interactive reference address databases along with the CAMEO databases.

The CAMEO profiles contain socio-demographic and lifestyle data at the microcell level based on parameters such as age, education, income, and general interests.

The following list shows some of the use cases for CAMEO profiles:

- Enhance and segment consumer databases
- Improve your understanding about customers and responders
- Find more prospects by finding look-alikes
- Perform area and location analysis
- Understand market potential
- Perform advanced statistical analysis and modeling

Database Filename Format

The CAMEO database filenames have the following format:

[ISO]_ADV_ENR_CAM_000_6_1_0.MD6

Note: [ISO] represents the three-letter ISO-3166 code for the country name.

Supported Countries

You can configure Address Verification to include CAMEO information in valid output for the following countries:

Australia	Austria
Belgium	Brazil
Canada	Czech Republic
Denmark	Estonia
Finland	France
Germany	Hong Kong
Hungary	India
Indonesia	Ireland
Italy	Japan
Korea	Mexico
Netherlands	New Zealand
Norway	Poland
Portugal	Romania
Singapore	Slovakia
South Africa	Spain
Sweden	Switzerland
United Kingdom	United States

Status Codes for CAMEO

Address Verification returns status codes that describe the status of the CAMEO data in the output address. For information about CAMEO status codes, see "Certification and Enrichment Data Status Codes" on page 102.

Enabling CAMEO Demographic Profiles

You can configure Informatica Address Verification to include CAMEO demographic profile information as an enrichment to addresses.

- Include the CAMEO element in your input address request and set the element to *true*. Find the CAMEO element under **Parameters** > **CountrySet** > **Enrichments** .

  The default value of the element is *false*.

CAMEO Output Fields

When you validate an address with CAMEO enrichment enabled, you receive the following information in the validated output:

CATEGORY

A code for the age and affluence of the address residents at the country level.

CATEGORY_DESCRIPTION

A description for the age and affluence of the address residents at the country level.

GROUP

A code for the neighborhood of the address at the country level.

GROUP_DESCRIPTION

A description for the neighborhood of the address at the country level.

INTERNATIONAL

A code for the age and affluence of the address residents at an international level.

INTERNATIONAL_DESCRIPTION

A description for the age and affluence of the address residents at an international level.

MVID

A match key that you can use to link your CAMEO-encoded addresses to CAMEO Analysis, a TransUnion Information Group product. CAMEO Analysis is a separate product that you can license directly from TransUnion Information Group.

Find the CAMEO output properties under **IO** > **Outputs** > **Results** > **Enrichments**.

Geocodes

You can use the Geocoding element to add geocoordinates as an enrichment to a verified address. Geocoordinates indicate the latitude and the longitude of an address. Informatica Address Verification follows the World Geodetic System standards (WGS 84) for geocoding.

Address Verification can return geocoordinates at different levels of precision, based on the properties that you configure and on geocoding reference data files that you install.Find the Geocoding element under **Process** > **Enrichments** in the `AVJob.schema.json` file.

The Geocoding element includes the following properties:

- Rooftop
- ArrivalPoint
- StreetCenter
- LocalityCenter
- PostalCodeCenter
- Preferred

You can set one or more of the Rooftop, ArrivalPoint, StreetCenter, LocalityCenter, and PostalCodeCenter properties to true or false to specify the type of geocoordinates that you require. Address Verification will return geocoordinates for each type that you set to true.

Alternatively, you can use the Preferred property to list multiple types of geocoordinates in an array. When you use the Preferred property, Address Verification returns the geocoordinates for the first type on your list that it finds in the reference data.

If Address Verification cannot find any type of geocoordinates that you specify in the reference data, Address Verification does not return any geocoordinates. The default value on the properties is false.

You can set the Preferred property to the following values:

- Rooftop
- ArrivalPoint
- StreetCenter
- LocalityCenter
- PostalCodeCenter

Note: A true value on a Rooftop, ArrivalPoint, StreetCenter, LocalityCenter, or PostalCodeCenter property takes precedence over any value that you set on the Preferred property.

Types of Geocoordinates

Address Verification can return the following types of geocoordinates in the verified output:

Rooftop

Rooftop geocoordinates map to the center of the roof of the primary building on a parcel of land.

ArrivalPoint

ArrivalPoint geocoordinates map to a point in the center of the street segment in front of a house or a building.

StreetCenter

StreetCenter geocoordinates map to an approximate mid-point of the street, based on known geocodes for locations nearby.

LocalityCenter

LocalityCenter geocoordinates map to an approximate mid-point of the locality, based on known geocodes for locations nearby.

By default, Address Verification returns the geocoordinates to the PrimaryLocalityCenter level of precision. Address Verification returns the geocoordinates to a SubLocalityCenter level of precision if the sub-locality data is present in the reference data.

PostalCodeCenter

PostalCodeCenter geocoordinates map to a post office that handles mail for the address.

Each geocoding property can return information on the following properties:

- Value. Contains the latitude and longitude coordinates of an address.
- Latitude. The latitude value that address verification can return for the input address.
- Longitude. The longitude value that address verification can return for the input address.
- Accuracy. The level of accuracy to which address verification returns geocodes.

GeoCoding Status Values

Informatica Address Verification returns the following geocoding status values to help you understand the status of a geocoding request.

ER_DATA_NOT_AVAILABLE

Verification cannot find the geocoding database.

ER_NOT_UNLOCKED

The geocoding database is not unlocked.

ER_DATA_CORRUPT

The geocoding database is corrupted.

NOTHING_FOUND

Verification cannot append geocoordinates to the input address because no geocoordinates are available for the address.

POCO_BASE_CENTER

Geocoordinates are only partially accurate to the postal code level. For example, 795xx.

POCO_CENTER

Geocoordinates are accurate to the postal code level.

LOCALITY_CENTER

Geocoordinates are accurate to the locality level.

STREET_CENTER

Geocoordinates are accurate to the street level.

INTERPOLATED

Geocoordinates are accurate to the house number level. (Estimated location of the parcel of land with street-side offset.)

POINT_ARRIVAL_POINT

Geocoordinates are accurate to arrival point level. (Measured entryway to the parcel of land.)

POINT_ROOFTOP

Geocoordinates are accurate to rooftop level.

Country-Specific and Global Enrichments

Informatica Address Verification can provide additional information as enrichments to addresses from several countries. Some enrichments are specific to addresses in individual countries, and some enrichments apply equally to every country.

Informatica Address Verification returns the requested enrichments based on the supplementary reference data files that you install and the valid license keys available for the data files.

Enrichments by Country

Use the CountrySpecific property to specify whether Address Verification returns the enrichments that it offers for individual countries. When you set the CountrySpecific property to true, Address Verification returns any country-specific enrichment that it finds in the reference data for the input addresses that identify the country. Find the CountrySpecific property in the AVJob.Schema.json file under **Process** > **Enrichments**. The default CountrySpecific property value is false.

You can additionally configure Address Verification to return enrichments for a subset of the countries for which enrichments are available. To do so, set the CountrySpecific property to true, and add a list of one or more countries as an array to the Countries element. Add the ISO three-character codes for each country in a comma-separated list, for example AUT, DEU. When you add a list of countries to the Countries element, Address Verification returns the available enrichments for the countries that you specify only. Find the Countries element under **Parameters** > **CountrySets**.

The following table lists the countries that support supplementary information, the databases to install, and the enrichment codes for each country:

Country	Database	Enrichment Codes
Australia	AUS_ADV_ENR_EN1_000_6_1_0.MD6	GNAFID | PrimaryGNAFID | PrimarySecondaryIndicator | SA1MainNo | SA1Digit7No | SA2MainNo | SA2Digit5No | SA2Name | SA3Name | SA3Code | SA4Name | SA4Code | GCCName | GCCCode | STEName | STECode | CCD06 | MeshBlock11 | MeshBlock16
Austria	AUT_ADV_ENR_EN1_000_6_1_0.MD6	PostalAddressCode | PostalAddressCodeID
Belgium	BEL_ADV_ENR_EN1_000_6_1_0.MD6	NISCode
Brazil	BRA_ADV_ENR_EN1_000_6_1_0.MD6	IBGECode
Czech Republic	CZE_ADV_ENR_EN1_000_6_1_0.MD6	RUIANSOID | RUIANAMID | RUIANTEAID
France	FRA_ADV_ENR_EN1_000_6_1_0.MD6	INSEECode | INSEE9Code
Germany	DEU_ADV_ENR_EN1_000_6_1_0.MD6	LocalityID | StreetID | AGS | StreetCode
Italy	ITA_ADV_ENR_EN1_000_6_1_0.MD6	ISTATCode

Country	Database	Enrichment Codes
Japan	JPN_ADV_ENR_EN1_000_6_1_0.MD6	ChoumeiAzaCode | NewChoumeiAzaCode | CurrentChoumeiAzaCode | GaikuCode
Poland	POL_ADV_ENR_EN1_000_6_1_0.MD6	GMINACode | LocalityTerytID | StreetTerytID
Russia	RUS_ADV_ENR_EN1_000_6_1_0.MD6	FIASID
Serbia	SRB_ADV_ENR_EN1_000_6_1_0.MD6	PostalAddressCode
South Africa	ZAF_ADV_ENR_EN1_000_6_1_0.MD6	NADID
Spain	ESP_ADV_ENR_EN1_000_6_1_0.MD6	INEProvinceCode | INEMunicipalityCode | INEStreetCode
Switzerland	CHE_ADV_ENR_EN1_000_6_1_0.MD6	PostalCodeExtension
United Kingdom	GBR_ADV_ENR_EN1_000_6_1_0.MD6	DeliveryPointSuffix | DeliveryPointType | UDPRN | AddressKey | OrganizationKey | UPRN
United States	USA_ADV_ENR_EN1_000_6_1_0.MD6	CountyFipsCode | StateFipsCode | MSAID | CBSAID | CensusTractNo | CensusBlockNo | CensusBlockGroup | CensusCountyFips | CensusCountyName | MCDID | PlaceFipsCode | FinanceNumber

You can find the list of country-specific enrichment elements in the AVJob.Schema.json file under **Results** > **Enrichments** > **CountrySpecific**.

Global Enrichments

Use the Global property to activate enrichments that can apply in the same way to addresses in every country. Find the Global property in the AVJob.Schema.json file under **Process** > **Enrichments**. To return the enrichments, set the property to true. The default value is false.

Note: The presence of an enrichment as a global property does not mean that Address Verification will return the enrichment for every country. Address Verification returns a global enrichment in the same manner for addresses in every country that supports the enrichment.

Address Verification supports the following global enrichments:

TimeZoneCode

A one- to three-character numeric value that indicates the offset between the address time zone and Greenwich Mean Time. For example, the time zone code for Eastern Standard Time is -5.

TimeZoneName

A three-character code that identifies the time zone to which the address belongs. For example, EST identifies Eastern Standard Time.

The global enrichments TimeZoneCode and TimeZoneName are available only for United States addresses.

Status Codes for Enrichments

Address Verification returns status codes that describe the status of the enrichment data in the output address. For information about enrichment status codes, see "Certification and Enrichment Data Status Codes" on page 102.

Enrichments for Australia Addresses

You can configure Informatica Address Verification to include address enrichments in the validated Australia addresses. Address Verification bases the enrichments on geographical areas that the Australian Bureau of Statistics defines.

To receive address enrichments for Australia, install the latest supplementary reference data files for Australia and ensure that the valid license keys are available for the data files.

Address Verification provides enrichment data for the following statistical elements:

CCD06

A seven-digit code that represents a census collection district that the Australia Bureau of Statistics defined for the 2006 census. A census collection district is an area that a census data collector might cover in a ten-day period. In urban areas, the district might comprise 220 homes. In rural areas, the district might comprise fewer homes and the geographical area might increase.

GCCCode

A five-character alphanumeric code that identifies the greater capital city (GCCSA) statistical area to which an address belongs. A greater capital city statistical area comprises multiple level-4 statistical areas.

There are 16 spatial GCCSA regions, including eight regions that represent the Australian state and territory capital cities and eight regions that cover the rest of each state and the Northern Territory. In addition, there are 18 non-spatial greater capital city statistical areas.

The five character code includes a single-digit state and territory identifier and a four-character GCCSA identifier.

GCCName

The name of the greater capital city statistical area to which an address belongs.

Greater capital city statistical areas are named for the cities they represent or, if they do not identify a city, for the rest of the state or territory that they represent.

GNAFID

A 14-digit code that identifies an address in the Geocoded National Address File (GNAF). Australian government departments and organizations recognize the code as a persistent, unique identifier for the address.

The GNAFID value is the most precise identifier in the address file for a given address. If the GNAFID value represents a sub-building address, Address Verification returns the GNAF value for the house or building that contains the sub-building in the PrimaryGNAFID field.

Local Government Area Name (LGAName)

The name of a region that is governed by local bodies.

An LGA is identified by unique five-digit codes and an LGAName is abbreviated for standardization across states and territories.

Local Government Area Code (LGACode)

A five-digit numeric code that identifies the Local Government Area (LGA) of an address.

An LGACode consists of a one-digit state or territory identifier and a four-digit LGA identifier. Note that these codes are unique only within a state or a territory. If you want to identify a unique LGACode, you must enter State or Territory code before the four-digit code.

MeshBlock11

An 11-digit code that identifies the mesh block to which an address belongs. MeshBlock11 represents the areas that the Australian Bureau of Statistics defined for the 2011 census. A mesh block represents the smallest geographical area that the Australian Bureau of Statistics uses to generate statistical information.

Mesh blocks are building blocks for statistical information rather than areas for which the Australian Bureau of Statistics releases information. The Australian Bureau of Statistics builds statistical areas and regions from mesh blocks. Mesh blocks broadly align with land use, such as residential use, commercial use, or parkland. A mesh block might contain thirty to sixty dwellings, although some mesh blocks are designed to contain no dwellings.

MeshBlock16

An 11-digit code that identifies the mesh block to which an address belongs. MeshBlock16 represents the areas that the Australian Bureau of Statistics defined for the 2016 census.

Note: The Australian Bureau of Statistics might maintain statistical information for some mesh blocks that it defined for the 2016 census.

See also *MeshBlock11*.

PrimaryGNAFID

A 14-digit code that represents the house or building in the GNAF that the address identifies. Australian government departments and organizations recognize the code as a persistent, unique identifier for the primary address. If the address identifies a sub-building in the house or building, the GNAFID field returns the code for the address at sub-building level.

PrimarySecondaryIndicator

Indicates whether an input address is a primary address or a secondary address. A primary address in the GNAF is a house or building. A secondary address is a sub-building within the house or building. The PrimarySecondaryIndicator value can be P for primary, or S for secondary, or the field can be empty when the input address is not a multi-resident dwelling.

SA1Digit7No

A seven-digit code that identifies the level-1 statistical area that an address belongs to but that does not provide the full hierarchy of geographical information that SA1MainNo provides. The seven-digit code comprises the state and territory identifier, SA2 identifier, and SA1 identifier.

A level-1 statistical area is typically the smallest area for which the Australian Bureau of Statistics releases statistical data.

SA1MainNo

An 11-digit code that identifies the level-1 statistical area to which an address belongs. A level-1 statistical area is typically the smallest area for which the Australian Bureau of Statistics releases statistical data.

The 11 digits of the SA1MainNo code include the values of the larger areas in the statistical area hierarchy. That is, the SA1MainNo includes a state and territory identifier, SA4 identifier, SA3 identifier, and SA2 identifier in addition to the SA1 identifier. The final two digits in the code identify the level-1 statistical area. In this way, the SA1MainNo uniquely identifies the area to which the address belongs.

SA2Digit5No

A five-digit code that identifies the level-2 statistical area that an address belongs to but that does not provide the full hierarchy of geographical information that SA2MainNo provides. The five-digit code comprises the state and territory identifier and the four-digit SA2 identifier.

A level-2 statistical area is a physical area that broadly defines a single social or economic community. A level-2 statistical area comprises multiple level-1 statistical areas.

SA2MainNo

A nine-digit code that identifies the level-2 statistical area to which an address belongs. A level-2 statistical area is a physical area that broadly defines a single social or economic community. A level-2 statistical area comprises multiple level-1 statistical areas.

The nine digits of the SA2MainNo code include the values of the larger areas in the statistical area hierarchy. That is, the SA2MainNo includes a state and territory identifier, SA4 identifier, and SA3 identifier in addition to the SA2 identifier. The final four digits in the code identify the level-2 statistical area.

SA2Name

The name of the level-2 statistical area to which the address belongs. Each level-2 name is unique, and each name contains no more than forty characters. In an urban area, the name is based on the suburb or suburbs that the area covers. In rural areas, the name is based on the locality that the area covers.

SA3Code

A five-digit code that identifies the level-3 statistical area to which an address belongs. A level-3 statistical area comprises multiple level-2 statistical areas.

The five digits of the level-3 statistical areas include the values of larger geographical regions. That is, the SA3Code includes a state and territory identifier and SA4 code in addition to the SA3 identifier. The final two digits in the code identify the level-3 statistical area.

A level-3 statistical area generally covers a population of between 30,000 and 130,000 people.

SA3Name

The name of the level-3 statistical area to which the address belongs. Each level-3 name is unique, and each name contains no more than forty characters. SA3 names reflect the names of the cities, towns, or rural areas that they cover.

SA4Code

A three-digit code that identifies the level-4 statistical area to which an address belongs. A level-4 statistical area comprises multiple level-3 statistical areas.

The three digits of the level-4 statistical areas include the values of the largest sub-state regions in the main structure of the Australian Statistical Geography Standard (ASGS). That is, the SA4Code includes a state and territory identifier and SA4 identifier. The final two digits in the code identify the level-4 statistical area.

A level-4 statistical area contain at least 100,000 people. In regional areas, a level-4 statistical area might include between 100,000 and 300,000 people. In urban areas, a level-4 statistical area might include between 300,000 and 500,000 people.

SA4Name

The name of the level-4 statistical area to which the address belongs. Each level-4 name is unique, and each name contains no more than forty characters. SA4 names reflect the names of the cities, towns, or rural areas that they cover.

STECode

An unique one-digit code that represents a state or territory.

STEName

Represents the name of a state or territory.

Enrichments for Austria Addresses

You can configure Informatica Address Verification to include unique identification codes in the output for valid Austria addresses.

To receive address enrichments for Austria, install the latest supplementary reference data files for Austria and ensure that the valid license keys are available for the data files.

Address Verification can return the following code values:

PostalAddressCode

The Postal Address Code, or PAC, is a unique identifier for a current Austria address.

PostalAddressCodeID

ThePostalAddressCodeID is the PAC of the address at which a building receives mail if the building has more than one address.

Address Verification reads the address from the `AUT_ADV_VRF_ACL_000_6_1_0.MD6` database.

PostalAddressCode Enrichment

The PostalAddressCode is a unique identifier for the current version of an address in Austria. For example, the following address returns a PostalAddressCode value of 105176447:

```
Plättenstraße 7
2380 Perchtoldsdorf
Niederösterreich
AUT
```

PostalAddressCodeID Enrichment

An Austria address has a PostalAddressCodeID value when the address identifies a mailbox that receives mail at another address. For example, a building at an intersection of two streets might have an address on both streets and might specify one of the addresses as the mailbox address.

Note: The address that gives access to the mailbox is called the Ident address.

A street address that does not receive mail has a PostalAddressCode value and a PostalAddressCodeID value. The PostalAddressCodeID value is the postal address code of the Ident address that receives the mail. The postal carrier delivers mail to the address that the PostalAddressCodeID identifies.

The following table lists street addresses that identify a single mail destination:

Address	PostalAddressCode	PostalAddressCodeID
Hauptplatz 4 8010 Graz AUT	100001915	100004254
Neue-Welt-Gasse 2 8010 Graz AUT	100004254	Not applicable

The address "Hauptplatz 4" does not receive mail because the mailbox is at another address at the same location. The building receives mail at “Neue-Welt-Gasse 2” and therefore “Neue-Welt-Gasse 2” is the Ident address. The PostalAddressCodeID is the PAC of the address that receives the mail. The PostalAddressCode value for the Ident address is 100004254.

Enrichment for Belgium Addresses

You can configure Informatica Address Verification to add the National Institute of Statistics (StatBel) or NISCode as an enrichment to validated Belgium addresses.

To receive address enrichments for Belgium, install the latest supplementary reference data files for Belgium and ensure that the valid license keys are available for the data files.

NIS codes are five-digit codes that uniquely identify geographic areas in Belgium. If you enable enrichment for Belgium, Address Verification returns a nine-digit code that contains the five-digit NISCode and a four-digit Neighborhood ID.

For example, Address Verification returns `21004A001` as an enrichment to the following address:

```
Rue au Beurre 1
1000 Bruxelles
BEL
```

In this example, `21004` is the NISCode and `A001` is the Neighborhood ID.

Enrichments for Brazil Addresses

You can configure Informatica Address Verification to include the Brazilian Institute of Geography and Statistics (IBGE) code as enrichment to validated Brazil addresses.

To receive address enrichments for Brazil, install the latest supplementary reference data files for Brazil and ensure that the valid license keys are available for the data files.

The IBGECode is a seven-digit numeric code that identifies cities and states in Brazil. The IBGECode is useful for e-commerce operations as you can use this code for taxation and audit purposes.

Example: IBGECode in Address Output

When you validate the following address with the address enrichment enabled, Address Verification returns an IBGECode of `IBGECode: 2606101` as an enrichment to the validated output.

```
Rua da Matriz 9
Centro
Glória do Goitá-pe
55620-000
Brazil
```

Enrichment for Czech Republic Addresses

You can configure Informatica Address Verification to add RUIAN ID codes as an enrichment to a valid Czech Republic address. The Czech Office for Surveying, Mapping and Cadastre (ČÚZK) maintains the RUIAN code data.

To receive address enrichments for Czech Republic, install the latest supplementary reference data files for Czech Republic and ensure that the valid license keys are available for the data files.

The RUIAN ID enrichment comprises the following codes:

- RUIANAMID. Uniquely identifies the address delivery point.
- RUIANSOID. Identifies the address to building level.
- RUIANTEAID. Identifies the building entrance.

The supplementary database for the Czech Republic includes RUIANAMID and RUIANSOID values for ninety-nine percent of Czech Republic addresses. The database includes RUIANTEAID values for a small percentage of addresses.

Enrichments for France Addresses

You can configure Informatica Address Verification to include the INSEECode and the INSEE9Code in the validated output for France addresses.

To receive address enrichments for France, install the latest supplementary reference data files for France and ensure that the valid license keys are available for the data files.

The INSEECode is a numeric indexing code that the French National Institute for Statistics and Economic Studies (INSEE) use to identify entities such as French communes and departments. INSEE codes are particularly helpful in uniquely identifying French communes that share the same name, spelling, and pronunciation. Of a five-digit INSEE code for a commune, the first two digits represent the department and the last three denote the commune. INSEE codes are also used as National Identification Numbers for French citizens.

The INSEE9Code is also known as the IRIS code. IRIS stands for aggregated units for statistical information in French, and represents a demographic group that contains a maximum of 2000 people. France is composed of around 16,100 IRIS units including 650 units in overseas departments.

Enrichments for Germany Addresses

You can configure Informatica Address Verification to include multiple address enrichments in validated Germany addresses.

To receive address enrichments for Germany, install the latest supplementary reference data files for Germany and ensure that the valid license keys are available for the data files.

You can configure Address Verification to include the following enrichments in validated Germany addresses:

AGS

The Amtliche Gemeindeschlüssel (AGS) is a variable-length code that uniquely identifies a locality in Germany.

LocalityID

The Locality ID is a variable length code that uniquely identifies a locality in Germany.

StreetID

The Street ID is a variable length code that uniquely identifies a street address in Germany.

StreetCode

The Street Code is a three-digit code that identifies a street in Germany. Positions 6,7, and 8 of the Frachtleitcode or Freight code form the street code. A street code value of 994 indicates that the address points to a packstation.

Note: You must install the `DEU_ADV_ENR_EN1_001_6_1_0.MD6` database to receive the StreetCode enrichment for Germany addresses.

For example, when you validate the following address:

```
Röntgenstr. 9
67133 Maxdorf
Germany
```

Address Verification returns the following additional information in the validated output:

```
AGS: 07338018
LocalityID: 68015519
StreetID: 100560690
StreetCode: 057
```

Enrichments for Italy Addresses

You can configure Informatica Address Verification to add ISTATCode data as an enrichment to a valid Italy address.

To receive address enrichments for Italy, install the latest supplementary reference data files for Italy and ensure that the valid license keys are available for the data files.

The ISTAT code contains a series of values that provide geographic and demographic information about the address locale, including the province, municipality, and region to which the address belongs. The Italian National Institute of Statistics (ISTAT) maintains the ISTAT codes.

Enrichments for Japan Addresses

You can configure Informatica Address Verification to add enrichments to valid Japan addresses.

To receive address enrichments for Japan, install the latest supplementary reference data files for Japan and ensure that the valid license keys are available for the data files.

Address Verification offers the following enrichments for Japan:

- **ChoumeiAzaCode**
- **NewChoumeiAzaCode**
- **CurrentChoumeiAzaCode**
- **GaikuCode**

Choumei Aza Codes

A Choumei Aza code is an 11-digit string that represents an address to delivery-point level in Japan. When Japan Post updates the address information for a delivery point, Japan Post also issues a new Choumei Aza code for the delivery point. Therefore, a delivery point might have a current address and one or more legacy addresses and a corresponding set of current and legacy Choumei Aza codes.

You can submit a Japan address in batch or interactive mode and return one or more Choumei Aza codes that represent different versions of the address. You can submit a Choumei Aza code in address code lookup mode to obtain the version of the address that the code identifies.

Address Verification returns the following types of Choumei Aza code:

ChoumeiAzaCode

The code that corresponds directly to the validated version of the address that you submit.

NewChoumeiAzaCode

The code that corresponds to the next iteration of the input address that you submit. For example, if you enter the first or oldest version of the address, the NewChoumeiAzaCode value that Address Verification returns represents the second version of the address.

CurrentChoumeiAzaCode

The code that corresponds to the current version of the address that you submit.

Example

The Japan reference data contains the current version of an address and two older versions. You have the first or oldest version of the address. You select the Japan enrichments, and you submit the address in batch mode. Address Verification verifies the address and returns the Choumei Aza code enrichments. You submit the CurrentChoumeiAzaCode value in address code lookup mode. Address Verification returns the current version of the address.

Note: The NewChoumeiAzaCode returns the Choumei Aza code for the update that directly followed the address that you submitted. You can rerun the steps with the NewChoumeiAzaCode value to find each version of the address in the reference data. Or, you can use the ArchiveHandling property to retrieve the current version of any older address in a single step. For more information on ArchiveHandling, see GUID-EDCA4648-4A25-43F6-9684-740AC9BFECA3.

GaikuCode

A Gaiku code is a four-digit code that identifies a city block in Japan.

Append the Gaiku code to a current Choumei Aza code to create a 15-digit string that you can submit in address code lookup mode to find an address.

Enrichments for Poland Addresses

You can configure Informatica Address Verification to include the GMINACode, LocalityTerytID, and StreetTerytID as enrichments for validated Poland addresses.

To receive address enrichments for Poland, install the latest supplementary reference data files for Poland and ensure that the valid license keys are available for the data files.

Official Register of the Territorial Division of the Country (TERYT) is the agency responsible for identifiers and names of territories, localities, roads, and buildings in Poland. Gmina is the Polish equivalent of communes or municipalities. TERYT assigns and manages Gmina code and TerytIDs.

Enrichments for Russia Addesses

You can configure Informatica Address Verification to include the Federal Information Addressing System (FIAS) ID in the validated output for Russia addresses.

To receive address enrichments for Russia, install the latest supplementary reference data files for Russia and ensure that the valid license keys are available for the data files.

The FIASID is an alphanumeric string.

Enrichments for Serbia Addresses

You can configure Informatica Address Verification to include the PostalAddressCode as an enrichment to the validated output for Serbia addresses.

To receive address enrichments for Serbia, install the latest supplementary reference data files for Serbia and ensure that the valid license keys are available for the data files.

The PostalAddressCode is a six-digit code that maps to the street level. Including the PostalAddressCode in an address ensures correct and prompt delivery to recipients in Serbia. You do not need the PostalAddressCode for items that you address to a P.O. Box, *poste restante*, or to a military address.

Enrichments for South Africa Addresses

You can configure Informatica Address Verification to include the National Address Database (NAD) ID in the validated output for South Africa addresses.

To receive address enrichments for South Africa, install the latest supplementary reference data files for South Africa and ensure that the valid license keys are available for the data files.

The NADID is a unique numeric ID that is assigned to street addresses in South Africa.

For example, Address Verification returns the NADID value of `2170232` in the output when you validate the following address:

```
4 Balmoral Road
Vincent
East London
5247
South Africa
```

Enrichments for Spain Addresses

You can configure Informatica Address Verification to add INE code data as an enrichment to a valid Spain address.

To receive address enrichments for Spain, install the latest supplementary reference data files for Spain and ensure that the valid license keys are available for the data files.

The INE code contains a series of values that identify the INEProvinceCode, INEMunicipalityCode, and INEStreetCode to which the address belongs. The National Statistics Institute of Spain (INE) maintains the INE codes.

Enrichments for Switzerland Addresses

You can configure Informatica Address Verification to include the additional postal code characters in the validated output for Switzerland addresses.

To receive address enrichments for Switzerland, install the latest supplementary reference data files for Switzerland and ensure that the valid license keys are available for the data files.

Address Verification returns the additional postal code characters in an enrichment field called PostalCodeExtension.

For example, when you validate the following Switzerland address with the address enrichment enabled, Address Verification returns a PostalCodeExtension value of 02:

```
Müllerboden
Buochserbergstrasse 5
6383 Niederrickenbach
Switzerland
```

Enrichments for the United Kingdom Addresses

You can configure Informatica Address Verification to include address enrichments in validated United Kingdom addresses.

To receive address enrichments for United Kingdom, install the latest supplementary reference data files for United Kingdom and ensure that the valid license keys are available for the data files.

Address Verification includes the following enrichments in validated United Kingdom addresses:

AddressKey

An eight-digit numeric code that maps to an address in the Postcode Address File (PAF) from the Royal Mail. An Address Key in conjunction with Organization Key and the Post Code Type uniquely identifies an address.

DeliveryPointSuffix

A two-character suffix that the Royal Mail assigns to a mailbox in a United Kingdom post code area. The first character in a delivery point suffix is a number and the second character is a letter. A combination of a post code and the delivery point suffix identifies a mailbox.

DeliveryPointType

A single-character code that indicates whether the address points to a residence (R), a small organization (O), or a large organization (L).

OrganizationKey

A unique 8-digit numeric code that Royal Mail assigns to small organizations.

UDPRN

An eight-character code that uniquely identifies each postal address in the Royal Mail PAF database. The Unique Delivery Point Reference Number (UDPRN) remains uniquely tied to the physical delivery point regardless of changes in the address.

UPRN

A numeric code that uniquely identifies a land or property unit in the United Kingdom. The Unique Property Reference Number (UPRN) is a code that the United Kingdom government assigns and can contain a maximum of 12 digits.

Examples

The following examples show enrichment values for different delivery point type addresses.

Delivery Point Type	Input Address	Enrichment Values
Residence	FLAT 17 GROVE HOUSE WAVERLEY GROVE LONDON N3 3PU UNITED KINGDOM	AddressKey: 18161676 DeliveryPointSuffix: 1H DeliveryPointType: R OrganizationKey: 00000000 UDPRN: 15498195 UPRN: 200123099
Large Organization	PO BOX 43078 LONDON NW1 1SF UNITED KINGDOM	AddressKey: 02356470 DeliveryPointSuffix: 1A DeliveryPointType: L OrganizationKey: 00000000 UDPRN: 17635833 UPRN: 10015054387
Small Organization	17A THE GROVE LONDON N3 1QN UNITED KINGDOM	AddressKey: 28470295 DeliveryPointSuffix: 1H DeliveryPointType: O OrganizationKey: 01464593 UDPRN: 15491057 UPRN: 200210632

Enrichments for United States Addresses

You can configure Informatica Address Verification to add multiple address enrichments when you verify United States addresses.

To receive address enrichments for the United States, install the latest supplementary reference data files for United States and ensure that valid license keys are available for the data files.

Address Verification includes the following enrichments for validated United States addresses:

CBSAID

A Core-Based Statistical Area (CBSA) identification number is a five-digit number that identifies an urban area with a population greater than 10,000. A CBSA can be a Metropolitan Statistical Area or Micropolitan Statistical Area. A Metropolitan Statistical Area has over 50,000 inhabitants. A Micropolitan Statistical Area has between 10,000 and 50,000 inhabitants.

CensusBlockGroup

A 12-digit number that identifies a Census Block Group. A Census Block Group is a group of Census Blocks that share the same first digit. The first digit in the Census Block number is the last digit in the 12-digit Census Block Group number.

CensusBlockNo

A four-digit number that identifies a Census Block. A Census Block is the smallest entity for which the Census Bureau collects census information.

CensusCountyFips

A three-digit number that identifies a county in the United States. The United States Census Bureau provides the county information.

Note: The Federal Information Processing Standards (FIPS) include a set of numbers that identify states, counties, and other territorial possessions in the United States. A two-digit state code identifies each state. A three-digit county code identifies a county within a state. Together, the five digits of the state and county codes uniquely identify a county.

CensusCountyName

The name of the county that the Census Bureau provides.

CensusTractNo

A six-digit number that identifies a Census Tract. A Census Tract is a statistical subdivision of a county.

CountyFipsCode

A three-digit number that identifies a county in the United States. The United States Postal Service (USPS) provides the county information.

Note: The United States Census Bureau and the USPS both maintain county FIPS codes. The county codes from each organization differ in approximately 5% of cases.

FinanceNumber

A finance number is a six-digit number that the USPS assigns to post offices and other postal facilities to support the collection of cost and statistical data. The first two digits of the finance number identify the state. The final four digits identify the post office or postal facility.

MCDID

A Minor Civil Division (MCD) identification number is a five-digit number. An MCD is a primary legal subdivision of a county.

MSAID

The Metropolitan Statistical Area (MSA) identification number is a five-digit number that identifies an urban area that has a population of 50,000 or more.

PlaceFipsCode

A five-digit number that identifies a locality in the United States.

StateFipsCode

A two-digit number that identifies a state in the United States.

Certification

Postal certification improves the quality of addresses and ensures that Address Verification services meet postal authority requirements. Certifications indicate if an address contains the data required by the certification standards of national mail carriers.

Address Verification returns the requested certifications based on the certification reference data files that you install and the valid license keys available for the data files.

The following table describes the certification codes that Informatica Address Verification can return:

Certification	Codes
AMAS	ErrorCode | DeliveryPointID | DeliveryID | LotNumber | PostalDeliveryNumber | PostalDeliveryNumberPrefix | PostalDeliveryNumberSuffix | HouseNumber1 | HouseNumber1Suffix | HouseNumber2 | HouseNumber2Suffix
SendRight	AddressType | SOARecordIgnored | DeliveryPointID | DeliveryServiceType | DeliveryServiceNumber | DeliveryServiceLocality | HouseNumber | HouseNumberAlpha | RdNumber | Hygiene | ValidityCode
CASS	ErrorCode | BarCode | DeliveryPoint | PoBoxOnly | RecordType | CarrierRoute | CongressionalDistrict | DeliveryPointCheckDigit | HighriseDefault | HighriseExact | RuralRouteDefault | RuralRouteExact | LACSIndicator | DPVConfirmation | DPVConfirmation3553 | DPVCMRA | DPVPBSA | DPVDNA | DPVNSL | DPVFalsePositive | DPVFootnote1 | DPVFootnote2 | DPVFootnote3 | DPVFootnoteValue | DPVFootnoteComplete | DPVThrowBack | LACSLinkReturnCode | SuiteLinkReturnCode | EWSReturnCode | ZIPMoveReturnCode | DPVNoStatIndicator | DPVNoStatReason | DPVVacantIndicator | DefaultFlag | LACSLinkIndicator | RDI | DPVNDD | ELOTFlag | ELOTSequence
SERP	Category | ExcludedFlag
SNA	Category

Status Codes for Certifications

Address Verification returns status codes that describe the status of the certification data in the output address. For information about certification status codes, see "Certification and Enrichment Data Status Codes" on page 102.

CHAPTER 5

Process Parameter

This chapter includes the following topics:

Process Parameter Overview

You can configure the properties present in the Process element to specify how you want Informatica Address Verification to process an address. You can set the process properties under the Process element in the `AVJob.schema.json` file.

Alternative Handling for Addresses

The AlternativeHandling element specifies whether you want Informatica Address Verification to consider outdated or alias address data while verifying an address. Find the AlternativeHandling element under **Parameters** > **CountrySets** > **Process** in the `AVJob.schema.json` file.

Configure the following elements in the AlternativeHandling element:

- OutdatedAddresses
- AliasAddresses

Note: You can use the GetNewAddress property only when you set the Match property to a value other than Off.

OutdatedAddresses

The OutdatedAddresses element includes the properties Match and GetNewAddress.

You can set the Match property to Off, PerfectOnly, or WithFaultTolerance:

- If you set the Match property to Off, Address Verification does not match an address to the outdated addresses and ignores the outdated addresses available in the reference data. If you submit an outdated address with Match property set to Off, Address Verification rejects or corrects the input address.

- If you set the Match property to PerfectOnly, Address Verification matches only a perfectly entered input address to outdated addresses in the reference data.
- If you set the Match property to WithFaultTolerance, Address Verification matches an input address to outdated addresses in the reference data and also corrects an address to match the outdated addresses.

You can set the GetNewAddress property to true or false. If you set the GetNewAddress property to true, Address Verification returns the current version of the address. If you set the GetNewAddress property to false, Address Verification returns the outdated address.

AliasAddresses

The AliasAddresses element includes the Match property. You can set the Match property to Off or PerfectOnly.

If you set the Match property to Off, Address Verification does not match an address to the alias addresses and ignores the alias addresses available in the reference data. If you submit an alias address with Match property set to Off, Address Verification rejects or corrects the input address.

If you set the Match property to PerfectOnly, Address Verification matches only a perfectly entered input address to alias addresses in the reference data.

For information about how to handle alias addresses, see "AliasHandling" on page 89.

Configuring the AlternativeHandling

To specify whether Informatica Address Verification should match the input addresses with outdated and alias addresses in the reference data, configure the OutdatedAddresses and AliasAddresses properties. Find the properties under **CountrySets** > **Process** > **AlternativeHandling** in the `AVJob.schema.json` file.

- In the OutdatedAddresses element, set the Match property to one of the following values:
  - Off. Address Verification does not match the input address to the outdated addresses in the reference data.
  - PerfectOnly. Address Verification matches only a perfectly entered input address to the outdated addresses in the reference data.
  - WithFaultTolerance. Address Verification matches the input address to the outdated addresses in the reference data and also corrects an address to match the outdated addresses.
- In the AliasAddresses element, set the Match property to one of the following values:
  - Off. Address Verification does not match an input address to the alias addresses in the reference data.
  - PerfectOnly. Address Verification matches only a perfectly entered input address to alias addresses in the reference data.

Reverse Geocoding

You can submit geocoordinates as input values to Informatica Address Verification and retrieve the closest address or addresses to the coordinates. To perform reverse geocoding, submit the address job in GeocodeToAddress mode.

Address Verification applies a search radius to the geocoordinates to define an area in which to search for addresses. The geocoordinates are the center of the search area. The default search radius is 50 meters. For best results, set the radius to a value below 1000 meters.

You can enter a pair of geocoordinates as a single input value, or you can enter the latitude and longitude coordinates for a location as discrete input values.

Address Verification also returns the distance from the geocoordinates of each output address and the direction in which the address lies.

If Address Verification cannot find any address within the search radius, Address Verification returns address with I process status.

To retrieve addresses in GeocodeToAddress mode, configure the following properties:

- SearchRadius. Defines the radius in meters in which Address Verification searches for addresses. Find the property under **Parameters** > **CountrySets** > **Process** > **Geocode** in the `AVJob.schema.json` file.
- Value. Contains the latitude and longitude values that constitute the geocoordinates. The latitude and the longitude values must be separated by a comma. Find the property under **Parameters** > **IO** > **Inputs** > **Geocode** in the `AVJob.schema.json` file. Use the property if you do not use the Latitude and Longitude properties.
- Latitude and Longitude. Contain the latitude and longitude coordinates as discrete values. Find the properties under **Parameters** > **IO** > **Inputs** > **Geocode** in the `AVJob.schema.json` file. Use the properties if you do not use the Value property.

Address Verification returns the following values in addition to each address:

- Distance. Indicates the distance in meters from the address to the center of the search radius. Find the property under **Parameters** > **IO** > **Outputs** > **Results** > **Variants** > **AddressPositionToInputGeocode** in the `AVJob.schema.json` file.
- Direction. Indicates the direction in which the address lies from the center of the search radius. The input geocoordinates define the center of the search radius. Address Verification adds the direction as an absolute bearing in the range 0 through 359. For example, 0 indicates that an address is due north of the center of the radius, and 270 indicates that an address is due west of the center of the radius. Find the property under **Parameters** > **IO** > **Outputs** > **Results** > **Variants** > **AddressPositionToInputGeocode** in the `AVJob.schema.json` file.

Consider the following rules and guidelines when you configure an address job for reverse geocoding:

- Address Verification returns addresses in a list based on their distance from the center of the search radius, starting with the closest address.
- Address Verification follows the World Geodetic System standards (WGS 84) for geocoding.
- Reverse geocoding requires GeocodeToAddress database. If you want to see the geocoordinates of an address, download the ArrivalPoint Geocoding database and enable Arrival Point Enrichment.
- The geocoordinates that you provide may identify a location on a given plot of land but may be closer to the arrival point on an adjacent plot of land. In this case, Address Verification selects the address of the adjacent plot as it represents the nearest arrival point.
- The MaxResultCount property governs the total number of addresses that reverse geocoding can return for the geocoordinates that you enter.
- GeocodeToAddress mode uses arrival point geocodes.

To read the list of countries that support reverse geocoding, see Appendix B, “Reverse Geocoding Coverage” on page 126.

Address Verification Level

You can set the minimum level of information that an address must contain before Informatica Address Verification accepts it as valid. Use the VerificationLevel element to set the level. The VerificationLevel element specifies a threshold of valid information at which Address Verification will not reject an address. Find the VerificationLevel element under **Parameters** > **CountrySets** > **Process** in the `AVJob.schema.json` file.

You can set the following properties on VerificationLevel:

- PreferredVerificationLevel. The desired level.
- MinimumVerificationLevel. The minimum level that you accept for an address to be valid.

You can set the PreferredVerificationLevel and MinimumVerificationLevel properties to one of the following values:

- All. Address Verification verifies all address elements and also checks for the presence of all postally relevant elements. Default is All.
- Address. Address Verification verifies all address elements except company information.
- Premise. Address Verification verifies an address to house number level or building level and ignores sub-building information.
- Street. Address Verification verifies an address to street level or delivery service with number level and ignores information such as house number, building, and sub-building.
- Locality. Address Verification verifies an address to locality level and ignores street and delivery service information. Address Verification also verifies the postal codes.
- PrimaryLocality. Address Verification verifies an address up to locality level 1 and ignores the more granular locality level information. Address Verification also verifies the postal codes.

Ensure that the value that you set in the PreferredVerificationLevel property is not lower than the MinimumVerificationLevel value.

If Address Verification cannot verify the address information at the preferred verification level, it attempts to verify the address at the next level between the preferred and the minimum levels. Address Verification can return the verification level met by the address in the UsedVerificationLevel property.

Address Verification copies any input information outside the minimum verification level to the output address. If Address Verification cannot verify the address information to the minimum level, it rejects the address.

Note: When you set both VerificationLevel element properties to All and Address Verification cannot verify an input data value, Address Verification rejects the address.

CHAPTER 6

Result Parameter

This chapter includes the following topics:

Result Parameter Overview

You can configure the properties in the Result element to specify the formatting of the validated output. Find the Result element in the `AVJob.schema.json` file.

Maximum Number of Results in an Address Job

The MaxResultCount property specifies the maximum number of address suggestions that an address job can return in interactive, quick capture, and GeocodeToAddress modes.

Find the property under **Parameters** > **CountrySets** > **Result** in the `AVJob.schema.json` file.

The MaxResultCount property works in conjunction with the DefaultMaxNumResults and MaxNumResults properties in the `IDVEConfig.json` file that the engine reads.

The DefaultMaxNumResults property defines a standard maximum value for jobs that return address suggestions. If you do not set a value on the MaxResultCount property in a job request, the job reads the value of the DefaultMaxNumResults property.

The MaxNumResults property defines an upper limit for the number of address suggestions that the engine can process. If you do not set a value on the DefaultMaxNumResults property in the `IDVEConfig.json` file, the engine uses the value on the MaxNumResults property in the same file to determine the maximum possible number of address suggestions. The individual values on a MaxResultCount property in a job request and on the DefaultMaxNumResults property in the `IDVEConfig.json` file must not exceed the MaxNumResults property value. The engine reads the MaxNumResults property during initialization. To change the MaxNumResults property value, first deinitialize the engine.

The default value on the MaxNumResults property is 20. The maximum value that you can set on a MaxResultCount, DefaultMaxNumResults, or MaxNumResults property is 100.

NumericRangeExpansion

The NumericRangeExpansion element specifies how Informatica Address Verification returns address suggestions when the input address data matches a range of addresses in the same building or street. For example, Address Verification might determine that an input address matches a range of house numbers on a given street. In such a case, Address Verification can return the upper and lower numbers in the available range in a single suggestion.

Find the NumericRangeExpansion element under **Parameters** > **CountrySets** > **Result** in the `AVJob.schema.json` file.

Address Verification can apply numeric range expansion to house number data, building data, and sub-building data. Address Verification can perform range expansion in QuickCapture mode and Interactive mode.

NumericRangeExpansion Properties

To define how Address Verification handles number ranges, configure the following properties on the NumericRangeExpansion element:

RangesToExpand

Specifies whether to expand number ranges in address suggestions from countries for which individual house number, building, or sub-building values are available in the reference data.

RangeExpansionType

Specifies the scope of range expansion in suggestion lists.

RangesToExpand

The RangesToExpand property has the following options:

All

Expands the house number, building, or sub-building values to match the individual house numbers available in the reference data.

OnlyValid

Expands the ranges only when the reference data contains the complete list of valid numbers in a range.

None

Does not expand any range. The default value is None.

When the complete list of house numbers is not available in the reference address database, OnlyValid works the same way as None.

RangeExpansionType

The RangeExpansionType property has the following options:

Full

Expands all the items in the suggestion list, depending on the current RangesToExpand options.

Flexible

Expands the initial items in the suggestion list, depending on the current RangesToExpand options.

If you set the RangeExpansionType property to Flexible, the internal engine logic decides on the number of results to expand and how many to keep as ranges without exceeding the maximum result count value. Therefore, the suggestion list might contain both expanded and unexpanded ranges if you set the RangeExpansionType property to Flexible.

Configuring the NumericRangeExpansion Properties

To determine how Informatica Address Verification handles number ranges in addresses, configure the RangesToExpand and RangeExpansionType properties. Find the properties under **Result** > **NumericRangeExpansion** in the `AVJob.schema.json` file.

- To specify whether or not to expand the ranges, set the RangesToExpand property to one of the following values:
  - All. Address Verification always expands the ranges.
  - OnlyValid. Address Verification expands the ranges only when the reference address database contains full list of numbers for the specified range.
  - None. Address Verification does not expand a range.
- To specify whether or not to expand all the items in the suggestion list, set the RangeExpansionType property to one of the following values:
  - Full. Address Verification might expand all the items in the suggestion list.
  - Flexible. Address Verification might expand only the initial items in the suggestion list.

Numerical Range Formats

Address Verification uses two periods (..) to indicate a numerical range in QuickCapture mode. Address Verification uses a dash symbol (-) to indicate a numerical range in interactive mode.

For example, 15500..15508 can indicate that houses from 15500 through 15508 on a given street are candidate matches for an input address in QuickCapture mode.

When the Address Verification engine returns multiple ranges for an input address in a single suggestion, each range is separated by a semicolon. For example:

23..27;30..36

The engine can return multiple ranges in a single suggestion.

If the engine reaches the MaxResultCount limit for a given address and additional ranges remain in the reference data, the engine adds three periods (...) after the values in the final suggestion. This indicates that additional address suggestions are available in the reference data.

Note: Generally, Address Verification returns even and odd numbers in a single range. However, Address Verification might return even-numbered and odd-numbered ranges in separate records for a single input address. For example, 1..9 MAIN STREET and 2..10 MAIN STREET may indicate odd-numbered and even-numbered houses respectively.

Rules and Guidelines for Numerical Ranges

Consider the following rules and guidelines for expanding numerical ranges:

- Address Verification ignores the RangeExpansionType property when the RangesToExpand property value is None.
- In jobs that run in QuickCapture mode, Address Verification uses All as the RangesToExpand property value and uses Flexible as the RangeExpansionType property value. Address Verification ignores the physical properties of the NumericRangeExpansion element in QuickCapture mode.

- Each address suggestion that the engine returns can identify a single address, a range of addresses, or multiple ranges. The combination of single addresses and ranges in the suggestion list depends on the RangesToExpand and RangeExpansionType properties and on the MaxResultCount property value.
  The engine may return single addresses in the initial address suggestions and return addresses in ranges in subsequent suggestions.

DeliverAdditionalSuggestions

Address Verification can return additional data for addresses with a V or C score when the data is present in address in the reference data and not used in the output. You can use the DeliverAdditionalSuggestions property to retrieve the data.

Find the property under **Parameters** > **CountrySets** > **Result** in the `AVJob.schema.json` file.

You can configure the property to one of the following values:

No

Address Verification does not return any additional data for V or C addresses. Default is No.

ForAllElements

Address Verification returns additional data for all of the address elements in the database that match the elements in the input address.

ForPostalRelevantElements

Address Verification returns additional data for elements that are postally relevant.

CHAPTER 7

Standardization Parameters

This chapter includes the following topics:

Standardization Overview

You can configure Informatica Address Verification to apply a standard policy to the presentation of different aspects of an output address.

For example, you can specify the character case for the output elements, and you can set a policy for returning element descriptors in their abbreviated or complete forms. You can specify a preferred language for the address data, and you can specify the character set in which to return the address.

Address-Level Properties and Element-Level Properties

You can set standardization policies at the level of the complete address and at the level of discrete address elements. You can set both property types. Element-level properties take precedence over address-level properties.

The following table describes the standardization properties:

PropertyName	Description	Set at Address Level	Set at Element Level
AliasHandling	Specifies how to handle alias names.	Yes	Yes
Casing	Specifies the character case.	Yes	Yes
CountryCodeType	Spcifies the ISO country code to use for country data.	Yes	No
CountryNameLanguage	Spcifies the language in which to return the country name.	Yes	No
DescriptorLength	Specifies whether to abbreviate street and directional descriptors.	Yes	Yes
FormatWithCountry	Specifies whether to include the country name in the output.	Yes	No
MaxItemCount	Specifies the maximum number of address lines that an address element can contain.	No	Yes
MaxItemLength	Specifies the maximum number of characters that an output field can contain.	Yes	Yes
PreferredLanguage	Specifies a list of one or more languages in which Address Verification will return the output address data.	Yes	No
PreferredScript	Specifies the character set in which Address Verification will return the output address data.	Yes	No
StandardizeInvalidAddresses	Specifies that Address Verification standardizes elements in an address that returns I as a Process Status score.	Yes	No

Standardization at the Address Level

Use the Standardizations property to define a general standardization policy for the elements in an address. Find the Standardizations property under **Parameters** > **CountrySets** in the `AVJob.schema.json` file.

You can standardize the following address elements:

- Organization
- Contact
- Building
- SubBuilding
- Street
- HouseNumber
- DeliveryService
- PostOffice
- Locality
- PostalCode
- AdministrativeDivision
- Country
- PostalFormattedAddressLines
- PostalDeliveryAddressLines
- PostalRecipientLines
- PostalLocalityLine
- StreetWithNumber
- NumberAndSubBuilding
- SingleAddressLine

Standardization at the Address Element Level

Use the ElementStandardizations property to define a standardization policy for discrete address elements. Find the ElementStandardizations property under **Parameters** > **CountrySets** > **Standardizations** in the `AVJob.schema.json` file.

You can standardize the following address elements:

- Organization
- Contact
- Building
- SubBuilding
- Street
- HouseNumber
- DeliveryService
- PostOffice
- Locality
- PostalCode
- AdministrativeDivision
- Country

- PostalFormattedAddressLines
- PostalDeliveryAddressLines
- PostalRecipientLines
- PostalLocalityLine
- StreetWithNumber
- NumberAndSubBuilding
- SingleAddressLine

PreferredLanguage

Informatica Address Verification can return address data in more than one language in multiple countries. You can specify the preferred output language for the addresses that you submit. Use the PreferredLanguage property to specify a preferred language. Find the property under **Parameters** > **CountrySets** > **Standardizations** in the `AVJob.schema.json` file.

Consider the following rules and guidelines when you configure the PreferredLanguage property:

- You can specify one or more languages on the PreferredLanguage property. Use the three-letter ISO 639 code associated with each language that you specify. Add a list of one or more languages as an array to the PreferredLanguage property. Address Verification tries to return an address in the most-preferred language based on the order that you specify in the array. If the first language that you specify is not present in the reference data, Address Verification tries to return the address in the next language in the array.

  For example, when you submit NLD, FRA, and DEU as the list of preferred languages, Address Verification returns the address in Flemish. If the reference data does not contain the address in Flemish, Address Verification returns the address in French if available.

  The following code fragment shows a sample of one or more languages in the PreferredLanguage property:

  ```
  "PreferredLanguage":
          {
            "NLD",
            "FRA",
            "DEU"
          }
  ```

  Note: Use a comma to separate the languages on each line of the array, as shown in the code fragment.
- By default, the input list of preferred languages is empty. When the input list is empty, Address Verification returns the default language.
- You can configure the property to return an address in the language that the input address uses. The PRESERVE option is useful in cases where the input address set contains address records that are not recorded in the default language. Address Verification can preserve the input language if the reference data contains the address information in the input language. To preserve the input language, add PRESERVE as the first entry in the PreferredLanguage property. You can also add language codes after the PRESERVE option.

  If Address Verification cannot return an address in the input language, Address Verification tries to return the address in the next language that you specified in the list of preferred languages. If the input list is empty, Address Verification returns the address in the default language.

- You can receive an address in more than one language in a single verification job. To return more than one language at once, add an instance of the Standardizations object for each language and set the PreferredLanguage property for each instance.

  For example, to receive Flemish and French languages in the output, add two instances of the Standardizations object and specify NLD and FRA as the preferred language in each instance.

Multilanguage Country Support

Informatica Address Verification can return address elements in multiple languages for addresses in several countries.

The following table lists the countries and languages that Address Verification can return:

Country	Supported Languages
Algeria	English, French
Austria	English, German
Belgium	English, Flemish, French, German
Bulgaria	English, Bulgarian
Canada	English, French
China	Chinese, English
Croatia	Croatian, Italian
Cyprus	English, Greek, Turkish
Egypt	English, Egyptian
Finland	English, Finnish, Swedish
Germany	English, German
Greece	English, Greek
Hong Kong	English, Chinese
Ireland	English, Irish
Israel	English, Hebrew
Kazakhstan	English, Russian
Lithuania	English, Lithuanian
Macau	English, Portuguese, Traditional Chinese
Portugal	English, Portuguese
Russian Federation	English, Russian

Country	Supported Languages
Slovakia	English, Slovak
Spain	English, Spanish
Switzerland	English, French, German, Italian
Taiwan	English, Chinese
Thailand	English, Thai
Timor-Leste	English, Portuguese
Ukraine	English, Ukrainian

PreferredScript

Informatica Address Verification can return addresses from many countries in more than one script. When necessary, Address Verification can transliterate the address into the Latin scripts that you specify and can add the appropriate Latin characters for the address output. Use the PreferredScript property to specify a preferred script. Find the property under **Parameters** > **CountrySets** > **Standardizations** in the `AVJob.schema.json` file.

The PreferredScript property supports the following options:

- Script. Specifies the policy that Address Verification uses to select the script for an output address.
- TransliterationType. Specifies the transliteration standard that Address Verification uses to transliterate the results into Latin when multiple Latin standards are available.
- LimitLatinCharacters. Specifies the depth of the Latin character set that Address Verification can use in an output address.

By default, Address Verification returns the addresses in the primary native script that the country uses for the addresses.

PreferredScript Process Flow

The process includes the following steps:

1. In the Script option, review or update the script policy for the output data.
2. If you select a Latin script, specify a transliteration type to apply to the output.
3. To further define the output, for example to omit special characters that a character set does not support, set the LimitLatinCharacters option.

Script

You can set one of the following values:

NativePrimary

Returns the address in the primary native script that the country uses for the addresses. NativePrimary is the default value.

NativeAlternative1

Returns the address in the alternative script that the country uses.

Latin

Transliterates the address into a Latin script.

Note: You can use the options TransliterationType and LimitLatinCharacters only when you request an output in a Latin script.

Preserve

Returns the address in the same script as the input address.

TransliterationType

You can set one of the following values:

Default

Transliterates the address into a default transliteration type. Default is the default value.

Alternative1

Transliterates the address into an alternative transliteration type if the alternative is available for the address.

LimitLatinCharacters

You can set one of the following values:

NoLimit

Returns the transliterated output in full. NoLimit is the default value.

Latin1

Limits the Latin characters to the Latin-1 (ISO-8859-1) character set. For example, replaces the following characters with Latin-1 characters:

Input	Output (Latin-1)
ä	ä
á	á
à	à
ả	a
ā	a

ASCIISimplified

Changes non-ASCII Latin characters to characters in the ASCII character set. Uses basic conversion, for example ö to o.

ASCIIExpanding

Changes non-ASCII Latin characters to characters in the ASCII character set. Uses extended conversion, for example ö to oe.

Note: Verify that your database supports the characters that the transliteration can return. For example, a database that supports ASCII characters only will not store all Latin characters.

PreferredScript Options

You determine the scripts that Address Verification returns when you set the preferred script options. The Script table displays the different scripts that Address Verification returns for each country. The TransliterationType table displays the transliteration methods used for each country when you request a Latin script in the output.

Script

The following table lists the preferred scripts that Address Verification can specify for different countries:

Country	NativePrimary	NativeAlternative1
BLR	Belarusian Cyrillic	Belarusian Cyrillic
BGR	Bulgarian Cyrillic	Bulgarian Cyrillic
CHN	Hanzi	Hanzi
CZE	Latin-2	Latin-2
FIN	Latin-1	Latin-1
GRC	Greek	Greek
HKG	Hanzi	Hanzi
HUN	Latin-2	Latin-2
ISR	Hebrew	Latin
JPN	Kanji	Kana
KAZ	Cyrillic	Cyrillic
KOR	Hangul	Hangul
LVA	Latin-7	Latin-7
MAC	Hanzi	Hanzi
MDA	Latin-2	Latin-2
MKD	Cyrillic	Cyrillic
POL	Latin-2	Latin-2
ROU	Latin-3	Latin-3
RUS	Cyrillic	Cyrillic
SVK	Latin-2	Latin-2
TWN	Hanzi	Hanzi

Country	NativePrimary	NativeAlternative1
UKR	Cyrillic	Cyrillic
VNM	Vietnamese	Vietnamese

TransliterationType

The following table lists the transliteration methods that Address Verification can use for different countries:

Country	Default	Alternative1
BLR	ASCII, transliterated to the Belarusian database standard	ASCII, transliterated to the Belarusian BGN standard
BGR	ASCII, transliterated to the Bulgarian BGN standard	ASCII, transliterated to the Bulgarian BGN standard
CHN	Latin, Mandarin transliteration	Latin, Cantonese transliteration
CZE	ASCII	Latin-2
FIN	ASCII	Latin-1
GRC	Latin-1, transliterated to the ISO standard	ASCII, transliterated to the BGN standard
HKG	Latin, Cantonese transliteration	Latin, Mandarin transliteration
HUN	ASCII	Latin-2
ISR	Latin	Latin
JPN	Latin	Latin
KAZ	Latin-2, transliterated to the ISO standard	ASCII, transliterated to the BGN standard
KOR	ASCII, transliterated using Revised Romanization	ASCII, transliterated using ISO/TR 11941 Romanization
LVA	ASCII	Latin-7
MAC	Latin, Cantonese transliteration	Latin, Mandarin transliteration
MDA	ASCII	Latin-2
MKD	ASCII	ASCII, transliterated to the Macedonian BGN standard
POL	ASCII	Latin-2
ROU	ASCII	Latin-3

Country	Default	Alternative1
RUS	Latin-2, transliterated to the ISO standard	ASCII, transliterated to the BGN standard
SVK	ASCII	Latin-2
TWN	Latin, Cantonese transliteration	Latin, Mandarin transliteration
UKR	ASCII	ASCII, transliterated to the Ukrainian BGN standard
VNM	ASCII	Vietnamese

Note: Address Verification can also return addresses in ASCII-simplified and ASCII-extended characters for each country.

Casing

The Casing property specifies the character casing policy to apply to the address output.

You can set the property at the element level and at the address level. The element-level Casing property takes precedence over the address-level property.

To set the casing policy at the address level, find the property under **Parameters** > **CountrySets** > **Standardizations** in the `AVJob.schema.json` file. To set the casing policy at the element level, find the property under the associated element name under **Parameters** > **CountrySets** > **Standardizations** > **ElementStandardizations** in the `AVJob.schema.json` file.

You can set the following casing policies at the address level and individually on each element:

Upper

Returns the output in uppercase letters.

Lower

Returns the output in lowercase letters.

Mixed

Returns the output in mixed case with an appropriate use of uppercase letters. For example, a mixed case policy will start proper nouns and street names with an uppercase letter.

Preserve

Preserves the style from the input.

If Address Verification corrects the input data, Address Verification endeavors to return the corrected data output in the same case.

PostalAdmin

Returns the output in the casing style that the local postal authority prefers. The default value is PostalAdmin.

You can set the following casing policies at the element level only:

UseDefault

Returns the element in the casing style that the address-level casing policy specifies. The default value is UseDefault.

You can set the following casing policies at the element level for elements that combine two or more other elements:

ElementSpecific

Returns each element in the string in the casing style that the element configuration specifies.

The following elements under ElementStandardizations support the ElementSpecific option:

- NumberAndSubBuilding
- PostalDeliveryAddressLines
- PostalFormattedAddressLines
- PostalLocalityLine
- PostalRecipientLines
- SingleAddressLine
- StreetWithNumber

DescriptorLength

The DescriptorLength property specifies whether to abbreviate street and directional descriptors when abbreviations are available in the reference data.

You can set the property at the element level and at the address level. The element-level DescriptorLength property takes precedence over the address-level property.

To set the descriptor length policy at the address level, find the property under **Parameters** > **CountrySets** > **Standardizations** in the `AVJob.schema.json` file. To set the descriptor length policy at the element level, find the property under the associated element name under **Parameters** > **CountrySets** > **Standardizations** > **ElementStandardizations** in the `AVJob.schema.json` file.

You can set the property to one of the following values at the address level and individually on each element:

Database

Returns the descriptor preferred by the postal authority or indicated by the common country rules. Database is the default value at the address level.

Long

Returns the expanded form of the element descriptor.

For example, Address Verification returns STREET for the input element ST.

Short

Returns the abbreviated form of the element descriptor.

For example, Address Verification returns AVE for the input element AVENUE.

Preserve

Returns the descriptor in the same form as the input.

If you select Preserve and Address Verification corrects or adds the descriptor, Address Verification returns the descriptor in the default format that reference data specifies.

You can set the following values at the element level only:

UseDefault

Returns the element descriptor in the form that the address-level property value specifies. UseDefault is the default value at the element level.

MaxItemLength

The MaxItemLength property specifies the maximum number of characters that an output field can contain. The default value for the property is 255.

You can set the property at the element level and at the address level. The element-level MaxItemLength property takes precedence over the address-level property.

To set the maximum item length at the address level, find the property under **Parameters** > **CountrySets** > **Standardizations** in the `AVJob.schema.json` file. To set the maximum item length at the element level, find the property under the associated element name under **Parameters** > **CountrySets** > **Standardizations** > **ElementStandardizations** in the `AVJob.schema.json` file.

You can set the MaxItemLength property for every address element under ElementStandardizations.

For more information about the maximum item length value for address elements, see "Table of Values for MaxItemLength and MaxItemCount" on page 88.

MaxItemCount

The MaxItemCount property indicates the maximum number of information levels that an address element can contain. For example, the locality element supports up to six levels of information. A United States state name can occupy the Locality 1 position in an address, and a county name within the state can occupy the Locality 2 position.

You can set the property at the element level. To set the maximum item length for an element, find the property under the associated element name under **Parameters** > **CountrySets** > **Standardizations** > **ElementStandardizations** in the `AVJob.schema.json` file.

For more information about the maximum item count value for address elements, see "Table of Values for MaxItemLength and MaxItemCount" on page 88.

Table of Values for MaxItemLength and MaxItemCount

The following table describes the MaxItemLength and MaxItemCount properties and supported values for different address elements:

Address Element	Value Range Supported for MaxItemLength	Value Range Supported for MaxItemCount	Default Value for MaxItemCount
Organization	20 through 1024	1 through 3	3
Contact	20 through 1024	1 through 3	3
Building	20 through 1024	1 through 6	6
SubBuilding	20 through 1024	1 through 6	6
Street	20 through 1024	1 through 6	6
HouseNumber	5 through 1024	1 through 6	6
DeliveryService	20 through 1024	1 through 3	3
PostOffice	25 through 1024	1 through 3	3
Locality	20 through 1024	1 through 6	6
PostalCode	5 through 1024	1 through 3	3
AdministrativeDivision	2 through 1024	1 through 3	3
Country	2 through 1024	1 through 3	3
PostalFormattedAddressLines	25 through 1024	1 through 19	19
PostalDeliveryAddressLines	25 through 1024	1 through 6	6
PostalRecipientLines	25 through 1024	1 through 6	6
PostalLocalityLine	25 through 1024	1 through 6	6
StreetWithNumber	25 through 1024	1 through 6	6
NumberAndSubBuilding	25 through 1024	1 through 6	6
SingleAddressLine	25 through 1024	Not Applicable	Not Applicable

The default MaxItemLength for an address element is 255.

AliasHandling

An alias is a recognized alternative name for an item of information in an address. For example, a street or a locality might have an official name and also an alias by which it is commonly recognized.

You can set the property at the element level and at the address level. The element-level AliasHandling property takes precedence over the address-level property.

To set the alias handling policy at the address level, find the property under **Parameters** > **CountrySets** > **Standardizations** in the `AVJob.schema.json` file. To set the policy at the element level, find the property under the associated element name under **Parameters** > **CountrySets** > **Standardizations** > **ElementStandardizations** in the `AVJob.schema.json` file.

You can configure the AliasHandling property to one of the following values at the address level and individually on each element:

PreserveAll

Retains the alias for the address element.

PreserveOfficial

Returns the version of the address element that the postal authority prefers. The version may be the official name or the alias, depending on the local usage.

PostalAdmin

Returns the official version of the address element. At the address level, the default value is PostalAdmin.

UseAbbreviation

Returns the address element in its abbreviated form if the reference data contains the abbreviation. The reference data can contain abbreviations for street information, such as street names, descriptors, and directional values, and for locality information.

When you validate addresses in certified mode, set the AliasHandling property to PreserveOfficial.

Note: If you set the AliasHandling property value to UseAbbreviation, the DescriptorLength property value takes precedence. For more information about the DescriptorLength property, see "DescriptorLength" on page 86.

You can set the following values at the element level only:

UseDefault

Returns the version of the address element that the address-level property value specifies. At the element level, the default value is UseDefault.

FormatWithCountry

The property specifies whether or not to include the country name in any postally formatted lines in the output. You can set the FormatWithCountry property to true or false. The default value is false.

The property applies at the address level. Find the property under **Parameters** > **CountrySets** > **Standardizations** in the `AVJob.schema.json` file.

StandardizeInvalidAddresses

The StandardizeInvalidAddresses property can standardize the address elements in an address that returns I as a Process Status score. Standardizing the address elements in an invalid address can improve the performance of any downstream data quality process, such as deduplication, that you apply to the address.

You set the property at the address level. Find the property under **Parameters** > **CountrySets** > **Standardizations** in the `AVJob.schema.json` file.

Set the StandardizeInvalidAddresses property to true or false. The default value is false.

When the StandardizeInvalidAddresses property is set to true, Informatica Address Verification standardizes the following address elements in addresses that have an I process status:

- Street types
- Pre- and post-directionals
- Delivery service items
- Subbuilding descriptors
- State/province/regions

For example, Address Verification can standardize California to CA.

CountryNameLanguage

The CountryNameLanguage property specifies the language in which to return the country name in the output address. You set the property at the address level. Find the property under **Parameters** > **CountrySets** > **Standardizations** in the `AVJob.schema.json` file.

The CountryName value includes the ISO-639 three-letter code for the language. The default value is ENG, which denotes the name of the country in English.

You can set the property to one of the following values:

- `CHI. Chinese.`
- `DAN. Danish.`
- `DEU. German.`
- `ENG. English.`
- `SPA. Spanish.`
- `FIN. Finnish.`
- `FRE. French.`
- `GRE. Greek.`
- `HUN. Hungarian.`
- `ITA. Italian.`
- `JPN. Japanese.`
- `COR. Korean.`
- `DUT. Dutch.`
- `POL. Polish.`
- `POR. Portuguese.`
- `RUA. Russian.`
- `ARA. Arabic.`
- `SWE. Swedish.`

CountryCodeType

The CountryCodeType property specifies how Address Verification returns the country code in the output address. You set the property at the address level. Find the property under **Parameters** > **CountrySets** > **Standardizations** in the `AVJob.schema.json` file.

Address Verification can return the ISO-3166-1 two-character code, ISO-3166-1 three-character code, or ISO-3166-1 three-digit number for the country. The default value is ISO3, which denotes the three-character code.

You can set the property to one of the following values:

- ISO2. ISO 3166-1 two-character code.
- ISO3. ISO 3166-1 three-character code.
- ISONumber. ISO 3166-1 three-digit code.

CHAPTER 8

Output Fields

This chapter includes the following topics:

Output Fields Overview

When you process an address, you receive a result output that contains a number of fields, such as address elements, address line elements, and enrichment values. The result output also contains process-related information and status codes.

The JSON output file defines the output formats and fields. The result output you receive depends on the process parameters and result properties that you configure.

Output Result Information

The output result that you receive depends on the process and result settings that you configure for an address.

The ResultInfo object contains the following properties:

SourceID

Indicates the user address identity.

ProcessStatus

Indicates status of the process call. For more information, see “Process Status” on page 98.

ProcessStatusReason

Indicates why the engine returns a ProcessStatus value of N for an address. For more information, see "Process Status Reason" on page 99.

ProcessModeUsed

Indicates the process mode that you specified.

DeliverAdditionalSuggestions

Indicates the property value that you specified.

ResultCount

Indicates the number of results. The value range is between 0 through 100. The default value is 0.

ResultCountOverflow

Indicates whether there are more results available when you verify addresses in the interactive or quick capture mode. The possible values are true and false. The default value is false.

ResultCountries

Indicates the country information in the result.

The Result Element

The Results object contains multiple properties. The elements of result properties contain various status codes that helps you analyze the results.

Address Elements

The AddressElements property of the Results object contains the following elements:

Building

Contains building information. A Building field contains building-level data, such as the building descriptor, building name, and building number.

The Building element includes the following properties:

- Value. The building information.
- SubItems. The sub-items of the element. The possible values are PreDescriptor, Name, Number, and PostDescriptor.
- Status. The address element status codes.

SubBuilding

Contains sub-building information, such as an apartment or suite number. A SubBuilding field contains sub-building level data, such as the sub-building descriptor, sub-building name, and sub-building number.

The SubBuilding element includes the following properties:

- Value. The sub-building information.
- SubItems. The sub-items of the element. The possible values are PreDescriptor, Name, Number, and PostDescriptor
- Status. The address element status codes.

Street

Contains street information. A Street field contains street-level data, such as the street name, street post-descriptor, street post-directional, street pre-descriptor, and street pre-directional.

The Street element includes the following properties:

- Value. The street information.
- SubItems. The sub-items of the element. The possible values are , PreDescriptor, PreDirectional, Name, Number, PostDirectional, PostDescriptor, and AdditionalInfo.
- Status. The address element status codes.

SupplementaryDeliveryInformation

Contains supplementary delivery information. A SupplementaryDeliveryInformation field adds supplementary data to an address to assist the mail carrier in mail delivery.

The SupplementaryDeliveryInformation element includes the following properties:

- Value. The supplementary delivery information.
- Status. The address element status codes.

HouseNumber

Contains house number information. The house number identifies a building at street level.

The HouseNumber element includes the following properties:

- Value. The house number information.
- SubItems. The sub-items of the element. The possible values are PreDescriptor, Number, Suffix, and PostDescriptor.
- Status. The address element status codes.

DeliveryService

Contains delivery service information. A delivery service is a mail pickup location managed by the post office, such as a P.O. Box.

The DeliveryService element includes the following properties:

- Value. The delivery service information.
- SubItems. The sub-items of the element. The possible values are PreDescriptor, Number, NumberSuffix, PostDescriptor, and AdditionalInfo.
- Status. The address element status codes.

PostOffice

Contains post office information.

The PostOffice element includes the following properties:

- Value. The post office information.
- SubItems. The sub-items of the element. The possible values are PreDescriptor, Name, Number, PostDescriptor, and AdditionalInfo.
- Status. The address element status codes.

Locality

Contains locality information.

The Locality element includes the following properties:

- Value. The locality information.
- SubItems. The sub-items of the element. The possible values are Prefix, Name, Suffix, SortingCode, and AdditionalInfo.
- Status. The address element status codes.

PostalCode

Contains postal code information.

The PostalCode element includes the following properties:

- Value. The postal code information.
- SubItems. The sub-items of the element. The possible values are Base and AddOn.
- Status. The address element status codes.

AdministrativeDivision

Contains province information.

The AdministrativeDivision element includes the following properties:

- Value. The province information.
- Variants. The variants of the element. The possible values are Extended, ISO, and Abbreviation.
- SubItems. The sub-items of the element. The possible values are Prefix, Name, and Suffix.
- Status. The address element status codes.

Residue

Contains residue information. A Residue field contains data duplicate or redundant data.

The Residue element includes the following properties:

- Value. The residue information.
- Type. The type of residue information. The possible values are Necessary, Superfluous, and Unrecognized.

Country

Contains country information.

The Country element includes the following properties:

- Code. The ISO2, ISO3, or the ISONumber of the country.
- Name. The name of the country, such as ENG.

The CountryNameLanguage and CountryCodeType properties present in the Standardizations element controls the country name and the country code.

Note: By default, Address Verification does not return sub-item data or variant data for address elements. Some properties of *CountrySets* in the AVJob.schema.json file determine whether Address Verification returns the data. To return sub-item data or variant data in an output address, set the *SubItems* property under *CountrySets* to *true* in the AVJob.schema.json file.

Contact Elements

The ContactElements property of the Results object contains the following elements:

Name

Contains full name of any contact. The Name element includes the data, such as a first name, middle name, and surname of any contact.

The Name element includes the following properties:

- Value. The contact name information.
- SubItems. The sub items of the element. The possible values are Salutation, Title, FirstName, MiddleName, Surname.

Function

Contains job title of any contact.

DeliveryIdentifier

Contains delivery identifiers data.

Gender

Contains gender of any contact

EMail

Contains email address of any contact.

The EMail element includes the following properties:

- Value. The email address information.
- SubItems. The sub items of the element. The possible values are LocalPart and Domain.

PhoneNumber

Contains phone number of any contact.

The PhoneNumber element includes the following properties:

- Value. The phone number information.
- SubItems. The sub items of the element. The possible values are CountryCode, LocalPrefix, and Number.

Organization Elements

The OrganizationElements property of the Results object contains the following elements:

Companies

Contains company information.

The Companies element includes the following properties:

- Value. The company information.
- SubItems. The sub items of the element. The possible values are Name, PreDescriptor, and PostDescriptor.
- Status. The address element status codes.

Departments

Contains department information.

The Departments element includes the following properties:

- Value. The department information.

- SubItems. The sub items of the element. The possible values are Name, PreDescriptor, and PostDescriptor.
- Status. The address element status codes.

Preformatted Data

The PreformattedData property of the Results object contains the following elements:

SingleAddressLine

Contains address elements in a single line. The value property of SingleAddressLine includes the output address information.

PostalRecipientLines

Contains recipient information, such as contact or organization name. Address Verification supports up to six lines of recipient information. The value property of PostalRecipientLines includes the output address information.

PostalDeliveryAddressLines

Contains delivery information, such as street, house number, building, and delivery service information. Address Verification supports up to six lines of postal delivery address line information. The value property of PostalDeliveryAddressLines includes the output address information.

PostalFormattedAddressLines

Contains unfielded data that includes, recipient information, delivery address information, and locality information. Address Verification supports up to 19 lines of postal formatted address line information. The value property of PostalFormattedAddressLines includes the output address information.

PostalLocalityLine

Contains information, such as locality, postal code, province, and country details. Address Verification supports up to six lines of postal locality line information. The value property of PostalLocalityLine includes the output address information.

StreetWithNumber

Contains street and house number information in a single element. For example, 1 MAIN ST. The value property of StreetWithNumber includes the output address information.

NumberAndSubBuilding

Contains number and sub-building information in a single element. For example, 1 #1. The value property of NumberAndSubBuilding includes the output address information.

Enrichments

The Enrichments property of the Results object contains the following elements:

Geocodes

Contains the closest latitude and longitude coordinates that the reference data can return for each address.

For information about the Geocodes element properties, see “Geocodes” on page 52.

Global

For information about the Global element properties, see “Country-Specific and Global Enrichments” on page 54.

CountrySpecific

For information about the CountrySpecific element properties, see "Country-Specific and Global Enrichments" on page 54.

Certifications

For information about the Certifications element properties, see "Certification" on page 67.

Process Status

The ProcessStatus property provides a summary of the address verification process for a single address. The process status code indicates the overall quality of the output address.

The following table describes the process status values:

Status Code	Description
V	Address Verification verified all postally relevant elements. The input data matches the reference data for the address.
A	Address Verification added one or more relevant address elements to the address.
C	Address Verification corrected one or more input address elements to match the address elements in the reference data.
I	Address Verification cannot correct the address. The address is not valid.
N	Address Verification cannot verify the address. For more information, see "Process Status Reason" on page 99.
F	QuickCapture and GeocodeToAddress modes. Address Verification returned complete addresses in the suggestion list.
P	QuickCapture and GeocodeToAddress modes. Address Verification returned aggregated address suggestions. **Note:** A status of P indicates that the verification process is incomplete and that the available suggestions are not fully extended in the result.

Process Status Reason

The ProcessStatusReason property provides information about the reason why the engine returns a ProcessStatus value of N for an address.

The following table describes the process status reason values:

Status Code	Description
N001	Address Verification cannot verify the address because the reference database is out of date.
N002	Address Verification cannot verify the address because the country data is not licensed.
N003	Address Verification cannot verify the address because the reference database for the input country is corrupted, not well-formatted, or not available.
N004	Address Verification cannot verify the address because the engine either does not recognize or does not support the input country data.
N005	Address Verification cannot verify the address because the geocodes in an input address are incorrect, incomplete, missing, or out of valid range. This error code applies to GeocodeToAddress and QuickCapture modes only.

Address Element Status Codes

The MatchStatus, ResultStatus, PostalRelevance, and ExtendedResultStatus properties provide granular, element-level details.

You can use the status information to assess the following values:

- Quality of the input address and result.
- Probability of successful delivery to the address.
- Postal relevance of address elements.
- Reasons for rejection in case of address that are not valid.
- Level of similarities between the input address and the result.

Match Status

The MatchStatus property describes the level of similarity between each data element in the input address and the corresponding element that the reference data files store for the address. Address Verification returns a discrete match status code for every address data element.

The following table describes the match status values:

Value	Description
0	The input element does not contain any data from the user.
1	The reference data does not contain the input address element for the country as a whole or for specific address, locality, and street.
2	The reference data does not contain a match for the input element value, and the input address does not contain enough verifiable elements to permit an estimate of a correct value for the element.
3	The reference data does not contain a match for the input element value, and Address Verification cannot correct the value. Address Verification rejects the address.
4	The input element value partially matches the reference data.
5	The input element value matches the reference data, but Address Verification corrects the data.
6	The input element value matches the reference data within the available range.
7	The input element value matches the reference data, but Address Verification standardizes the data
8	The input element value perfectly matches the reference data without any errors.

Result Status

The ResultStatus property describes any change made to the input data during processing. Address Verification returns a discrete result status code for every address data element.

The following table describes the result status values:

Value	Description
1	The element value is not changed.
2	The element value is standardized.
3	The element value is replaced by a preferred version because the input contains outdated synonyms, archived names, or a combination of languages.
4	Address Verification changed the secondary part of the elements. For example, Address Verification corrected street descriptors, street directionals, or the +4 data in ZIP Codes.
5	Address Verification changed the primary part of the elements. For example, Address Verification corrected street names or five-digit ZIP Codes.
6	Address Verification added the element to the result output.

Postal Relevance

You can use the PostalRelevance property value to identify the address elements that are relevant to an address according to the local postal guidelines. Address Verification returns a discrete postal relevance value for every address data element.

The following table describes the postal relevance values:

Value	Description
2	Element mandatory. The address element is relevant.
1	Element optional. The address might be valid without the element.
0	Element not used in the postal format. Address element is not relevant.

All address elements with an element relevance value of 2 must be present for the local postal authority to consider an address valid. Element relevance value might vary from address to address for countries with different address types, such as rural address and metropolitan addressing. Furthermore, address elements that have actually been validated against reference data might override the default element relevance value defined for that address element.

Note: Element relevance values are available only for addresses with a process status value of C or V in batch and interactive modes or I values in the interactive mode. Other element-level codes such as match status, result status, and extended result status always return a value regardless of the process status.

Extended Result Status

You can use the ExtendedResultStatus property to see whether more information is available for an address element in the reference data. Address Verification can return an extended result status value for every address data element.

The following table describes the values that the property can return for each element in an input address:

Note: Unless otherwise indicated, Address Verification can return ExtendedResultStatus values in batch, interactive, and certified modes.

Extended Result Status Values	Description
FromDBNotUsed	The reference data contains information for the element that the engine did not use to verify the address. This additional information is not required for a valid address, and the engine did not add the information to the address.
WrongFormatMoved	The input address contains the element at an incorrect position in the formatted address, and the engine moved the element to the correct position. For example, the postal code appears after the locality in a country that requires the postal code to appear before the locality.
AlternativeExists	The reference data includes at least one additional alternative for the address element.
UncheckedParts	One or more values in the element are not verified because they do not appear in the reference data. The engine cannot confirm their relevance and therefore did not remove them from the address.

Extended Result Status Values	Description
CertificationRule	The output element is determined by a postal certification rule. This may contradict the address job properties that you configured.
DominantMatch	The current element matches the reference data, but two or more elements in the input address contradict each other. For example, the postal code does not match the locality.
Deprecated	The address element matches an outdated or archived name.
LanguageNotAvailable	The requested language is not available in the reference data for the element. The engine returns the element in the default language.
RangeInterpolation	The engine returns the input value, as the value falls within a valid range in the reference data. The reference data contains a range of values for the element in the address rather than individual values.
RangeGuess	Applies in interactive mode when the element in the address suggestion contains a range of values. Indicates that the reference data includes the lower and upper values in the range only.
RangeCovered	Applies in interactive mode when the element in the address suggestion contains a range of values. Indicates that the reference data includes values for every address in the range.

Certification and Enrichment Data Status Codes

Informatica Address Verification returns status codes that describe the successful delivery of enrichment and certification data in the output address.

The status codes describe the completeness of the data that address verification returns for global enrichments, country-specific enrichments, CAMEO data, and the supplementary data returned in certification mode.

Address verification returns the following status codes for enrichment and certification data along with the address output:

- AllValuesAvailable. Verification returns all requested enrichment and certification data in the output address.
- SomeValuesAvailable. Verification returns some but not all of the requested enrichment and certification data in the output address.
- NoValuesAvailable. Verification did not return any of the requested enrichment or certification data.

The status code fields are empty when you did not request the enrichment or certification item concerned. The status code fields are also empty when address verification did not find a match for the input address.

Note: Address Verification does not return the empty fields in the JSON output.

CHAPTER 9

Address Status Values and Return Codes

This chapter includes the following topics:

Address Status Values and Return Codes Overview

Informatica Address Verification returns several values that help you to understand and analyze the result output. The status values include a high-level summary of the address verification process.

You also receive return codes that indicate the status of API functions. The API function calls provide information about the status of an API call.

Address Types

You can use the AddressType property information in the result output to identify the type of mail box an address points to.

For United States addresses, Informatica Address Verification returns the address type values that the United States Postal Service specifies. The United States Postal Service includes a RecordType value in the reference data for domestic addresses.

If the reference data does not contain a formal address type designator, Address Verification uses different data elements to assign address types to addresses. Address Verification assigns address type values to addresses from countries that do not define address types based on criteria that Address Verification defined.

Address Verification employs a range of criteria to decide the address type for the following countries:

- Australia
- Canada
- France
- New Zealand

Address Type Indicators for United States Addresses

Informatica Address Verification returns the following address type values for United States addresses:

- F. The address identifies an organization.
- G. The address is a general delivery address. In a general delivery address, the postal code and the recipient data identify the address.
- H. The address identifies a high-rise building. The address contains subbuilding elements such as apartment or suite.
- N. The address cannot be verified. N is the default address type.
- P. The address identifies a Post Office Box or a delivery service.
- R. The address is a rural route/highway contract address.
- S. The address identifies a street.
- U. Unidentified. The address is not valid, and Address Verification does not assign an address type.

Address Type Indicators for Australia Addresses

Informatica Address Verification returns the following address type values for Australia addresses:

- B. The address identifies a building.
- F. The address identifies an organization.
- L. The address post code identifies the organization as a large volume receiver. The reference data adds or validates the organization name.
  Address Verification can determine that the address is a large volume receiver in one of the following ways:
  - The address post code identifies the organization as a large volume receiver.
  - The reference data does not contain street or building information.
- N. The address cannot be verified. N is the default address type. If Informatica Address Verification cannot determine the address type from the address data, it returns the default value.
- P. The address identifies a Post Office Box or a delivery service.
- S. The address identifies a street. S is the default address type. If Address Verification cannot determine the address type from the address data, it returns the default value.
- U. Unidentified. The address is not valid, and Address Verification does not assign an address type.

If an address meets the criteria for more than one address type, Address Verification assigns the first applicable address type from the following list:

L, F, P, B, S

Address Type Indicators for Canada Addresses

Informatica Address Verification returns the following address types for Canada addresses:

- B. The address identifies a building.
- F. The address identifies an organization. In Canada addresses, the type F addresses are a subset of the type L addresses. Therefore, the address type F also indicates a large volume receiver.
- G. The address is a general delivery address. In a general delivery address, the postal code and the recipient data identify the address. Address Verification uses the delivery record in the reference data to identify the address type.
- L. The address post code identifies the organization as a large volume receiver. The address might or might not contain an organization name.
- N. The address cannot be verified. N is the default address type. If Informatica Address Verification cannot determine the address type from the address data, it returns the default value.
- P. The address identifies a Post Office Box or a delivery service.
- R. The address identifies a rural route. Address Verification uses the delivery record in the reference data to identify the address type.
- S. The address identifies a street. S is the default address type. If Address Verification cannot determine the address type from the address data, it returns the default value.
- U. Unidentified. The address is not valid, and Address Verification does not assign an address type.

If an address meets the criteria for more than one address type, Address Verification assigns the first applicable address type from the following list:

F, L, P, B, R, S, G.

Addresss Type Indicators for France Addresses

Informatica Address Verification returns the following address type values for France addresses:

- B. The address identifies a building.
- F. The address identifies an organization. The address does not include a CEDEX post code.
- G. The address is a general delivery address. The reference data does not contain a match for the street information. However, the reference data contains a match for the CEDEX postal code in the address.
- L. The postal code identifies the organization as a large volume receiver. The address might or might not contain an organization name. The reference data uses the CEDEX postal code to add or validate the organization name.
- N. The address cannot be verified. N is the default address type. If Informatica Address Verification cannot determine the address type from the address data, it returns the default value.
- P. The address identifies a Post Office Box or a delivery service.
- S. The address identifies a street address. S is the default address type. If Address Verification cannot determine the address type from the address data, it returns the default value.
- U. Unidentified. The address is not valid, and Address Verification does not assign an address type.

If an address meets the criteria for more than one address type, Address Verification assigns the first applicable address type from the following list:

L, F, P, B, S, G.

Address Type Indicators for New Zealand Addresses

Informatica Address Verification returns the following address type values for New Zealand addresses:

- B. The address identifies a building.
- F. The address identifies an organization.
- L. The address post code identifies the organization as a large volume receiver. The reference data adds or validates the organization name.

  Address Verification can determine that the address is a large volume receiver in one of the following ways:

  - The address post code identifies the organization as a large volume receiver.
  - The reference data does not contain street or building information.
- N. The address cannot be verified. N is the default address type. If Informatica Address Verification cannot determine the address type from the address data, it returns the default value.
- P. The address identifies a Post Office Box or a delivery service.
- R. The address identifies a rural route. Address Verification uses the delivery record in the reference data to identify the address type.
- S. The address identifies a street. S is the default address type. If Address Verification cannot determine the address type from the address data, it returns the default value.
- U. Unidentified. The address is not valid, and Address Verification does not assign an address type.

If an address meets the criteria for more than one address type, Address Verification assigns the first applicable address type from the following list:

L, F, P, B, R, S

Address Type Indicators for Addresses from the Rest of the World

Informatica Address Verification returns the following address type values for addresses from countries other than Australia, Canada, France, New Zealand, and the United States:

- B. The address identifies a building.
- F. The address identifies an organization.
- L. The address post code identifies the organization as a large volume receiver. The reference data adds or validates the organization name.

  Address Verification can determine that the address is a large volume receiver in one of the following ways:

  - The address post code identifies the organization as a large volume receiver.
  - The reference data does not contain street or building information.
- N. Not Certain. The address cannot be verified. N is the default address type. If Address Verification cannot determine the address type from the address data, it returns the default value.
- P. The address identifies a Post Office Box or a delivery service.
- S. The address identifies a street.
- U. Unidentified. The address is not valid, and Address Verification does not assign an address type.

If an address meets the criteria for more than one address type, Address Verification assigns the first applicable address type from the following list:

L, F, P, B, S

Note: Address Verification can return information relevant to the address type on other output elements. Consult the Process Status, Match Status, and Result Status values.

Result Group

When you verify an address in quick capture mode, Informatica Address Verification returns the address suggestions in the following result groups:

- Organization
- Building
- Street
- DeliveryService
- PostalCode
- Locality
- AdministrativeDivision

Address Verification returns the best address suggestions for each result group. The ResultGroup property identifies the group with the best results. You can find the ResultGroup property under **Outputs** > **Results** > **StatusValues** > **ResultGroup** in the `AVJob.schema.json` file.

Address Verification adds each address suggestion to the group in which the address information most closely matches the reference data.

Note: Depending on the suggestions returned and on the MaxResultCount limit, Address Verification may not populate all result groups.

LanguageISO3 Code

When a result output contains data from the reference address database, Informatica Address Verification specifies the language in the ISO 639 3-letter code in the output. For example, `DEU` for German.

For transliterated output, Address Verification specifies the original language in the output. For example, `JPN` in case of romanized Japanese output.

If the reference address database does not contain language information, Address Verification returns a value of `???` in the LanguageISO3 property.

Note: If you request an address in more than one language or script, Address Verification returns the address results in the ISO 639 3-letter code for every language or script that you requested.

Level of Verification

The UsedVerificationLevel property reports the depth or level to which the input address was verified.

Match Percentage Value

The MatchPercentage property describes the degree of similarity between the input data and the reference data as a percentage value. Higher values indicate higher degrees of similarity.

Use the match percentage values to filter out results with extensive corrections. Similarly, use the match percentage values to identify interactive mode suggestions that have the least deviation from the input.

Note: Match percentage values refer explicitly to the similarity between the input and the reference data and not to the similarity between the input and the output address. In most cases, the degrees of similarity will be the same. However, this will not be the case if the output address contains an alternative form of a value, such as MUNICH for MÜNCHEN or PKWY for PARKWAY.

Script

Indicates the type of script used in the output address.

Address Count

When you verify an address in quick capture mode, the AddressCount property returns the number of complete addresses in the reference data that match the input address suggestion. For example, if the reference data contains 12 sub-buildings for a given house number, Address Verification returns a value of 12 in the AddressCount property.

The AddressCount property can also return the possible number of houses, buildings, or sub-buildings for output addresses that contain range-based house numbers, sub-buildings, or buildings.

For addresses in a Street, Locality, or PostalCode group, Address Verification returns an approximate value followed by a + symbol on the AddressCount property.

Address Verification returns an AddressCount value of 1 when the output address suggestion is a single complete address. Address Verification returns a value of ? in the AddressCount property if Address Verification cannot find a house number match in the database and fails to return address suggestions.

For information about the output groups in which Address Verification returns the address suggestions, see "Result Group" on page 107.

Result Quality

The ResultQuality property provides a general estimate of the likelihood that a mail carrier can deliver mail to an address. Use the result quality value with other status values, such as the process status value, to evaluate the address quality and deliverability.

The following table describes the result quality values:

Value	Summary	Description
6	Completely confident	Indicates that Address Verification verified all postally relevant elements.
5	Very confident	Indicates that Address Verification cannot completely verify the address. For example, Address Verification may be unable to verify secondary delivery information such as sub-building data. Or, the address might be a high-rise default address.
4	Secondary delivery point not verified	Indicates that Address Verification can check the primary delivery identifier but cannot verify the secondary delivery information in the address record. If an input address contains a unique ZIP Code, Address Verification corrects other input values and returns the address with a result quality value of 4.
3	Primary delivery identifier not verified	Indicates that Address Verification cannot check the primary delivery identifiers such as house number and P.O. Box number. In countries where the house number information is not available, the primary delivery identifier might be a building name.
2	Delivery information not verified	Indicates that Address Verification cannot find a match for the input street or delivery service information within the reference data. The address is verified to locality level.
1	Incomplete or contradictory address	Indicates that Address Verification cannot find a match for the address in the reference data. This can occur when locality information is not available in the reference data or the address includes contradictory postal code data.
0	Not an address	Indicates that the input is not a deliverable address.

Note: Address verification is likely to return a lower score for a poor-quality address in Verification only mode than in Verification with suggestions mode. Verification with suggestions mode can apply a relatively higher tolerance when calculating the result quality score, as it returns multiple addresses from which you can select the correct address.

API Return Codes

Informatica Address Verification returns numeric codes to indicate the status of API function calls. All API function calls to Address Verification receive an IDVE_I32 (32 bit signed integer) return code value.

- A value of 0 (zero) indicates success.
- A negative value of -20000 or below indicates a very critical error, and further processing is usually impossible. Immediately shut down the whole process as it might be in an unstable state.

- A negative value between -10000 and -12099 indicates a critical error, and further processing might be impossible.
- A positive value of 20000 or above indicates a non-critical error, and further processing is possible.
- A positive value between 10000 and 12099 indicates a warning, including a possible issue with configuration settings, address input, or output.

The return value must always be checked for by the calling logic. While the return values inform about fundamental errors, the actual verification results are returned through separate API functions.

Success

On successful completion of an API call, Informatica Address Verification returns a return code of 0 (zero).

Warnings

Informatica Address Verification returns one of the following return codes when an API call receives an unexpected result:

Code	Description
10000	The Core/System/MaxMemoryMegabytes setting in the IDVE config file was too small to fulfill all preloading settings.
11000	A database file was found but skipped, possibly because a newer file was preferred.
11001	A database file was not found.
11002	A database with an unsupported version was found.
11003	A corrupted database file was found.
12000	A license file is expired or not yet valid.
12001	The environment settings (for example, Operating System) in a license are incompatible with the current machine.
12002	A license with an unsupported version was found.
12003	The maximum amount of license files was reached. The additional license files will not be loaded.
12004	A license file was not found.
12005	No active license was found.
12006	A license file is corrupted.

Errors

Informatica Address Verification returns the following return codes if an API call failed to run because of an error condition:

Code	Description
20000	IDVE has already been initialized or is currently being initialized.
20001	The attempted operation was not valid.
21000	The output buffer is too small, no output was written.
21001	Buffer misalignment, for example an IDVE_WChar* points to an odd address.
21002	No configuration was given as a parameter for IDVE_Initialize().
21003	A pointer parameter is NULL.
21004	A numeric parameter is 0.
21005	A parameter is out of range or illegal.
21006	Multiple parameters are present that are mutually exclusive.
22000	The syntax of a parameter or value is not valid.
22001	The given URI is not valid either for this operation or in general.
22002	JSON string is not valid.
22003	Missing access rights for this JSON operation.
22004	A JSON value is not available.
22005	A JSON value is of a different type (for example, called IDVE_GetString() on an integer value).
22006	A JSON integer value is outside the bounds of the provided integer type. Choose a larger integer type.
22007	The character sequence of a JSON string is not valid (for example, it contains escape sequences that are not valid).
22008	The JSON file is in an unsupported encoding.
23000	Subfolder FileSetA or FileSetB (or both) were not found on disk.
23001	The FileSetsInfo file has content that is not valid or could not be found.
24000	No free job is available.
24001	The job ID is not valid.
25000	Payload exceeded IPC capacity.
25001	Lost connection to FunctionServer instance, a restart has been initiated.

Critical Errors

If a critical error occurs, Informatica Address Verification returns one of the following return codes:

Code	Description
-10000	IDVE has not been initialized (call IDVE_Initialize() to initialize) or is still being initialized. Address Verification does not allow any function call other than IDVE_Initialize() or IDVE_Deinitialize() after this error.
-10001	A version mismatch between IDVE components was detected, and the engine did not initialize.
-10002	IDVE is trapped in an unrecoverable error state. Call IDVE_Deinitialize() after this error.
-11000	A FunctionServer process could not be started. If the error occurs during initialization, the initialization will not complete. If the error occurs after initialization, the engine enters an unrecoverable state and you must call IDVE_Deinitialize().
-12000	A memory allocation request failed. The engine invariably remains in a functioning state. However, you may consider re-initializing the engine.
-12001	Access to a system resource failed (for example, files and mutexes). Re-initialize the engine after this error.

Very Critical Errors

Very critical errors are extremely unlikely to occur. When a very critical error occurs, we recommend that you reinitialize the Address Verification engine and report the error to Informatica Global Customer Support.

The following return calls indicate very critical errors:

Code	Description
-20000	An unknown exception has been thrown.
-20001	An internal error has occurred.
-20002	An internal assertion has failed.

Difference in Warning and Error Handling in C, Java, and Microsoft .NET Installations

Address Verification returns warning and error messages with identical meanings in C, Java, and .NET. However, the messages in Java and .NET are defined in an enum type of name `IDVEStatusCode`. Also, the Java and .NET messages do not have the `IDVE_SC_` prefix that the messages in C use.

Warning and Error Code Handling in C

Each API function has a return type `IDVE_StatusCode` and an optional argument that accepts a pointer to an output string buffer. When this argument is set, it must point to a buffer with a size greater than or equal to the `IDVE_EXT_STATUS_MSG_BUFFER_SIZE` value. Each function returns one of the documented status codes. If

the string buffer is provided, the function additionally fills the string buffer with an extended status message. This may be the static descriptive text in the `IDVE.h` file or a more detailed explanation of the warning or error.

Warning and Error Code Handling in Java and Microsoft .NET

Each API function is overloaded with an additional argument of type `IDVEStatus`. The `IDVEStatus` argument can store an `IDVE_StatusCode` value and an extended status message.

You can call the API function overload with or without the `IDVEStatus` argument. When you call a Java or .NET function, it internally calls the corresponding C function and evaluates the `IDVE_StatusCode` and extended status message that it will return.

If the `IDVE_StatusCode` is of type error, critical error, or very critical error, then the function throws an exception of type `IDVEException`. When you capture the error, you can extract an object of type `IDVEStatus` from the exception that contains the `IDVE_StatusCode` value and the extended status message.

If the underlying C function returns a status code of OK or returns a warning-level status code, the optional `IDVEStatus` argument is filled with that information and the function returns normally. If you do not call the function overload with the `IDVEStatus` argument, then this information is lost and you will not see the warnings. Exceptions for errors are thrown regardless of whether you call the function overload.

APPENDIX A

Geocode Countries

This appendix includes the following topic:

GeoCoding Databases

This appendix identifies the countries for which you can retrieve geocodes, the geocoding databases that you can install, and the level of geocode precision that Informatica Address Verification can achieve with each database.

The following table describes the geocode capabilities that Address Verification currently supports:

Country	Database File Name	Geocoding Precision
Afghanistan	AFG_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Albania	ALB_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Algeria	DZA_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Andorra	AND_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Andorra	AND_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Angola	AGO_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Anguilla	AIA_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Antarctica	ATA_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Antigua and Barbuda	ATG_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Argentina	ARG_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Argentina	ARG_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Armenia	ARM_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Aruba	ABW_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter

Country	Database File Name	Geocoding Precision
Australia	AUS_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Australia	AUS_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Australia	AUS_ADV_ENR_GAP_001_6_1_0.MD6	ArrivalPoint
Australia	AUS_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Australia	AUS_ADV_ENR_GRT_001_6_1_0.MD6	Rooftop
Austria	AUT_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Austria	AUT_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Austria	AUT_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Azerbaijan	AZE_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Bahamas	BHS_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Bahrain	BHR_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Bahrain	BHR_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Bahrain	BHR_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Bangladesh	BGD_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Barbados	BRB_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Belarus	BLR_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Belgium	BEL_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Belgium	BEL_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Belgium	BEL_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Belize	BLZ_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Benin	BEN_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Bermuda	BMU_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Bhutan	BTN_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Bolivia	BOL_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Bolivia	BOL_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Bosnia and Herzegovina	BIH_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Botswana	BWA_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter

Country	Database File Name	Geocoding Precision
Brazil	BRA_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
British Indian Ocean Territory	IOT_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Brunei Darussalam	BRN_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Burkina Faso	BFA_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Bulgaria	BGR_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Bulgaria	BGR_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Burundi	BDI_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Cambodia	KHM_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Cameroon	CMR_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Canada	CAN_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Canada	CAN_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Canada	CAN_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Central African Republic	CAF_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Chad	TCD_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Chile	CHL_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Chile	CHL_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Chile	CHL_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
China	CHN_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Colombia	COL_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Comoros	COM_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Congo	COG_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Congo, The Democratic Republic of the	COD_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Cook Islands	COK_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Costa Rica	CRI_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Côte d'Ivoire	CIV_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Cuba	CUB_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Curaçao	CUW_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter

Country	Database File Name	Geocoding Precision
Curaçao	CUW_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Curaçao	CUW_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Croatia	HRV_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Croatia	HRV_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Cyprus	CYP_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Czech Republic	CZE_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Czech Republic	CZE_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Denmark	DNK_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Denmark	DNK_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Djibouti	DJI_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Dominica	DMA_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Ecuador	ECU_ADV_ENR_GST_000_6_1_0.MD6	PostalCodeCenter
Egypt	EGY_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Equatorial Guinea	GNQ_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Eritrea	ERI_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Estland	EST_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Estland	EST_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Estonia	EST_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Eswatini	SWZ_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Ethiopia	ETH_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Falkland Islands	FLK_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Faroe Islands	FRO_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Faroe Islands	FRO_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Fiji	FJI_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Finland	FIN_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Finland	FIN_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Finland	FIN_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop

Country	Database File Name	Geocoding Precision
France	FRA_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
France	FRA_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
France	FRA_ADV_ENR_GAP_001_6_1_0.MD6	ArrivalPoint
Gabon	GAB_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Gambia	GMB_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Georgia	GEO_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Germany	DEU_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Germany	DEU_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Germany	DEU_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Ghana	GHA_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Gibraltar	GIB_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Gibraltar	GIB_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Greece	GRC_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Greece	GRC_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Greece	GRC_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Greenland	GRL_ADV_ENR_GST_000_6_1_0.MD6	PostalCodeCenter
Grenada	GRD_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Guatemala	GTM_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Guinea	GIN_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Guinea-Bissau	GNB_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Guyana	GUY_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Haiti	HTI_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Hong Kong	HKG_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Hong Kong	HKG_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Hungary	HUN_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Hungary	HUN_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Iceland	ISL_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter

Country	Database File Name	Geocoding Precision
Iceland	ISL_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
India	IND_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
India	IND_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Indonesia	IDN_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Iran, Islamic Republic of	IRN_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Iraq	IRQ_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Ireland	IRL_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Ireland	IRL_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Israel	ISR_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Italy	ITA_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Italy	ITA_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Italy	ITA_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Ivory Coast	CIV_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Japan	JPN_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Jordan	JOR_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Kazakhstan	KAZ_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Kenya	KEN_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Kiribati	KIR_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Korea, Democratic People's Republic of	PRK_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Korea, Republic of	KOR_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Korea, Republic of	KOR_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Kosovo	XKS_ADV_ENR_GST_000_6_3_0.MD6	StreetCenter
Kuwait	KWT_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Kyrgyzstan	KGZ_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Lao, People's Democratic Republic	LAO_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Latvia	LVA_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Latvia	LVA_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint

Country	Database File Name	Geocoding Precision
Lebanon	LBN_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Lesotho	LSO_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Liberia	LBR_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Libyan Arab Jamahiriya	LBY_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Liechtenstein	LIE_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Liechtenstein	LIE_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Lithuania	LTU_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Lithuania	LTU_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Luxembourg	LUX_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Luxembourg	LUX_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Macau	MAC_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Macau	MAC_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Macedonia, The Former Yugoslav Republic of	MKD_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Macedonia, The Former Yugoslav Republic of	MKD_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Macedonia, The Former Yugoslav Republic of	MKD_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Madagascar	MDG_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Malawi	MWI_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Malaysia	MYS_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Malaysia	MYS_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Maldives	MDV_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Mali	MLI_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Mauritania	MRT_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Mauritius	MUS_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Mexico	MEX_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Mexico	MEX_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Mexico	MEX_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop

Country	Database File Name	Geocoding Precision
Moldova	MDA_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Moldova	MDA_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Moldova	MDA_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Mongolia	MNG_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Montenegro	MNE_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Montserrat	MSR_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Morocco	MAR_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Mozambique	MOZ_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Myanmar	MMR_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Namibia	NAM_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Nauru	NRU_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Nepal	NPL_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Netherlands	NLD_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Netherlands	NLD_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Netherlands	NLD_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
New Zealand	NZL_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
New Zealand	NZL_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Nicaragua	NIC_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Niger	NER_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Nigeria	NGA_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Niue	NIU_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Norfolk Island	NFK_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Norway	NOR_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Norway	NOR_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Norway	NOR_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Oman	OMN_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Pakistan	PAK_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter

Country	Database File Name	Geocoding Precision
Panama	PAN_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Papua New Guinea	PNG_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Paraguay	PRY_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Peru	PER_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Peru	PER_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Philippines	PHL_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Philippines	PHL_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Philippines	PHL_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Pitcairn	PCN_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Poland	POL_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Poland	POL_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Poland	POL_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Portugal	PRT_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Portugal	PRT_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Portugal	PRT_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Qatar	QAT_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Qatar	QAT_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Qatar	QAT_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Romania	ROU_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Romania	ROU_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Romania	ROU_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Russia	RUS_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Russia	RUS_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Rwanda	RWA_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Saint Helena	SHN_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Saint Kitts and Nevis	KNA_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Saint Lucia	LCA_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter

Country	Database File Name	Geocoding Precision
Saint Vincent and the Grenadines	VCT_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Samoa	WSM_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Sao Tome and Principe	STP_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Saudi Arabia	SAU_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Senegal	SEN_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Serbia	SRB_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Serbia	SRB_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Serbia	SRB_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Seychelles	SYC_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Sierra Leone	SLE_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Singapore	SGP_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Singapore	SGP_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Sint Maarten (Dutch part)	SXM_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Slovakia	SVK_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Slovakia	SVK_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Slovenia	SVN_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Slovenia	SVN_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Solomon Islands	SLB_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Somalia	SOM_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
South Africa	ZAF_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
South Africa	ZAF_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
South Georgia and the South Sandwich Islands	SGS_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Spain	ESP_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Spain	ESP_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Sri Lanka	LKA_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
St Martin	SXM_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Sudan	SDN_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter

Country	Database File Name	Geocoding Precision
Suriname	SUR_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Sweden	SWE_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Sweden	SWE_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Switzerland	CHE_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Switzerland	CHE_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Switzerland	CHE_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Syrian Arab Republic	SYR_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Taiwan	TWN_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Taiwan	TWN_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Taiwan	TWN_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Tajikistan	TJK_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Tanzania, United Rep.	TZA_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Thailand	THA_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Thailand	THA_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Thailand	THA_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Timor-Leste	TLS_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Togo	TGO_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Tokelau	TKL_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Tonga	TON_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Trinidad and Tobago	TTO_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Tunisia	TUN_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Turkey	TUR_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Turkey	TUR_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Turkmenistan	TKM_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Turks and Caicos Islands	TCA_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Tuvalu	TUV_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Uganda	UGA_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter

Country	Database File Name	Geocoding Precision
Ukraine	UKR_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Ukraine	UKR_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
United Arab Emirates	ARE_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
United Kingdom	GBR_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
United Kingdom	GBR_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
United Kingdom	GBR_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
United States	USA_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
United States	USA_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
United States	USA_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Uruguay	URY_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Uruguay	URY_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Uruguay	URY_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Uzbekistan	UZB_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Vanuatu	VUT_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Venezuela	VEN_ADV_ENR_GST_000_6_1_0.MD6	StreetCenter
Venezuela	VEN_ADV_ENR_GAP_000_6_1_0.MD6	ArrivalPoint
Venezuela	VEN_ADV_ENR_GRT_000_6_1_0.MD6	Rooftop
Virgin Islands, British	VGB_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Western Sahara	ESH_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Yemen	YEM_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Zambia	ZMB_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter
Zimbabwe	ZWE_ADV_ENR_GST_000_6_1_0.MD6	LocalityCenter

APPENDIX B

Reverse Geocoding Coverage

The following table lists the countries for which Informatica Address Verification can perform reverse geocoding:

Reverse Geocoding Countries		
Andorra	Argentina	Australia
Austria	Bahrain	Belgium
Bolivia	Brazil	Bulgaria
Canada	Chile	Croatia
Curaçao	Czech Republic	Denmark
Estonia	Faroe Islands	Finland
France	Germany	Gibraltar
Greece	Hong Kong	Hungary
Iceland	India	Ireland
Israel	Italy	Kuwait
Latvia	Liechtenstein	Lithuania
Luxembourg	Macau	Macedonia, The Former Yugoslav Republic of
Malaysia	Mexico	Moldova
Netherlands	New Zealand	Norway
Palestine	Peru	Philippines
Poland	Portugal	Qatar
Romania	Russian Federation	Serbia
Singapore	Sint Maarten	Slovakia
Slovenia	South Africa	South Korea

Reverse Geocoding Countries		
Spain	Sweden	Switzerland
Taiwan	Thailand	Turkey
Ukraine	United Kingdom	United States
Uruguay	Venezuela	

APPENDIX C

Certified Mode Values

This appendix includes the following topics:

CASS Certification Values

The following table describes the values and data indicators that Informatica Address Verification can return when you verify a United States address in certified mode:

Field	Property	Description
Barcode	BarCode	An 11-digit number that represents the delivery point for the record. It consists of the nine-digit ZIP+4 code and the two-digit Delivery Point Answer.
Carrier Route Answer	CarrierRoute	A four-character code assigned to a mail delivery or collection route within a 5-digit ZIP Code. The first character of the code is alphabetical and the last three are numeric: - Bnnn = PO box - Hnnn = Highway contract - Rnnn = Rural route - Cnnn = City delivery - Gnnn = General delivery
Concatenation of DPV Footnotes*	DPVFootnoteComplete	All data values from the populated DPV Footnote fields in a single string.
Congressional District	CongressionalDistrict	A standard value that identifies a geographic area served by a member of the United States House of Representatives or Senate. If the address is an Army/Air Force (APO) address or a fleet post office (FPO) address, the field is blank. If there is only one member of congress within the state, the value is AL (At Large).

Field	Property	Description
Default Flag	DefaultFlag	Indicates that the record matched to a high-rise, rural route, or street default record in the ZIP + 4 data. The flag has the value Y for a matching record.
Delivery Point Answer	DeliveryPoint	The final two digits of the house/box number, or if a High-rise record is matched, the secondary unit number that represents the delivery point information. The Delivery Point Answer contributes to the 11-digit delivery point barcode (DPBC).
Delivery Point Check Digit Answer	DeliveryPointCheckDigit	A number that you can add to the sum of the other digits in the delivery point barcode (DPBC) to yield a number that is a multiple of ten. For example: ZIP+4 code = 123456789 Delivery Point Answer = 01 Sum of digits (1+2+3+4+5+6+7+8+9+0+1) = 46 Check digit = 4 (46+4 = 50)
DPV 3553 Confirmation*	DPVConfirmation3553	Indicates whether to add the address to the total number of addresses recorded in the ZIP + 4/DPV Confirmed column on form 3553: - Increment the total number of addresses. - Do not increment the total number of addresses.
DPV CMRA Indicator*	DPVCMRA	Indicates the result of the call to the DPV CMRA (Commercial Mail Receiving Agent) hash table: - Y =Address was found in the CMRA table. - N =Address was not found in the CMRA table. - Blank = Address was not presented to the hash table.
DPV Confirmation Indicator*	DPVConfirmation	Indicates the result of the call to the Delivery Point Validation (DPV) Confirmation hash table: - Y =Address was DPV confirmed for both primary and (if present) secondary numbers. - D =Address was DPV confirmed for the primary number only, and secondary number information was missing. - S= Address was DPV confirmed for the primary number only, and the secondary number information was present but not confirmed. Or, a single trailing alpha character on a primary number was dropped to make a DPV match, and secondary information is required. - N= The primary number information failed to DPV confirm. - Blank = Address was not presented to the hash table.

Field	Property	Description
DPV Confirmation Enhanced*	DPVConfirmationEnhanced	Provides additional information on the reasons for the DPV Status assignment: - Y= Address was DPV confirmed for both primary and secondary numbers to determine a valid delivery point. - D= Address was DPV confirmed for the primary number only, and secondary number information was missing. - S= Address was DPV confirmed for the primary number only, and the secondary number information was present but not confirmed. Or, a single trailing alpha character on a primary number was dropped to make a DPV match, and secondary information is required. - N= The primary number information failed to DPV confirm. - R= Address was DPV confirmed but assigned to phantom route R777 or R779. The USPS does not provide delivery. - Blank= Address was not presented to the hash table.
DPV Door Not Available*	DPVDNA	Identifies addresses that do not provide a door or entry point that the postal carrier can access. A Door Not Available (DNA) address might not provide a door for mail delivery, or the mailbox might reside behind a locked gate. The DPV DNA indicator can contain the values of Y or N: - Y = Address does not have a door available for postal delivery. - N = Address has a door available for postal delivery. - Blank = Address was not presented to the hash table.
DPV Drop Indicator*	DPVDropIndicator	Indicates the status of the address in the DSF2 Drop Indicator table. The Drop Indicator table identifies mailbox addresses that serve multiple households or businesses. The DPV Drop indicator can contain the values of Y or N: - Y= Address was found in the table. - N= Address was not found in the table. - Blank = Address was not presented to the hash table.
DPV False Positive Indicator*	DPVFalsePositive	Indicates the results of the call to the DPV False Positive hash table: - Y = Address was found in the False Positive table. - N = Address was not found in the False Positive table. - Blank = Address was not presented to the hash table.

Field	Property	Description
DPV Footnote 1* DPV Footnote 2* DPV Footnote 3* DPV Footnote 4* DPV Footnote 5*	DPVFootnote1 DPVFootnote2 DPVFootnote3 DPVFootnote4 DPVFootnote5	The footnote(s) returned for an address from DPV processing: - AA - Input address matched to the ZIP + 4 product. - A1 - Input address not matched to the ZIP + 4 product. - BB - Input address matched to DPV for both primary and secondary numbers necessary to determine a valid delivery point. - CC - Input address primary number matched, secondary number not matched, secondary number not required. - C1 - Input address primary number matched, secondary number not matched, secondary number required. - F1 - Input address matched to a military address. - G1 - Input address matched to a general delivery address. - IA - Informed address identified. - M1 - Input address primary number missing. - M3 - Input address primary number not valid. - N1 - Input address primary number matched to DPV, but required secondary number is missing. - PB - Identified Post Office Box Street Address. - P1 - Input address Post Office Box, Rural Route, or Highway Contract box number missing. - P3 - Input address Post Office Box, Rural Route, or Highway Contract box number not valid. - RR - Input address matched to a Commercial Mail Receiving Agency (CMRA), but a Private Mailbox (PMB) designator is present (PMB 123 or # 123). - R1 - Input address matched to CMRA, but PMB designator is not present (PMB 123 or # 123). - R7 - Addresses that are assigned to a phantom route R777 or R779. - TA - Input address primary number matched to DPV by dropping trailing alphabetic character. - U1 - Input address matched to a unique ZIP Code.
DPV No Secure Location*	DPVNSL	Identifies an address that does not provide a reliable mailbox or reception point for mail: - Y = Location is not secure. - N = Location is secure. - Blank = Address was not presented to the hash table.
DPV PBSA*	DPVPBSA	Indicates that the address is a Post Office Box Street address (PBSA): - Y =Address was found in the PBSA table. - N =Address was not found in the PBSA table. - Blank = Address was not presented to the hash table. Address Verification also returns the footnote PB for a PBSA address.
DPV Throwback Indicator*	DPVThrowBack	Indicates a valid street address for which the USPS forwards mail to a Post Office Box: - Y = The address is a Throwback address. - N = The address is not a Throwback address. - Blank = Address was not presented to the hash table.
DSF2 No Stats Indicator*	DPVNoStatIndicator	Indicates the results of the call to the DPV No-Stat table: - Y = Address was found in the No-Stat table. - N = Address was not found in the No-Stat table. - Blank = Address was not presented to the hash table.

Field	Property	Description
DSF2 No Stats Reason*	DPVNoStatReason	A single-digit code that identifies the reason why an address returned a Y in the *DSF2 No Stats Indicator* field. The code can have the following values: - *1 - IDA* Internal Drop Address. The verified address does not physically receive mail. Instead, the USPS delivers mail to a 'drop' address associated with the verified address. - *2 - CDS* The address identifies a new construction that cannot yet accept delivery. Or, the address lies on a Rural Route, Highway Contract Route, or Contract Delivery Service route and the delivery point is unoccupied for more than 90 days. - *3 - Collision* The address does not DPV confirm. Address Verification users will not see the collision value, because address validation will change the NoStats indicator to N in this case and clear the NoStats Reason code. - *4 - CMZ* The address is in a college, military, or other zone. A CMZ address is a ZIP+4 address that the USPS has added to the reference data. - *5 - Regular No-Stat* The address is no longer deliverable or lies on an R777 route, Or, the address includes a Post Office Box that has never been rented or is not available to rent.
DSF2 Vacant Indicator*	DPVVacantIndicator	Indicates the results of the call to the DPV Vacant table: - Y = Address was found in the Vacant table. - N = Address was not found in the Vacant table. - Blank = Address was not presented to the hash table.
Early Warning System (EWS) Return Code*	EWSReturnCode	Indicates whether the address was found in the EWS data: Y = Address was found in the EWS data, thus resulting in a ZIP + 4 No Match. Blank = Address was not found in the EWS data.
eLOT Ascending/ Descending	ELOTFlag	The Enhanced Line of Travel (eLOT) order in which delivery points are delivered within a given add-on code. For example, if the code is A (Ascending) and the house numbers are 1, 3, and 5, the carrier delivers 1, 3, and 5, in that order (low to high). If the code is D (Descending), the carrier delivers first to 5, then 3, and then 1 (high to low).
eLOT Sequence Number	ELOTSequence	A number that indicates the order in which add-on codes are arranged within a given carrier route.
High-rise Default	HighriseDefault	A flag that indicates that the record is assigned to a default high-rise record.
High-rise Exact	HighriseExact	Identifies high-rise addresses that contain unit identifiers.

Field	Property	Description
LACS Indicator*	LACSIndicator	Identifies an address that matched to a LACS address in the ZIP + 4 file. A LACS address is an address that was converted to the city-style address format so that emergency vehicles can more easily find the address location. Return values: - L = Address matched to a LACS address in the ZIP + 4 file. - Blank = Address is not a LACS address in the ZIP + 4 file.
LACSLink Indicator*	LACSLinkIndicator	An indicator returned when the LACSLink hash tables are queried: - Y = The input record matched a record in the master file. - N = The input record did not match a record in the master file. - Blank = Address was not presented to the hash table.
LACSLink False Positive Indicator*	LACSLinkFalsePositive	Indicates the results of the call to the LACSLink False Positive hash table: - Y = Address was found in the table. - N = Address was not found in the table. - Blank = Address was not presented to the table.
LACSLink Return Code*	LACSLinkReturnCode	A return value from LACSLink processing: - *A = LACS record match* A new address can be furnished. The input record matched to a record in the master file. - *00 = No match* A new address cannot be furnished. The input record cannot be matched to a record in the master file. - *14 = Found LACS record, but new address did not convert at run time.* The new address cannot be converted to a deliverable address. The input record matched to a record in the master file. - *92 = LACS record, secondary number dropped from input address.* The record is a ZIP + 4 street-level or high-rise match. The input record matched to a master file record, but the input address had a secondary number and the master file record did not.
Non-Delivery Days*	DPVNDD	A seven-character code that identifies the day or days of the week on which an address cannot receive mail. The code contains a seven-character string that represents the days of the week from Sunday through Saturday. Address Verification returns the first letter of a weekday in the corresponding position in the code if the address does not receive mail on that day. Address Verification returns a dash symbol in the corresponding position otherwise.
Non-Delivery Days Flag*	DPVNDDFlag	Returns Y if the address is not deliverable on one or more days of the week. Otherwise, returns N. Address Verification populates the Non-Delivery Days field for every address that returns Y in the Non-Delivery Days Flag field. The Non-Delivery Days Flag can contain the values of Y or N: - Y= Address was found in the table. - N= Address was not found in the table. - Blank = Address was not presented to the hash table.

Field	Property	Description
P. O. Box Only	PoBoxOnly	Indicates if the address is located in a ZIP code that contains post office box addresses only. Y = Address is in a PO Box only Delivery Zone.
Record Type Code	RecordType	An alphabetic value that identifies the type of data in the record. Record type codes include the following: - F = Firm - G = General delivery - H = High-rise - P = PO box - R = Rural route/highway contract - S = Street
Residential Delivery Indicator *	RDI	Indicates if the delivery point is residential: - Y = Indicates residential delivery. - N = Not residential delivery. - Blank = Did not query the Residential Delivery Indicator (RDI) data.
Rural Route Default	RuralRouteDefault	A flag that indicates that the record is assigned to a default rural route record. This occurs when the input house number does not match to the primary numbers in the reference data and there is a corresponding rural route record with no primary numbers.
Rural Route Exact	RuralRouteExact	Indicates if the address matches a rural route address in the USPS address reference data set.
SuiteLink Return Code*	SuiteLinkReturnCode	A return value of SuiteLink Processing: - *A = SuiteLink Record Match* The input record matched to a record in the master file. An improved business address can be furnished. - *00 = No Match* Business address not improved. The input record cannot be matched to a record in the master file. An improved business address can not be furnished.
ZIPMove Return Code*	ZIPMoveReturnCode	A return value of ZIPMove processing: - Y = ZIPMove match was made. - N = ZIPMove match was not made.
ZIP5 Valid	ZIP5ValidFlag	Indicates whether the address record can be added to Form 3553. Five-digit validation requires that the last line values of city, state, and ZIP Code correspond to each other. When CASS Software makes a ZIP+4 match, the five-digit ZIP Code is valid. **Note:** Five-digit validation applies to No Match ZIP+4 records.

Note: Fields marked with * will only be populated for United States customers, as per USPS licensing restrictions.

AMAS Certification Values

The following table describes the values and data indicators that Informatica Address Verification can return when you verify an Australia address in certified mode:

Field	Property	Description
Error Code	ErrorCode	A two-character code that represents the validity of the address with respect to the AMAS standard. The code can indicate that the address is fully valid, or the code can describe the reason why the address does not meet the AMAS standard.
Delivery Point Identifier	DeliveryPointID	An eight-digit identifier that Australia Post assigns to a delivery point. Each DPID is unique and randomly generated. Australia Post creates DPIDs in a range from 30,000,000 to 99,999,999.
Delivery Identifier	DeliveryID	An eight-digit identifier that Australia Post assigns to an entire street or locality. The identifier represents all the addresses in the street or locality. Australia Post creates the street DIDs in a range from 26,000,000 through 27,999,999 and the locality DIDs in a range from 28,000,000 through 28,999,999.
Lot Number	LotNumber	The lot reference number that a government department assigned to a property during the sub-division of a parcel of land and prior to road numbering.
Postal Delivery Number	PostalDeliveryNumber	The number information in the delivery service. For example, the post office box number in a P.O. Box address. Australia Post stores a postal delivery number as a five-digit number in the Postal Address File (PAF). Therefore, the PAF stores the postal delivery number 500 as 00500. If the address does not contain a postal delivery number, Address Verification does not populate the field.
Postal Delivery Number Prefix	PostalDeliveryNumberPrefix	The prefix character in a postal delivery number. For example, A in A500.
Postal Delivery Number Suffix	PostalDeliveryNumberSuffix	The suffix character in a postal delivery number. For example, A in 500A.
House Number 1	HouseNumber1	The primary house number in a street address. Australia Post stores a house number as a five-digit number in the PAF. Therefore, the PAF stores the house number 123 as 00123.
House Number 1 Suffix	HouseNumber1Suffix	An alphabetical suffix to a House Number 1 value. For example, A in 123A.

Field	Property	Description
House Number 2	HouseNumber2	The secondary house number in a street address. Australia Post stores a house number as a five-digit number in the PAF. Therefore, the PAF stores the house number 456 as 00456. If the address does not contain a secondary house number, Address Verification returns 00000 as the field value.
House Number 2 Suffix	HouseNumber2Suffix	An alphabetical suffix to a House Number 2 value. For example, B in 456B.

SendRight Certification Values

The following table describes the values and data indicators that Informatica Address Verification can return when you verify a New Zealand address in certified mode:

Field	Property	Description
Address Type	AddressType	The address type. Contains one of the following values: - Bag - Box - CMB Rural - CMB Urban - Counter - Rural - Urban
Delivery Service Type	DeliveryServiceType	Specifies the type of the delivery service. It may be PO Box, Private Bag, CMB, Response Bag, Counter Delivery, or Poste Restante.
Delivery Service Identifier	DeliveryServiceNumber	The box or bag number. It contains no leading zeros or spaces, separators, or other punctuation.
Delivery Service Locality	DeliveryServiceLocality	Part of the Suburb line for New Zealand addresses. Contains Box Lobby as the value.
House Number	HouseNumber	Number of the structure erected on the property associated with the delivery point.
House Number Alpha	HouseNumberAlpha	The alphabetical part of the house number of the property associated with the delivery point.
Validity Code	ValidityCode	Helps to determine the validity of the input address. Contains one of the following values: - Unique match - VALID-U - Base Address match - VALID-B - Not valid - INVALID
Rural Delivery Number	RdNumber	A six-character field that identifies the rural delivery route number of an address.

Field	Property	Description
Delivery Point Identifier	DeliveryPointID	An eight-digit number that uniquely identifies each delivery point in New Zealand.
Hygiene	Hygiene	A flag that indicates whether address cleansing is possible.
SOA Record Ignored	SOARecordIgnored	Indicates whether the Statement of Accuracy ignores the record. The following address records maybe ignored for SOA calculation: - Poste Restante - Private Bags with no number The field contains the value Excluded for an ignored record. Otherwise, the field is empty.

SERP Certification Values

The following table describes the values and data indicators that Informatica Address Verification can return when you verify a Canada address in certified mode:

Field	Property	Description
SERP category	Category	The verification category for the address. The category summarizes the result of the verification operation for the address. Contains one of the following values: - V. Address valid. - C. Address corrected. - N. Not valid. - VQ. Considered valid, but questionable because of insufficient data or input. For example, some "general delivery" addresses. - V1A. Valid apartment type record. - V2A. Valid commercial type record. - C1A. Corrected apartment type record. - C2A. Corrected commercial type record. The 1A and 2A types usually refer to database records containing building information.
Excluded	ExcludedFlag	Indicates that the value is excluded from SERP calculations because the sub-building input data is incorrect. Occurs in conjunction with SERP category N. The field contains the value EXCLUDED for an ignored record. Otherwise, the field is empty.

SNA Certification Values

The following table describes the values and data indicators that Informatica Address Verification can return when you verify a France address in certified mode:

Field	Property	Description
SNA category	Category	The verification category for the address. The category summarizes the result of the verification operation for the address. Contains one of the following values: - ORI. Address valid, output identical to input. - RES. Address valid or corrected, output not identical to input. - AVE. Address rejected, user input required. - NOK. Address rejected.