Informatica® SSA-NAME3
10.2

Best Practices Guide

Informatica SSA-NAME3 Best Practices Guide
10.2
December 2020

Publication Date: 2022-01-26

Table of Contents

Preface 6

Informatica Resources 6

- Informatica Network 6
- Informatica Knowledge Base 6
- Informatica Documentation 6
- Informatica Product Availability Matrices 7
- Informatica Velocity 7
- Informatica Marketplace 7
- Informatica Global Customer Support 7

Chapter 1: Introduction 8

Chapter 2: Major Concepts 9

Multiple Keys 9

Search Strategies 10

Match Purposes 10

Standard Populations 11

Custom Populations 11

Overriding Population Rules 11

Multi-Country Support 12

Chapter 3: Prototyping 13

Choosing the Data to Search 13

Choosing the Mode of Search 13

Defining a New Database Table, Column or File 14

Chapter 4: The Design Issues 15

What Naming Data is used in Searches 15

What Identification Data do we Match with 15

Objectives of Name Search and Matching Systems 16

File and Field Design Issues 17

The Name Change Transaction 18

The Telephone Book as Metaphor for Name Search Index Design 18

File Design for Optimal Name Search Performance 19

Coping with a Small % of Foreign Name and Address Data 20

When Partitioning Keys Makes Sense 21

Storing the Good with the Bad 21

Chapter 5: Standard Population Choices 23

Overview 23

Standard Populations. . . . 24
A Primer on Keys and Search Strategies. . . . 24
Key Fields. . . . 25
Key Levels. . . . 27
Search Levels. . . . 27
Match Purposes. . . . 29
Field Types. . . . 30
Purposes Types. . . . 31
Match Levels. . . . 40
Managing Population Rule Sets. . . . 41
Effect of File Size on Name Search Performance. . . . 41
Impact of Risk on the Search Transaction. . . . 42
The Critical Exhaustive Search. . . . 42
Balancing Missed Matches with Finding Too Much. . . . 43
Undermatching or Overmatching. . . . 44
Discovering the Missed Matches. . . . 44
The Importance of Prototyping with Production Data. . . . 45
Overview. . . . 45
Field Design for Multinational Systems. . . . 46
Deployment of Multinational Systems. . . . 47
Code Pages, Character Sets and other Encoding Issues. . . . 47
Unicode Issues. . . . 48
Transliteration Realities. . . . 49
Transliteration and Data Matching. . . . 49

Chapter 6: Customer Identification Systems. . . . 50

Overview. . . . 50
What Data to Use for Customer Look-up. . . . 50
Use of Full Name in the Customer Search. . . . 51
Responsibilities of the Customer Take-on Transaction. . . . 51
The Customer Take-on Transaction and Duplication. . . . 52

Chapter 7: Identity Screening Systems. . . . 53

Overview. . . . 53
Characteristics of a Screening Application. . . . 53
Identity Data in Screening Systems. . . . 54
How do you Prove that you have not Missed Any Records?. . . . 55
The False Hit Problem. . . . 55

Chapter 8: Fraud and Intelligence Systems. . . . 56

Overview. . . . 56
Identity data in Fraud and Intelligence Systems. . . . 56
What Search Strategy to Use. . . . 57

How Well do these Systems have to Match?. 57

Chapter 9: Marketing Systems. 58

Different Uses of Names and Addresses in Marketing Systems. 58

Conflicting Needs of Name and Address Data. 58

Index. 60

Preface

Welcome to the Informatica SSA-NAME3 Best Practices Guide. It is intended to be read by the designer of the application or system that uses SSA-NAME3.

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.

CHAPTER 1

Introduction

This manual brings together relevant sections of the SSA-NAME3 manual set in order to provide the reader with a single source of best practice information for implementing SSA-NAME3 applications.

It is intended to be read by the designer of the application or system that uses SSA-NAME3. It is best read prior to embarking on the SSA-NAME3 application design.

All of the sections in this manual can be found distributed in the other SSA-NAME3 manuals.

CHAPTER 2

Major Concepts

This chapter includes the following topics:

- Multiple Keys, 9
- Search Strategies, 10
- Match Purposes, 10
- Standard Populations, 11
- Custom Populations, 11
- Overriding Population Rules, 11
- Multi-Country Support, 12

Multiple Keys

The initial step in a system that needs to support searching on names or addresses is to build keys from the name or address data.

These keys must be able to overcome the error and variation in the data, including missing words, extra words and word sequence variations. To provide this level of reliability, multiple keys must be generated for each name or address.

Out-of-the-box, SSA-NAME3 provides three levels of keying. **Standard** keys are used by the typical user. **Extended** keys are used by the user with critical search needs. **Limited** keys are used by the user who is concerned with disk space and is willing to trade some reliability for savings in space.

The keys must also support efficient access. The Algorithms used to build SSA-NAME3 Keys are designed to provide efficient access, even for common names.

In addition, to get optimal performance it is necessary for the user to store these keys in a new table that contains not only the SSA-NAME3 Keys and foreign key to the source record, but also any additional identity data that will be used for matching, filtering or display purposes. This is so that the search application does not need to perform table joins at search time to get the data necessary for matching and display.

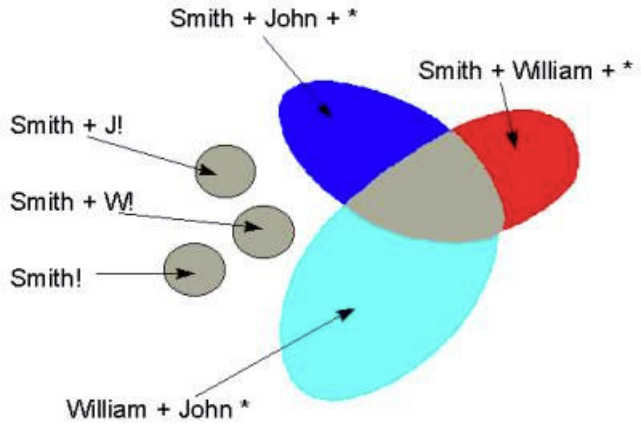
The SSA-NAME3 Key column must be indexed.

Search Strategies

The first step in an online or batch process that has to find records about a person, organization or address is to get "candidates" from the database using the keys defined for the names or addresses. The "search strategy" used to achieve this goal must balance the conflicting objectives of making sure it does not miss possible candidates, yet at the same time not slow the process with too many irrelevant candidates. SSA-NAME3 provides a variety of "search strategies" or access paths to data using names or addresses as search keys.

Applications dealing with relatively reliable and complete data can use a high performance strategy, while applications dealing with less reliable data or with more critical problems need more complex strategies. For any name or address, SSA-NAME3 provides the business application with a logical access path to the set of records that are most likely to include relevant matching records. Out-of-thebox, four search strategies or "search levels" are provided. These are *Narrow* , *Typical* , *Exhaustive* and *Extreme* .

For example, an Exhaustive search for the name "John William Smith" may result in:



This figure is a diagrammatic example of a search strategy where all records containing two similar words are found, as well as the major word (in this case "Smith") together with explicitly only the initials well as a set of records that contain only the major word.

Match Purposes

The second step in most search and matching applications requires that the system can automate or assist the operator in "choosing" the relevant matching records. In online transactions, where the operator makes the final choice, this Matching process can be used to eliminate the entirely irrelevant records, or rank them in order of relevance. In many transactions the application system can be made more reliable, if instead of allowing the operator the choice, the Matching process is fully automated. In the majority of batch systems the Match decisions have to be fully automated.

To allow the easy implementation of such high quality Matching processes, the SSA-NAME3 software provides a set of Match Purposes to "mimic" the matching decisions made by the very best business users. A Match Purpose will compare two records, typically the search record and a file record, and compute a Score and Match Decision.

The out-of-the-box Match Purposes provided are used for determining same `Person_Name`, `Individual`, `Resident`, `Household`, `Family`, `Wide_Household`, `Address`, `Organization`, `Division`, `Corporate Entity`,

`Contact` and `Wide_Contact`. In addition, three different Match Levels can be chosen in conjunction with each Match Purpose. These Match Levels are `Conservative`, `Typical` and `Loose`.

Each Match Purpose will have one or more mandatory fields and a number of optional fields. For each Match Purpose and Match Level, special rules are defined for each field of available data that control how the results from each field should be combined into the overall Match Score. Pre-set Score thresholds deliver a Match "Decision" as either `Accept`, `Undecided`, or `Reject`. The user's application can override these thresholds, or use the raw Score to determine its own thresholds.

Standard Populations

The rules that define how the Key Building, Search Strategies and Match Purposes operate on a particular population of data are packaged with SSA-NAME3 in what are called Standard Population sets. There is one Standard Population set for each country, language or population that Informatica supports.

Each Standard Population set supports:

- Key Building and Searching on Names (Person and Organization) and Addresses
- Selectable Key and Search levels
- Matching for Purposes such as Same Person Name, Same Organization, Same Contact, Same Individual, Same Resident, Same Household, Same Division and Same Address
- Selectable Match levels and thresholds.

Custom Populations

For the user with unusual or special needs, Informatica provides a customization service that results in the generation of a `Custom Population` set. This can be plugged in and used just like a Standard Population set.

This Customization Service might be used, for example, by the user that needs to search on an entity other than a name or address, for example a Song Title.

Overriding Population Rules

SSA-NAME3 includes the ability to override certain types of Population Rules. There are two levels at which these overrides can be managed.

Via the **Population Override Manager**, a data analyst trained in the syntax of the SSA-NAME3 Edit rules, and versed in the consequences of making changes that may effect search and match quality and performance, and possibly require indexes to be rebuilt, can add or remove Edit rules from the Population set. Via this utility, the analyst can also replace the packaged Frequency tables with ones built from the organization's own data.

Via the **Edit Rule Wizard**, a business or non-IT user can safely add certain types of Edit rules, typically new nicknames or synonyms, new noise words, or new phrase replacement rules, without requiring involvement from IT or the need to re-build indexes.

Multi-Country Support

Because SSA-NAME3 is ready to deal with various populations of data, it is practical to build systems in a manner that allows the same system to be deployed for many countries.

It is also practical to build systems where the data in one database is mixed from several countries. It is possible to "tune" critical systems to handle the fact that the searches are being generated in one country's language and character set yet the database has been built according to international rules so that it is usable in many countries.

In countries like Greece, or Israel, where both Roman character forms of names and addresses as well as the local country language and character forms are common, SSA-NAME3 allows searching and matching from Roman form to local language and vice versa.

SSA-NAME3 comes with Standard Population rule sets for over 50 countries and languages. These Standard Population sets include all of the Key Building, Search Strategy and Matching services required to effectively search and match on identity data sourced from that country and character set.

Informatica can also generate Custom Populations for new countries or for unusual searching and matching requirements.

SSA-NAME3 CJK-SUPPORT

SSA-NAME3 CJK-SUPPORT is the name given to a separately licensable product that expands the full facilities of SSA-NAME3 to handle the special nature of Chinese, Japanese and Korean data. Its features include the ability to recognize and encode double-byte characters in names and addresses and handle special representations of Chinese numbers. It also supports the mixed use of local phonetic (for example, Katakana) and Roman, often used to record foreign names and addresses in these countries.

Unicode Support

SSA-NAME3 also supports Unicode source data. For more information on how to use and specify Unicode input, see the *API REFERENCE* manual.

CHAPTER 3

Prototyping

This chapter includes the following topics:

- Choosing the Data to Search, 13
- Choosing the Mode of Search, 13
- Defining a New Database Table, Column or File, 14

Choosing the Data to Search

SSA-NAME3 can be used to search on the following types of names:

- Person names;
- Organization (Company/Business) names (can also be used to support mixed Person/Organization names);
- Addresses (typically that part of the Address up to but not including the locality details, otherwise known as the Street address or Address Part 1).

It is important to choose data that is available in both reasonable quality and quantity. It is not a good idea to build the prototype to search on the made-up names in the development database. It is a much better idea to use an extract or copy of some real data.

Choosing the Mode of Search

The following factors may help you choose between an online search prototype and a batch prototype:

- What is the main purpose for evaluating or purchasing SSA-NAME3?
- How quickly does the prototype need to be delivered?
- What are the analyst/programmers skills?

For example, if the main purpose for using SSA-NAME3 is for batch file matching, or the prototype needs to be delivered quickly and either the analyst/programmer does not have online skills or the online environment is difficult to build in, then a batch program could be the right choice, otherwise an online program is usually more interesting and the results are easier to present to others.

Defining a New Database Table, Column or File

A new database table or indexed file needs to be defined to hold the SSA-NAME3 Keys and other data. The SSA-NAME3 Keys are 8 bytes in length (by default) and consists of ASCII printable character values.

The minimum number of columns or fields in this new table or file is two - the SSA-NAME3 Key, and a foreign key to point back to the master record where the name or address was sourced from.

It is recommended, however, that for performance optimization, this table should be de-normalized to include the search fields, the data to be used for matching, any data to be used for filtering, and any data to appear on the user's search results screen. For more information on this, see the *APPLICATION AND DATABASE DESIGN* manual.

CHAPTER 4

The Design Issues

This chapter looks at the various design issues that go into building an efficient and reliable search and matching system.

What Naming Data is used in Searches

In many systems, computerized or manual, we need to find things that have been filed away using a Person's name, a Customer's name, a Company name, a Place name, an Address, a File Title, an Author's name, a Book title, etc. . . All such names are collections of words, numbers and codes that have been used to "label" or "christen" the original real world item.

In the real world we use these names in speech and writing as the labels for "proper nouns" in sentences:

```
Geoff Holloway lives at 17 Congham Drive, Pymble NSW
Holloway, Geoffrey Norman is the name on loan # 1256347
The Data Clustering Engine V2.21 is used by XYZ Co.
```

In systems and databases we use such names to find files, transactions, accounts, and any variety of data recorded about the "entity" identified by the name or naming data.

Names are not normally unique. Names when said, written and especially when entered into computer systems are subject to considerable variation and error. You cannot avoid this variation and error. Even if the data on file is "perfect", the "search name" will come from the real world and be subject to natural error and variation.

What Identification Data do we Match with

In addition to the words and codes in Names, Addresses, Titles and Descriptions, we frequently use other data to make decisions about whether we believe two reports or records are about the same entity.

Search for	ROBERT J. JOHNSTON	12 RIVER SIDE SPRINGVALE	(807) 2334 657	1962/02/12
Yes	BOB JAMES JOHNSTON	SPRING VALE	2334 657	1962

Maybe	MR. R. J. JOHNSTONE	35 CITYVIEW CT. SPRINGVALE	1 807 4456 721	1962/12/02
No	ROBERT JOHNSON	2 MAPLE RD. BROOKFORD	555 763 2413	1973/10/04

Data such as dates of birth, dates of contract, ages, phone numbers and identity numbers are all subject to error and variation.

When a name is used to bring up candidates on a screen, people use all of the identification data returned to choose whether the records displayed are relevant or appropriate. In automated matching systems, the system itself has to be able to use the same data that people would use.

When people make choices about whether things match or not, they compensate for the error and variation. Our systems have to achieve the very same compensation that people make.

To confirm that records are in fact matching requires that our systems use the same data in the same manner as the human users of our systems would use. In fact our systems need to mimic our very best users doing the same job.

Objectives of Name Search and Matching Systems

Whether the process is an online inquiry like Customer Identification, or a Criminal Records search, or a batch matching process like merging Marketing Lists before a selection for mailing, we must find all the candidates that could possibly be the same as each other, or are the same as our "search data".

We must mimic a human expert in finding all the candidate records, and then make the same matching choices as the human expert would make for that specific business purpose.

This means that our searching and matching technology must overcome the natural error and variation that unavoidably occurs in all real world identification data. We must do this despite the fact that the process of capturing the real world data into computer systems actually introduces even more error and variation.

In many systems the objective is also to overcome fraudulent modification of identity data. This "class of error" is more aggressive in that it does not occur naturally, but is introduced to defeat or control aspects of matching systems while retaining the defense that it was in error rather than fraudulent.

Any attempt to overcome error and variation increases the work done and therefore the cost. We will also see that, in order to compensate for more error, we always run the risk of introducing false matches.

The task is a balancing or tuning exercise between:

- "Performance" and "Quality",
- "Under-matching" versus "Over-matching",
- "Missing the Right data" versus "Finding Wrong data".

File and Field Design Issues

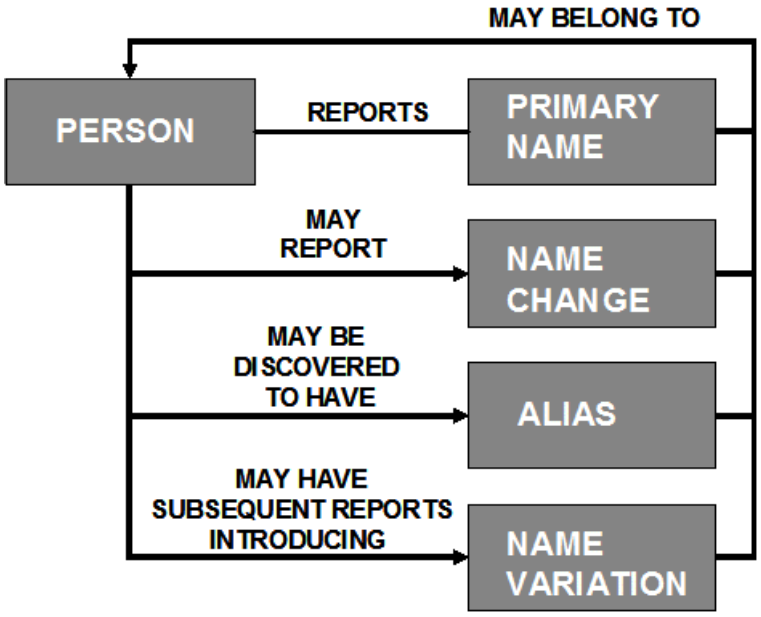
The design of file and field structures to support reliable name search and matching requires a good understanding of the nature of the data.

More Than One Name Field in a table or file, Names are truly "Many to Many"

It is obvious that two people or companies, or products, can have exactly the same name. It is also obvious that, even ignoring error and variation, people, places and things have more than one name:

- People have maiden names and married names;
- People have aliases and professional names;
- Companies have registered names, trading names and division names;
- Places have several addresses, on two separate streets, old addresses, billing addresses, postal addresses, etc.;
- People and places can have names in more than one language.

The relationship between a name and that which it names is quite naturally a true "many-to-many relationship".



It is not surprising that indexing these "many to many" relations requires careful design in the majority of today's relational databases, whose constructs are limited (with some good reason) to architectures based on "one to many" relations.

The design of a record or row that contains two fields, one for "name" and one for "maiden name", or "registered name" and "trading name", may make logical business sense, but it is not good for indexing.

When we are searching for a person name, company name or address we do not know which "role" it plays. We do not know if it is a birth name, married name or maiden name, we do not know if it is a current or prior address. In order to address this problem effectively, it is necessary to have several index entries pointing to the same record. The alternative of declaring a separate index on each field or attribute is totally prohibitive from a performance point of view.

Solving this "many to many" characteristic of names leads to an additional table or file in most databases. It therefore requires that this table is maintained in sync with the main business tables.

The Name Change Transaction

While it is arguably necessary that whenever you have a name field in a system, then there will be a "name change transaction", great care must be taken in deciding what to do about a name change.

In most cases the need to change a name will arise because a new transaction about the same person or company or product has been encountered. Another case is when the person has changed their name as a result of marriage, divorce, preference or simply discovered that he has it "wrong".

Usually removing the "old name" from the system is a bad idea; simply keep it as a known alias. References from "old documents" are very likely to create searches about "old names". Every name you encounter about a person, place or product is clearly evidence that rightly or wrongly that name is in use or has been in use in the real world about that same person. To maximize your ability to find or match this entity in the future, the strongest way to deal with name changes is to add an additional name to the index for the same entity. For business reasons it may be necessary to identify one name field as the preferred, current or registered name.

The Telephone Book as Metaphor for Name Search Index Design

In the telephone book, a search for the name Ann Jackson Smith would normally succeed, on the "Smith A" page.

Page 321		SMITH A
Smith A J	10 Main St Springvale	9257 5496

When the name being searched for is A J Smith or Ann Jackson Smith, the entry is found relatively easily by browsing through all of the Smith A J entries.

A search for A Smith or Ann Smith is slower because more names must be browsed. If the full name had been indexed, the search for Ann Smith would be faster and the search for Ann Jackson Smith even quicker.

Page 327		SMITH Alan
Smith Ann Jackson	10 Main St Springvale	9257 5496

Though this increases the size of each entry and the cost of capturing the information, the overall performance of searches is improved when there is more data in the name. Given a full name to search with, its entry can be found more quickly.

In addition, when the name being searched for has missing or extra words or words in a different order, the simple telephone book indexing system starts to break down.

Searches for Ann Jackson-Smith, Ann Smith Jackson or Smith Ann will fail unless the searcher, after failing on the "J" and "A" pages, permutes the words and looks on the "S" page.

Regardless, a search for Ann Jackson will never succeed if the entry in the book was Smith, J.A. or Smith, Ann Jackson.

If, however, the name Ann Jackson Smith was indexed on three pages of the telephone book, on an "Ann", "Jackson" and "Smith" page, by permuting the order of the words, then any of the above searches would succeed by opening one page.

Page 17		ANN Smith B
Ann, Smith Jackson	10 Main St Springvale	9257 5496

Page 119		JACKSON Ann K
Jackson, Ann Smith	10 Main St Springvale	9257 5496

Page 327		SMITH Alan
Smith Ann Jackson	10 Main St Springvale	9257 5496

The size of the telephone book increases, but search cost does not. The extra "index entries" increases the physical size, yet improves overall quality and performance because any search succeeds.

In computer databases, with today's low data storage costs, regardless of the volume of the file, the right solution for name indexes is permutation of words in the index entries at update time. And storing multiple records on separate "pages" in the database just like our example in the telephone book above. Permutation of naming words at search time alone can not guarantee to overcome the missing word, extra word or gross single word errors. This is not a design problem that can be overcome with better design, it is a mathematical constraint.

File Design for Optimal Name Search Performance

A search for all records that are relevant candidates for one set of search data, requires that one must display a list of good candidates on a screen or present this list to a batch matching/selection program.

To achieve this, the search data will be computed and used to Find, Read or Select a range of candidates from the database. This may be one or more logical requests to the database (for example, several "select" or "find" statements may be necessary).

The database size affects the average number of candidates in a given range. The bigger the file the more candidates are on file.

DBSize	Ellen Dodds	John Smiths
100,000	4	50
1,000,000	40	500
10,000,000	400	5,000

Searches are usually distributed the same way - if John Smith is .05% of the file, it's also .05% of the transactions.

The online name search transactions logically require:

- computation to build search ranges based on the data used in the keys;
- physical access to the database to get index entries;
- physical I/O to retrieve the display and matching data for all candidates;
- computation to eliminate, rank, or sort before display.

The time consuming work is the physical I/O:

- One or more physical I/O per index entry per logical database command;
- One or more physical I/O per block of candidate data records;
- If "joins" are necessary to get complete data for Matching & Display, more than one physical I/O will occur per data record.

The only way more than one candidate can be in a physical block is if the database file or table is ordered in the name key sequence. Even if this is true, little advantage is gained if access or "joins" to other tables are necessary to complete the display of a candidate line. Unless the tables are small enough to totally fit into memory, to achieve acceptable response time, all display or matching data for a candidate must be in the same record and candidates must be in physically adjacent records.

Achieving acceptable response time for even a single screen of candidates can not be done if each line requires multiple physical I/Os. You can reduce the number of candidates or screens by automating the choice, selection or matching process, but the data still has to be read from the database and presented to a "matching" program, so the need for physical optimization is still very necessary.

Of course the average number of candidate records read should be kept to a minimum, but this minimum will relate to the size of file, how common the name is, and to what degree it is important not to miss possible matches. This decision should be tied to individual transaction and business risk/benefit.

To get good response time in name search, de-normalizing & maintaining a copy of the relevant name search & matching data in optimum physical sequence is essential. It is the only way to avoid "joins" and extra physical I/O.

Coping with a Small % of Foreign Name and Address Data

With today's electronic communications, WEB based sales & marketing, and global business environment, it is inevitable that some prospects and customers in a local or regional file will have addresses from other countries. The percentage of this data in your files may be small but it is growing, especially in prospect files that are purchased or rented.

A common problem in coping with such data is thinking that rigid local standards can be made to work for this foreign data.

Asking the input data to be formatted into detailed fields according to strict local rules is inviting assumptions and choices which can vary from person to person, country to country. This leads to country name in state fields or postal code fields, apparently invalid postal codes, postal codes in address line fields, etc.

Requesting input in unformatted or loosely formatted fields is the best way of obtaining reliability and completeness. If transaction and file formats for names and addresses are designed like the lines on an envelope you will be able to capture both local and foreign data with complete integrity. This will mean that

the search and matching system should be designed to cope with unformatted data. Systems can be reliable in dealing with unformatted data, people are not reliable when they are asked to format data.

This approach is essential for multinational systems but also very relevant for maximum value in local systems.

Don't try to overcome these problems before the data is stored. Let the system overcome them. Use simple large fields for names and addresses that allow users to input data as they would on an envelope or business card.

When Partitioning Keys Makes Sense

It is a misconception that partitioning search keys improves the reliability of a name search. Partitioning will always result in some loss of reliability. However, all name search systems are susceptible to a conflict between performance and reliability. When extreme volumes of data are to be searched, and performance is more critical than reliability, there is a case for partitioning the keys.

The choice of what data to partition with also creates a conflict between quality and performance. An attribute which achieves the performance objective, but is not measurably reliable, is not helpful. An attribute which is measurably reliable, but does not meet the performance objective, is also not helpful.

For some systems a year of birth may be a good partition, but no good if the error rate in birth dates is high. For other systems a state code may be a good partition, but no good if there is a high rate of movement between states, or a lack of truth in the state codes.

The need for partitioning should be empirically derived (as a result of tests on real data, in real volumes, in a production-like environment) and not decided upon theoretically.

If partitioning is used, when null or suspicious values of the partitioning attribute are encountered, these must be added to a common partition which is searched whenever a specific partition is searched. Also when nulls or errors are found in the search data's partitioning attribute then all partitions must be searched.

A strong search system will allow searches across all partitions, even if this is not the default search.

Storing the Good with the Bad

In many business and government systems it is necessary to index data about both the "good guys" and the "bad guys":

- Customers, rather than ex Customers who have Bad Debts or for whom Service is Denied;
- Prospects, rather than Do Not Mail names;
- People being protected, rather than the Terrorists and Trouble Makers;
- Persons with Petty Criminal Records, rather than Dangerous Criminals.

While the data stored may be identical, this is not a good reason for storing the information in the same file. If they are stored together and indexed together it is easy to miss a critical "bad guy".

In many system designs, a central name index, or personality file is created, with one common Name Search dialogue built for it. Then simply because it exists and contains names, addresses, account numbers, and other identity data together with system references, all forms of data are stored in this one "cross reference" index.

For both system performance and quality, and to allow user dialogues to be more efficient and effective, the records about negative or risk related information should be indexed separately using more exhaustive and expensive techniques for the negative data. Certainly the commonality of the process and formats can be taken advantage of by sharing code and inheriting designs, but mixing the good with the bad is never a strong design.

In order to maximize the chance of finding the high risk "bad guys" keep them in separate files, index them more exhaustively, and use wider search strategies.

CHAPTER 5

Standard Population Choices

This chapter includes the following topics:

- Overview, 23
- Standard Populations, 24
- A Primer on Keys and Search Strategies, 24
- Key Fields, 25
- Key Levels, 27
- Search Levels, 27
- Match Purposes, 29
- Match Levels, 40
- Managing Population Rule Sets, 41
- Effect of File Size on Name Search Performance, 41
- Impact of Risk on the Search Transaction, 42
- The Critical Exhaustive Search, 42
- Balancing Missed Matches with Finding Too Much, 43
- Undermatching or Overmatching, 44
- Discovering the Missed Matches, 44
- The Importance of Prototyping with Production Data, 45
- Overview, 45
- Field Design for Multinational Systems, 46
- Deployment of Multinational Systems, 47
- Code Pages, Character Sets and other Encoding Issues, 47
- Unicode Issues, 48
- Transliteration Realities, 49
- Transliteration and Data Matching, 49

Overview

This section is designed to help the analyst, designer or developer make the right choices when choosing the Standard Population and the search and match controls, levels and data to use in the search application.

Standard Populations

SSA-NAME3 is delivered with over 60 Standard Populations covering different countries, languages and regions. As new Standard Populations are added regularly, the most current list is that which is shown by the Informatica IR Product Installer.

Before installing SSA-NAME3, an analysis should be done of the data that is to be searched and matched. Which country(ies) is it from? What codepage is it in? Does it contain mixed scripts?

When installing SSA-NAME3, choose the Standard Population(s) that suit the data you will be searching and matching. An Informatica Corporation consultant can be contacted for assistance with the decision. In many cases the decision will be simple (e.g. a USA company doing business in the USA alone would choose the USA Standard Population).

Note: All standard populations currently supported by Informatica Corporation are delivered with the SSA-NAME3 install. However, some require a separate license to use.

If you have selected a Population during the install process that requires a separate license, a license warning screen will be shown prompting verification that the license is held.

Currently, the Standard Populations requiring a separate license are:

- The Chinese, Japanese and Korean double-byte populations;
- The Arabic Mixed population (supporting bi-directional Arabic / Latin searching and matching)

A Primer on Keys and Search Strategies

The safest way of finding a name match in a database is to first perform a search on an index built from name alone, thus building a candidate list of possible matches, and then to refine, rank or select the matches in that candidate list based on other identification data.

Name only keys are built from one or more parts of the name field (words & words, words & initials). Of course the method used for constructing the database keys must match the method used for constructing the search keys.

The more name parts used in the key, and the greater the number of keys built per name, the greater the variety of search strategies which can be supported.

A name key for "ANN JACKSON-SMITH" built from family name plus first initial, "SMITH A", can support search strategies using the family name word and initial and also using only the single family word. A name key built from family name and first name, "SMITH ANN" can support search strategies using two words from the name (at the "two word level" or wider). The fewer words used in the key the larger or wider the set of responses will be.

An extra name key, say "JACKSON ANN", supports a search where the search name is missing a certain part or the parts are in a certain different order.

The choice of keys and search strategies together defines the width or depth of the search (by the number of name parts used in the search keys) and the degree of sequence variations and missing parts overcome (by the number of different keys).

The greater the number of name parts used in a search key, the fewer candidates on average will be returned, and the quicker the search. A search strategy which uses the full name makes sense when the name is expected to be generally reliable, when the match is expected to be in the database, or when the search will

be stopped, or at least interrupted, at the first match. This type of search strategy is thought of as a Typical search and is used to find data that is expected to be on file.

As confidence in the quality of the search or database names declines, or as the risk of missing a match increases, so will the need for a different search strategy arise. A high-risk search, or a search using poor quality data, should use a wider search strategy to compensate for severe spelling errors and more sequence variations, missing and extra words in the names. This type of search strategy is thought of as an Exhaustive search and is frequently used to prove that data is not on file.

In large scale systems the choice and sophistication of the search strategy is consequential to both performance demands, risk of missing critical data, need to avoid duplication of data and the volume of data under indexing.

The choice of search strategy should match the business needs of the search. The search strategy used for one set of data or one system may be very different from that used in another.

A search strategy is affected by decisions on the following Standard Population components:

- Key Field - the field to use for indexing and search
- Key Level - the type of keys built
- Search Level - the breadth of search performed

Matching, filtering and ranking of the candidates returned from a search is affected by decisions on the following Standard Population components:

- Match Purpose - the fields used in Matching and the business purpose of the Match
- Match Level - the degree of Match chosen

Key Fields

Using Standard Populations, an application may be set up to index and search on three field types:

- Person Names
- Organization Names
- Addresses

Person Names

The Algorithm that builds keys and search ranges for Person Names is invoked by a calling SSANAME3 by passing `FIELD=Person_Name` in the Controls parameter of the "get keys" or "get ranges" calls.

The `Person_Name` Algorithm is designed to overcome the error and variation that would be typically found in a person's full name. This may include salutations and honorifics, special characters, embedded spaces, nicknames, different word orders, use of initials, spelling errors, concatenated words, localized words, foreign words, etc.

An application should pass the full person name to SSA-NAME3 functions. The word order, i.e. the position of the first name, middle names and family names, should be the normal word order used in your data population. For example, in English speaking countries, the normal word order would be:

```
First Name + Middle Name(s) + Family Name(s)
```

Depending on your table design, your application may have to concatenate these separate fields into one field before calling SSA-NAME3.

While SSA-NAME3 includes Search Strategies that overcome word order variations, the word order does have some significance in the quality of Narrow and Typical searches, and when matching using the Purposes "same Household", "same Family" or "same Wide_Household".

The application (or SSA-NAME3) may pass multiple names (such as a married name and a former name) in the one call to SSA-NAME3.

The `Person_Name` algorithm has an Edit-List whose rules may be overridden by the Population Override Manager or Edit RuleWizard.

Organization Names

The Algorithm that builds keys and search ranges for Organization Names is invoked by a calling application by passing `FIELD=Organization_Name` or `FIELD=Organisation_Name` in the SSANAME3 Controls parameter of the "get keys" or "get ranges" calls.

The `Organization_Name` Algorithm is designed to overcome the error and variation that would be typically found in a business, company, institution or other organization name. The Algorithm also caters for multiple names in the one field, and a mixture of Organization and Person names in the data population. The error and variation may include different legal endings, abbreviations, salutations and honorifics, special characters, embedded spaces, nicknames, different word orders, missing and extra words, spelling errors, concatenated words, use of initials, mixed use of numbers and words, foreign words, localization, etc.

This field supports matching on a single name, or a compound name (such as a legal name and its trading style).

The application (or SSA-NAME3) may also pass multiple names (such as a current name and a former name) in the one call to SSA-NAME3.

The `Organization_Name` algorithm has an Edit-List whose rules may be overridden by the Population Override Manager or Edit Rule Wizard.

Addresses

The Algorithm that builds keys and search ranges for Addresses is invoked by a calling application by passing `FIELD=Address_Part1` in the SSA-NAME3 Controls parameter of the "get keys" or "get ranges" calls.

The `Address_Part1` Algorithm is designed to overcome the error and variation that would be typically found in addresses. The error and variation may include the presence of care of information, abbreviations, special characters, embedded spaces, different word orders, spelling errors, concatenated words and numbers, use of initials, mixed use of numbers and words, foreign words, missing words, extra words and sequence variations, etc.

An application should pass the part of address up to, but not including, the locality "last line". The word order, i.e. the position of the address components, should be the normal word order used in your data population. These should be passed in one field. Depending on your table design, your application may need to concatenate these attributes into one field before calling SSA-NAME3.

For example, in the US, a typical string to pass would comprise of:

```
Care-of + Building Name + Street Number + Street Name + Street Type + Apartment Details
```

But not including City, State, Zip, Country.

The application (or SSA-NAME3) may pass multiple addresses (such as a residential address and a postal address) in the one call to SSA-NAME3. See the *API REFERENCE* manual for more details.

The `Address_Part1` algorithm has an Edit-List whose rules may be overridden by the Population Override Manager or Edit Rule Wizard.

Key Levels

Using Standard Populations, a user's database may be indexed on Person Names, Organization Names and Addresses using one of three Key Levels:

- Standard
- Extended
- Limited

The choice of Key Level is passed to the SSA-NAME3 "get keys" function directly by the user's application.

Standard

Standard is the recommended Key Level for typical applications. Its use overcomes most variations in word order, missing words and extra words.

It also maximizes the likelihood of finding candidates in cases of severe spelling error in multi-word names.

Standard is the default if no Key Level is specified.

Standard Keys or Extended Keys should be implemented if the Edit Rule Wizard is being used.

Extended

For high-risk or critical search applications, SSA-NAME3 can generate "Extended" Keys. Extended Keys extend Standard Keys by adding more keys based on token concatenation. The designer/developer should be aware that the use of Extended Keys will increase disk space requirements and result in larger candidate sets at search time. However, the intended use of Extended Keys is to improve reliability by finding matches regardless of word order variation and concatenation.

Standard Keys or Extended Keys should be implemented if the Edit Rule Wizard is being used.

Limited

If disk space is limited, SSA-NAME3 can generate "Limited" Keys. Limited Keys are a subset of Standard Keys. The designer/developer should be aware that the use of Limited Keys, while saving on disk space, may also reduce search reliability.

Search Levels

Using Standard Populations, an application may be set up to search on Person Names, Organization Names and Addresses using four different Search Levels:

- Typical;
- Exhaustive;
- Narrow;
- Extreme.

The choice of Search Level is passed to the SSA-NAME3 "get ranges" function directly by the user's application.

It is good practice to test using different Search Levels on real production data and volumes to measure both the response time and the reliability differences.

Typical

A Typical search level for most applications will provide a practical balance between quality and response time. It should be used in typical online or batch transaction searches. It is the default if no Search Level is specified.

For `Person_Name` searches, it is designed to find common, but not extreme, error and variation including cases where initials are present instead of full given names and where the initial of a name has changed due to the internal rules applied.

For `Organization_Name` searches, it is designed to find common, but not extreme, error and variation including instances of word concatenation.

For `Address_Part1` searches, it is designed to find common, but not extreme, error and variation.

Exhaustive

An Exhaustive search level is provided for applications that have an increased risk associated with missing a match, where data quality is a concern or where data volumes are low enough to make it the default search. It increases the number of candidates returned and consequently response times may be extended. An Exhaustive search will occasionally find matches that a Typical search misses, however, these will generally be where there is more extreme error and variation.

For `Person_Name` searches, it is designed to find more error and variation than a Typical search, especially where there is extreme spelling error in the family or middle names.

For `Organization_Name` searches, it is designed to find more error and variation than a Typical search, especially where there is extreme spelling error in the major word or trailing words.

For `Address_Part1` searches, it is designed to find more error and variation than a Typical search, especially where there are more cases of missing words, extra words or sequence differences.

Narrow

A Narrow search level compromises on completeness of search in favor of faster and more direct answers. It may be an option in search applications that do not have a high risk associated with missing a match, require very tight levels of matching, or where data volumes are extreme and response time is a critical factor.

For `Person_Name` searches, it is designed to find the very common error and variation including cases where initials are present instead of full given names.

For `Organization_Name` searches, it is designed to find the very common error and variation and primarily where the words are in a stable order.

For `Address_Part1` searches, it is designed to find the very common error and variation and primarily where the tokens are in a stable order.

Extreme

An Extreme search level uses every possibility to discover a candidate match; consequently response times may be extended. It is provided for applications that have a critical need to find a match if one is present in the database, despite the error and variation.

An Extreme search may only occasionally find matches that an Exhaustive search misses, however, because the risk is very high, every possible match is deemed important.

The types of candidates returned for all Field types is the same when using an Extreme search. Extreme spelling error is picked up in names or addresses with two or more words or tokens.

Match Purposes

SSA-NAME3's Matching services are used by applications, such as Informatica IR, MDM Registry- Edition & DCE, to filter, rank or match the candidate records returned from a search. The identity data from the search is compared to the identity data from the candidate record, and a score or a ruling is returned. Pre-built Matching algorithms are provided to address today's common business purposes. These are called "Match Purposes". In combination with the Match Purpose, a selectable Match Level determines the tightness or looseness of the match. The application may also override the Score threshold, which determines the match ruling returned.

SSA-NAME3 Matching is designed to compensate for the error and variation in identity data. The matching logic is comprised of heuristic algorithms that are optimized for each class of data (e.g.: name, organization, address, dates, codes). The algorithms include numerous rules and switches to handle initials, aliases, common variations, prefixes, suffixes, transpositions and word order.

Additionally, all Match Purposes use string cleaning routines, Edit-Lists, different matching Methods for different data types, optimized Matching options, field and token level weighting and phonetic/ orthographic stabilization.

Each Match Purpose supports a combination of mandatory and optional fields and each field is weighted according to its influence in the match decision. Some fields in some Purposes may be "grouped". Two types of grouping exist:

- A "Required" group requires at least one of the field members to be non-null;
- A "Best of" group will contribute only the best score from the fields in the group to the overall match score.

For example, in the "Individual" Match Purpose:

- `Person_Name` is a mandatory field.
- One of either ID Number or Date of Birth is required.
- Other attributes are optional.

The overall score returned by each Purpose is calculated by adding the participating field scores multiplied by their respective weight and divided by the total of all field weights. If a field is optional and is not provided, it is not included in the weight calculation.

The weights and matching options used in the Standard Populations are internally set by Informatica's Population experts based on years of tuning experience. They are not available to be overridden by the application. However, if a user has a different need not supported by the Standard Population, Informatica Corporation may offer to build a Custom Population for that client.

Field Types

Below are descriptions of the fields supported by the various Match Purposes, provided in alphabetical order.

Field	Description
Address_Part1	Typically includes that part of address up to, but not including, the locality "last line". The word order, i.e. the position of the address components, should be the normal word order used in your data population. These should be passed in one field. Depending on table design, your application may need to concatenate these attributes into one field before calling SSA-NAME3. For example, in the US, a typical string to pass would comprise of: `Care-of + Building Name + Street Number + Street Name + Street Type + Apartment Details` Matching on `Address_Part1` uses methods and options designed specifically for addresses. It has its own Edit-List whose rules can be overridden by the Population Override Manager or Edit RuleWizard. It is also possible to supply the entire address in the `Address_Part1` field for matching. The application may pass multiple addresses (such as a residential address and a postal address) in the one call to SSA-NAME3. Refer to the *API REFERENCE* manual for more details. See the *Key Fields* section for more details on `Address_Part1`.
Address_Part2	Typically includes the "locality" line in an address. For example, in the US, a typical string to pass would comprise of: `City + State + Zip (+ Country)` Matching on `Address_Part2` uses methods and options designed specifically for addresses. It uses the same Edit-List as `Address_Part1`. The rules in this Edit-List can be overridden by the Population Override Manager or Edit RuleWizard.
Attribute1, Attribute2	Attribute 1 and Attribute 2 are two general purpose fields. They are matched using a general purpose string matching algorithm that compensates for transpositions and missing characters or digits.
Date	The `Date` field is used for matching any type of date (e.g. date of birth, expiry date, date of contract, date of change, creation date, etc). It expects the date to be passed in `Day+Month+Year` order. It supports the use or absence of delimiters between the date components. Matching on dates uses methods and options designed specifically for dates. It overcomes the typical error and variation found in this data type.
ID	The ID field is used for matching any type of ID number (e.g. Account number, Customer number, Credit Card number, Drivers License number, Passport, Policy number, SSN or other identity code, VIN, etc). It uses a string matching algorithm that compensates for transpositions and missing characters or digits. It also has its own Edit-List whose rules can be overridden by the Population Override Manager or Edit RuleWizard.
Organization_ Name	Used to match the names of organizations. These could be company names, business names, institution names, department names, agency names, trading names, etc. This field supports matching on a single name, or a compound name (such as a legal name and its trading style). It has its own Edit-List whose rules can be overridden by the Population Override Manager or Edit RuleWizard. The application may also pass multiple names (e.g. a legal name and a trading style) in the one call to SSA-NAME3. Refer to the *API REFERENCE* manual for more details. See the *Key Fields* section for more details on `Organization_Name`.

Field	Description
Person_Name	Used to match the names of people. An application should pass the full person name. The word order, i.e. the position of the first name, middle names and family names, should be the normal word order used in your data population. For example, in English speaking countries, the normal word order would be: `First Name + Middle Name(s) + Family Name(s)` Depending on table design, your application may have to concatenate these separate fields into one field before calling SSA-NAME3. This field supports matching on a single name, or an account name (such as JOHN & MARY SMITH). The application may also pass multiple names (e.g. a married name and a former name) in the one call to SSA-NAME3. Refer to the *API REFERENCE* manual for more details. It has its own Edit-List whose rules can be overridden by the Population Override Manager or Edit RuleWizard. See the *Key Fields* section for more details on `Person_Name`.
Postal_Area	The `Postal_Area` field can be used to place more emphasis on the postal code than if it were included in the `Address_Part2` field. It is used for all types of postal codes, including Zip codes. It uses a string matching algorithm that compensates for transpositions and missing characters or digits. It also has its own Edit-List whose rules can be overridden by the Population Override Manager or Edit RuleWizard.
Telephone_Number	The `Telephone_Number` field is used to match telephone numbers. It uses a string matching algorithm that compensates for transpositions and missing digits or area codes. It also has its own Edit-List whose rules can be overridden by the Population Override Manager or Edit RuleWizard.

Purposes Types

Below are descriptions of the Purposes supported by the Standard Populations, provided in alphabetical order.

Address

This Purpose is designed to identify an address match. The address might be postal, residential, delivery, descriptive, formal or informal.

This Match purpose is typically used after a search by `Address_Part1`.

Field	Required?
Address_Part1	Yes
Address_Part2	No
Postal_Area	No
Telephone_Number	No
ID	No
Date	No

Field	Required?
Attribute1	No
Attribute2	No

The only required field is `Address_Part1`. The fields `Address_Part2`, `Postal_Area`, `Telephone_Number`, `ID`, `Date`, `Attribute1` and `Attribute2` are available as optional input fields to further differentiate an address. For example if the name of a City and/or State is provided as `Address_Part2`, it will help differentiate between a common street address [100 Main Street] in different locations.

To achieve a "best of" score between `Address_Part2` and `Postal_Area`, pass `Postal_Area` as a repeat value in the `Address_Part2` field. For example:

```
*Address_Part2*100 Main St*Address_Part2*06870***
```

In this case, the `Address_Part2` score used will be the higher of the two scored fields.

Contact

This Purpose is designed to identify a contact within an organization at a specific location.

This Match purpose is typically used after a search by `Person_Name`. However, either `Organization_Name` or `Address_Part1` could be used as the search criteria.

For ultimate quality, a tiered search using two or all three of these fields could be used in the search. (A tiered search is for example, a `Person_Name` search followed by an `Address_Part1` search).

Field	Required?
Person_Name	Yes
Organization_Name	Yes
Address_Part1	Yes
Address_Part2	No
Postal_Area	No
Telephone_Number	No
ID	No
Date	No
Attribute1	No
Attribute2	No

The required fields are `Person_Name`, `Organization_Name`, and `Address_Part1`. This is designed to successfully match person X at company Y and address Z.

To further qualify a match, the fields `Address_Part2`, `Postal_Area`, `Telephone_Number`, `ID`, `Date`, `Attribute1` and `Attribute2` may be optionally provided.

To achieve a "best of" score between `Address_Part2` and `Postal_Area`, pass `Postal_Area` as a repeat value in the `Address_Part2` field. For example:

```
*Address_Part2*100 Main St*Address_Part2*06870***
```

In this case, the `Address_Part2` score used will be the higher of the two scored fields.

Corporate Entity

The `Corporate Entity` Purpose is designed to identify an Organization by its legal corporate name, including the legal endings such as `INC, LTD`, etc. It is designed for applications that need to honor the differences between such names as ABC TRADING INC and ABC TRADING LTD.

This Match purpose is typically used after a search by `Organization_Name`.

Field	Required?
Organization_Name	Yes
Address_Part1	No
Address_Part2	No
Postal_Area	No
Telephone_Number	No
ID	No
Attribute1	No
Attribute2	No

It is in essence the same purpose as Organization, except that tighter matching is performed and legal endings are not treated as noise.

To achieve a "best of" score between `Address_Part2` and `Postal_Area`, pass `Postal_Area` as a repeat value in the `Address_Part2` field. For example:

```
*Address_Part2*100 Main St*Address_Part2*06870***
```

In this case, the `Address_Part2` score used will be the higher of the two scored fields.

Division

The `Division` Purpose is designed to identify an Organization at an Address. It is typically used after a search by `Organization_Name` or by `Address_Part1`, or both.

Field	Required?
Organization_Name	Yes
Address_Part1	Yes
Address_Part2	No
Postal_Area	No

Field	Required?
Telephone_Number	No
ID	No
Attribute1	No
Attribute2	No

It is in essence the same purpose as Organization, except that Address_Part1 is a required field. Thus, this Purpose is designed to match company X at an address of Y (or Z, etc, if multiple addresses are supplied).

To achieve a "best of" score between `Address_Part2` and `Postal_Area`, pass `Postal_Area` as a repeat value in the `Address_Part2` field. For example:

```
*Address_Part2*100 Main St*Address_Part2*06870***
```

In this case, the `Address_Part2` score used will be the higher of the two scored fields.

Family

The Family purpose is designed to identify matches where individuals with the same or similar family names share the same address or the same telephone number.

This purpose is typically used after a tiered search (multi-search) by `Address_Part1` and `Telephone_Number`.

Note: It is not practical to search by `Person_Name` because ultimately only one word from the `Person_Name` needs to match, and a one-word search will not perform well in most situations.

Field	Required?
Person_Name	Yes
Address_Part1	Yes
Telephone_Number	Yes
Address_Part2	No
Postal_Area	No
Attribute1	No
Attribute2	No

Note: Score will be based be on best of the group specified in the above table.

Emphasis is placed on the Last Name, or "Major Word" of the `Person_Name` field, so this is one of the few cases where word order is important in the way the records are passed to SSA-NAME3 for matching.

However, a reasonable score will be generated provided that a match occurs between the major word in one name and any other word in the other name.

Required fields are `Person_Name`, `Address_Part1` and `Telephone_Number`. Optional qualifying fields are `Address_Part2`, `Postal_Area`, `Attribute1`, and `Attribute2`.

To achieve a "best of" score between `Address_Part2` and `Postal_Area`, pass `Postal_Area` as a repeat value in the `Address_Part2` field. For example:

```
*Address_Part2*100 Main St*Address_Part2*06870***
```

In this case, the `Address_Part2` score used will be the higher of the two scored fields.

Fields

This Purpose is provided for general non-specific use. It is designed in such a way that there are no required fields. All field types are available as optional input fields.

Field	Required?
Person_Name	No
Organization_Name	No
Address_Part1	No
Address_Part2	No
Postal_Area	No
Telephone_Number	No
ID	No
Date	No
Attribute1	No
Attribute2	No

One way this Purpose could be used is as a non-exact match filter before applying some other Match Purpose. (For exact match filters, use the `Filter` Purpose). For example, before passing a record to the `Division` Purpose, use the `Fields` Purpose to eliminate any company with `ID` numbers which do not score above 80%. To do this, the application would first pass the ID numbers to SSA-NAME3 for matching using `PURPOSE=FIELDS`, and then decide based on the score returned whether to pass the full records for matching by the `Division` Purpose.

Filter1-9

The `Filter` Purpose is provided so that the application can perform exact match filtering based on the setting of one or more flags in the records. One call to `ssan3_match` can use up to nine Filters (`Filter1-9`).

Field	Required?
Filter1-9	Yes

For example, say an index supported searching and matching across two types of names: Company names (identified by a Name-Type-Flag of "C"), and Person names (identified by a Name-Type-Flag of "P"). A search application may need to support searches across both name types, as well as within each name type. To support the "within each name type" search, the application can use the Filter Purpose to filter out exact matches based on the name type flag.

The fields `Filter1-9` can be any code or flag.

For non-exact filtering, use the `Fields` Purpose.

Household

The Household purpose is designed to identify matches where individuals with the same or similar family names share the same address.

This purpose is typically used after a search by `Address_Part1`.

Note: It is not practical to search by `Person_Name` because ultimately only one word from the `Person_Name` needs to match, and a one-word search will not perform well in most situations.

Field	Required?
Person_Name	Yes
Address_Part1	Yes
Address_Part2	No
Postal_Area	No
Telephone_Number	No
Attribute1	No
Attribute2	No

Emphasis is placed on the Last Name, or "Major Word" of the `Person_Name` field, so this is one of the few cases where word order is important in the way the records are passed to SSA-NAME3 for matching.

However, a reasonable score will be generated provided that a match occurs between the major word in one name and any other word in the other name.

Required fields are `Person_Name` and `Address_Part1`. Optional qualifying fields are `Address_Part2`, `Postal_Area`, `Telephone_Number`, `Attribute1`, and `Attribute2`.

To achieve a "best of" score between `Address_Part2` and `Postal_Area`, pass `Postal_Area` as a repeat value in the `Address_Part2` field. For example:

```
*Address_Part2*100 Main St*Address_Part2*06870***
```

In this case, the `Address_Part2` score used will be the higher of the two scored fields.

Individual

This Purpose is designed to identify a specific individual by name and with either the same ID number or Date of Birth attributes.

It is typically used after a search by `Person_Name`.

Field	Required?
Person_Name	Yes
ID	At least one of these two

Field	Required?
Date	
Attribute1	No
Attribute2	No

The required fields are `Person_Name`, and one of either `ID` and `Date`.

The fields `Attribute1` and `Attribute2` may be optionally provided to further qualify the match.

Organization

The Organization Purpose is designed to match organizations primarily by name. It is targeted at online searches when a name only lookup is required and a human is available to make the choice. Matching in batch would typically require other attributes in addition to name to make match decisions.

Field	Required?
Organization_Name	Yes
Address_Part1	No
Address_Part2	No
Postal_Area	No
Telephone_Number	No
ID	No
Date	No
Attribute1	No
Attribute2	No

The only required field is `Organization_Name`. The fields `Address_Part1`, `Address_Part2`, `Postal_Area`, `Telephone_Number`, `ID`, `Date`, `Attribute1` and `Attribute2` may are also provided as optional input fields to refine the ranking.

To achieve a "best of" score between `Address_Part2` and `Postal_Area`, pass `Postal_Area` as a repeat value in the `Address_Part2` field. For example:

```
*Address_Part2*100 Main St*Address_Part2*06870***
```

In this case, the `Address_Part2` score used will be the higher of the two scored fields.

Person_Name

This Purpose is designed to identify a Person by name. It is targeted at online searches when a name only lookup is required and a human is available to make the choice. Matching in batch would typically require other attributes in addition to name to make match decisions.

Field	Required?
Person_Name	Yes
Address_Part1	No
Address_Part2	No
Postal_Area	No
Telephone_Number	No
ID	No
Date	No
Attribute1	No
Attribute2	No

The only required field is `Person_Name`. The optional fields available for this purpose are `Address_Part1`, `Address_Part2`, `Postal_Area`, `Telephone_Number`, `ID`, `Date`, `Attribute1`, and `Attribute2`.

To achieve a "best of" score between `Address_Part2` and `Postal_Area`, pass `Postal_Area` as a repeat value in the `Address_Part2` field. For example:

```
*Address_Part2*100 Main St*Address_Part2*06870***
```

In this case, the `Address_Part2` score used will be the higher of the two scored fields.

Resident

The Resident Purpose is designed to identify a person at an address.

This purpose is typically used after a search by either `Person_Name` or `Address_Part1`, or both in a multi-search.

Field	Required?
Person_Name	Yes
Address_Part1	Yes
Address_Part2	No
Postal_Area	No
Telephone_Number	No
ID	No

Field	Required?
Date	No
Attribute1	No
Attribute2	No

The required fields are `Person_Name` and `Address_Part1`. The fields `Address_Part2`, `Postal_Area`, `Telephone_Number`, `ID`, `Date`, `Attribute1` and `Attribute2` are optional input fields to help qualify or rank a match if more information is available.

To achieve a "best of" score between `Address_Part2` and `Postal_Area`, pass `Postal_Area` as a repeat value in the `Address_Part2` field. For example:

```
*Address_Part2*100 Main St*Address_Part2*06870***
```

In this case, the `Address_Part2` score used will be the higher of the two scored fields.

Wide_Contact

This Purpose is designed to loosely identify a contact within an organization - that is without regard to actual location.

It is typically used after a search by `Person_Name`, however, a second search by `Organization_Name` could be used to get better quality.

Field	Required?
Person_Name	Yes
Organization_name	Yes
ID	No
Attribute1	No
Attribute2	No

The fields required for this Purpose are `Person_Name` and `Organization_Name`. This is designed to successfully match a person X at company Y.

In addition to the required fields, `ID`, `Attribute1` and `Attribute2` may be optionally provided for matching to further qualify a contact.

Wide_Household

The `Wide_Household` purpose is designed to identify matches where the same address is shared by individuals with the same family name or with the same telephone number.

This purpose is typically used after a search by `Address_Part1`.

Note: It is not practical to search by `Person_Name` because ultimately only one word from the `Person_Name` needs to match, and a one-word search will not perform well in most situations.

Field	Required?
Address_Part1	Yes
Person_Name	Yes
Telephone_Number	Yes
Address_Part2	No
Postal_Area	No
Attribute1	No
Attribute2	No

Note: This score will be based on best of the group in the above table.

Emphasis is placed on the Last Name, or "Major Word" of the `Person_Name` field, so this is one of the few cases where word order is important in the way the records are passed to SSA-NAME3 for matching.

However, a reasonable score will be generated provided that a match occurs between the major word in one name and any other word in the other name.

Required fields are `Person_Name`, `Address_Part1` and `Telephone_Number`. Optional qualifying fields are `Address_Part2`, `Postal_Area`, `Attribute1` and `Attribute2`.

To achieve a "best of" score between `Address_Part2` and `Postal_Area`, pass `Postal_Area` as a repeat value in the `Address_Part2` field. For example:

```
*Address_Part2*100 Main St*Address_Part2*06870***
```

In this case, the `Address_Part2` score used will be the higher of the two scored fields.

Match Levels

Using Standard Populations, an application may be set up to match on any of the defined Match Purposes using one of three different Match Levels:

- Typical;
- Conservative;
- Loose.

The choice of Match Level is passed to the SSA-NAME3 "match" function directly by the user's application.

It is good practice to test using different Match Levels on real production data and volumes to measure the reliability differences.

Typical

A Typical match level for most applications delivers "reasonable" matches. It should be used in typical online or batch transaction searches. It is the default if no Match Level is specified.

Conservative

A Conservative match level for most applications delivers "close" matches. It is generally used in batch systems where accuracy of match is paramount.

Loose

A Loose match level for most applications delivers matches with a higher degree of variation than Typical. It is generally used in systems where the risk of missing a match is high and manual review is available.

Managing Population Rule Sets

A Population rule-set is a file used by the SSA-NAME3 callable routine to modify its behavior for different countries, languages or data populations.

Population rule-sets may be one of three types:

- Standard Populations are provided with the product.
- A Custom Population may be built by an Informatica Corporation consultant for a customer with unusual or special needs.
- A Local Population is the result of local rules modifications done via the Population Override Manager or Edit RuleWizard.

It is possible for a system to have all three types of Population rule-sets. If so, there is an order of precedence in loading by SSA-NAME3. If a Local Population (file extension of `.YLP`) is present in the folder identified by the "System" Control, it is loaded; else if a Custom Population is present (file extension of `.YCP`), it is loaded; else the Standard Population is loaded (file extension of `.YSP`).

The task of developing name search and matching systems is a balancing act between:

- "Performance" and "Quality";
- "Under-matching" versus "Over-matching";
- "Missing the Right data" versus "Finding Wrong data".

Effect of File Size on Name Search Performance

Because there is an extreme skew in the distribution of words used in people's names, company names and addresses, some names will cover many candidate records, while other names will have only a few candidates.

If SMITH represented 1% of the population and Lebedinsky .001%:

Population Size	Number of SMITHs	Number of LEBEDINSKYs
1,000	10	1
100,000	1,000	1
1,000,000	10,000	10

If the family name alone was used in the search, a search for SMITH in a 100,000 record file would be slow; in million record file, prohibitive.

The more data that is given to the search, the better performance it can potentially achieve. However, even when more data is supplied in the search, coping with the skew of common and uncommon names requires careful key design. SSA-NAME3's key-building algorithms use a proprietary approach that gives the best balance between reliability and performance.

Impact of Risk on the Search Transaction

In many business systems the risk of missing a match must determine the scale of the search.

Compare the risk of missing:

- a bad credit record when lending $1,000,000 as opposed to $1,000;
- a criminal history record for a serial murderer as opposed to a petty thief;
- a border alert record for a terrorist as opposed to a visa overstayer;
- a medical history record as opposed to a prospect history record;
- a dangerous material advice record as opposed to a yellow pages entry.

In fraud, criminal and alert data, the important high risk record will often be harder to find because the identity alteration becomes more devious and complex.

In data which is collected over long periods, the important record will be hard to find because time may have altered the identification in the search data.

With complex or locally entered foreign data, an important record will be hard to find because of its tendency to contain severe error.

High-risk searches must be thorough. With today's data volumes, thorough searching must use intelligent keys and search strategies to manage the volume and quality of records returned.

Even with intelligent keys and search strategies, being thorough necessarily increases the volume of candidates returned. Because of this, reliable matching must also be used to assist the user by refining and ranking the list, and in some systems actually by matching the record, based on all available identity data.

Lower risk searches can afford to be less thorough, and can take advantage of assumptions about the stability of the data to provide quick access.

If you value your business, don't trust the same strategy or scale of name search for transactions of different risk values. You may need to automate the choice of strategy relative to the transaction's risk. Index the critical data separately and more thoroughly than the non-critical.

The Critical Exhaustive Search

Some examples of critical exhaustive searches are: the search of a fraud file in a high-risk financial transaction; top level security clearance for government; a border control search of a high-risk person alert list.

Typical characteristics of such searches are:

- the volume of records to be searched is relatively low compared to the volume of searches done;

- the bulk of the search data is more reliable and has different characteristics than the file data;
- the search needs to overcome the fact that in many cases, that are very critical to find, the identity will have been manipulated to try to defeat the search;
- the need to find a match if one exists is critical.

A critical exhaustive search must also be able to find identities, which have been deliberately manipulated to defeat the system while still retaining enough similarity to be explained as mistakes. It will need to succeed despite the country of origin of the identity. To do this, the critical exhaustive search must work harder and look deeper. It will also benefit from working more intelligently.

Quality and performance will improve the more that is known about patterns used to manipulate identity data. Quality will improve the more identification attributes are available for matching. Attributes with null values may need to be considered close to a match.

Because there will be more candidates on average returned from a search, maximizing the true matches and minimizing the false becomes harder. In many cases the computer system alone cannot make the choice "is this a match". The system's success is measured by how well it assists the user to make this choice.

Balancing Missed Matches with Finding Too Much

A designer of a strong name search will understand both the risk of a missed match and its cost to the business. When designing name search applications, recognize that each data population to be searched may have different risk attributes and costs of failure.

A missed match can be due to human error, because the name search failed to find the record, or because the match was "hidden" in the results set (due to the list being too large, or not in a useful sequence).

A name search, which fails to find a candidate match, either did not cater for some types of error and variation, or did not look exhaustively enough.

The more error and variation that is overcome, and the more exhaustive the search, the greater the potential for finding more true matches. The reality is, finding more real matches increases the amount of work and the cost. It also increases the risk that more false matches will be presented.

The goal of a good name search process is to maximize the true matches while minimizing the false matches. Even after the name search process has been tuned to provide this balance, there will always be the tendency to find more true matches at the expense of introducing more false matches.

In the final analysis, a well-informed decision should establish the cut-off point. If it is decided that no matches are to be missed within the power of the name search, then more human resources will be required to select the true matches from the false. If it is decided that human and machine resources take priority, then the name search can be tuned to deliver to that level.

One of the serious problems of finding too much for an operator to look at, is that the human operator themselves then make poor decisions.

Even good well-trained operators cease to be diligent if they are expected to be searching hour after hour, day after day.

With well designed automated matching it is possible to build systems that mimic the very best human operators looking at all the available data and making decisions that are significantly better than the average human operator can achieve.

Undermatching or Overmatching

Before a designer or user can decide what to show in a search or matching application, it is imperative to understand whether it is best to Undermatch or Overmatch.

It comes down to which case causes more or less problems for the business.

If it is simply a case of reducing the cost of mailing by avoiding duplicates then undermatching is good. Yet if it was important to avoid annoying the recipient, then overmatching would be good.

If it is a matter of not letting a known terrorist into a country or on to a plane, then overmatching is essential and, as in all security systems, a necessary consequence will be that some innocent people get inconvenienced by the process.

In a statistical process the consequences of undermatching can not be measured, but experiments can be designed to measure the amount of overmatching in the results.

In all designs it is necessary to know whether one would rather miss things, or rather find some things you did not want to find. Once one accepts that error and variation in the data is normal and unavoidable, then it is true that absolutely correct matching cannot be achieved, and it becomes necessary to decide if the "maybe true" answers should be seen or hidden from view. This is a fundamental business decision.

Discovering the Missed Matches

One of the greatest myths regarding name search systems is that they are successful simply because they find what was expected or is known to be on file.

To truly measure the success of a name search, one also needs to have an understanding of what matches have been missed. In many organizations, missed matches are only discovered once they adversely affect the business, operation or system. While this is often too late from a business viewpoint, such discoveries are useful input for improving the name search process.

Missed name matches can also be discovered from within the organization’s data by finding existing duplicates based on attributes other than name (for example, address and date of birth), or by exhaustively running a background matching process that uses less of the name data in its keys. To be useful for tuning the name search, this requires expert users to review the missed matches now found and help establish rules to avoid missing these matches in future.

Whatever the method of discovering matches that otherwise would not have been found, the goal should be to create and maintain a set of model answers, based on both real data and expert user input, as a benchmark for the reliability of the name search process.

It is not enough for a user to test only with the difficult cases not found by the old system. Tests should be carried out on more common names to ensure the search finds them as well and does not return too many.

The Match Level should be set to Loose during testing to assist the discovery of matches which otherwise would be missed.

A batch test of an online customer name search which uses as search criteria a file of new business transactions, or even the customer file itself, provides a valuable report for users to evaluate the reliability of the search.

Because the system resource usage of the name search transaction is higher than most business transactions, it is vital that the expected volume and concurrency of searches be factored into any capacity planning.

When it is critical to a business or system to absolutely avoid missing data, then it is critical to implement procedures and processes to discover real world cases and examples of what can be missed. Only then can systems be improved.

The Importance of Prototyping with Production Data

The performance, response time and "number of records returned" problems associated with name search relate, among other things, to the volume of data in the database and the skew of the distribution of names.

The reliability problems associated with name search relate, among other things, to the quality and make-up of the data being searched.

Name searches should be tested, or the results evaluated, by expert users who can feedback reliable information to the designer.

Normal test data cannot illustrate these volume & quality related problems. A name search system may pass design and acceptance testing but fail miserably in production for this reason. For example:

- a customer search which tests successfully on the 500 record employee file, is no test of how it will perform on the 5 million record customer file;
- a search which finds "TEST MICKY MOUSE, XXXXXXX XXXXXXXXXXX" or "THIS IS A VERY LONG NAME FOR TESTING", is no test of whether it will find "EYAL LEBEDINSKY", "ABUL MOHD AZIZ RAMAN" or "BILLY SAY LIM HO";
- a test search which uses the full name as search criteria is no good if the user ultimately only has a surname and an initial to search with.

Therefore, all but the initial functional testing of name search applications should be carried out on Production data and Production volumes. This also means that the data used to search with must also be appropriate for the production scenario.

If the Production data is loaded into a development or test environment, care should be taken to not deduce "production" response times from these environments, as the production system environment may be very different. It may be possible to monitor the average number of records returned from a search and extrapolate the average record access time to the production scenario, but this requires some careful investigation.

This chapter provides a background to why Informatica’s approach to identity search and matching supports strong multi-national systems.

Overview

Foreign name and address data could be data sourced from foreign countries, local data from a different geographic or cultural background, or simply data which has been previously unseen by your systems.

Such data is becoming more common in computer systems because of increasing multiculturalism, business globalization, electronic commerce, and because increasing amounts of identity data are being sold or shared in the market place.

A common problem in coping with foreign data is thinking that rigid standards as applied to known local data, will work for the foreign data. Requesting unformatted or loosely formatted name and address data is

the best way of obtaining reliability and completeness. Asking the data to be formatted according to strict rules is inviting assumptions and choices which can vary from person to person, country to country.

The different character sets used to capture and store the data also poses another problem. It does not make sense to stabilize and lose that information if the data is to be used to reach the source again. Yet, to match such data it is necessary to ignore variation in the character forms.

The best approach is to request foreign data unformatted and in its raw form, and to store it as such. Now, at least you have the best possible data available on the system.

Recognize that different business systems will want to use the data in different ways and leave it up to specialized software to overcome the problems associated with each business need. Don't try to overcome these problems before the data is stored.

William Stuart Harison	117- 2a Jacksen Rd., East Hartford, CT 06987
Kwok Ki Ho (William)	Block C, 4th Floor, Unit 7, 234 Wan Chai Road, Hong Kong
Mmd Farook Akbar	Block A 27 Jalan Tuanku Abdul Raman, Kuala Lumpur
Augusto Frederico R.Schneider	Aven. Maria C. de Aguiar, 235 cj. 32 São Paulo, SP - 02593.001
Keser Geylani Abdulkadir	Urt. Mahallesi Karaafat C.603/97 S.No.186 Syhan/Adana

Field Design for Multinational Systems

Whether the multinational system is to operate in one country and accept data from multiple countries, or whether the system is to be deployed in multiple countries, the way that names and addresses are captured and stored is crucial to the reliability of future matching on that data.

Names and addresses from different countries have different structures, follow different rules and differ in average quality (In Canada, it may be difficult to get a letter delivered without a post code; in Hong Kong, almost no one uses the post code).

A data model which assumes that the data for each country can be mapped into a detailed universal name and address format look nice on paper in the specifications, but will be costly to implement and generally unsuccessful in practice. The universal format for a name is a single field holding all name parts. Simply make sure the field is big enough.

The universal format for an address is multiple lines, as written on an envelope. Simply make sure the field is big enough.

If the success of matching name and address data in your multinational system is important, do not trust match keys or matching logic which rely on the data being parsed, cleaned or formatted.

Use simple large single fields for name data, and a box of multiple lines as is used on an envelope for addresses. A search and matching system, which succeeds with the full unformatted name and unparsed address lines, will be easier to implement, more flexible and ultimately give more reliable results.

Deployment of Multinational Systems

One goal of the designer of a system, which is to be deployed multinationally, is to reduce costs by minimizing customization, except where it is clearly necessary or benefits the user.

An example of where customization is often necessary is the language and font of the screens and reports.

An example of where customization is unnecessary is in the format of the fields used to capture, store and key name and address data. To simplify system expansion, these fields should be the same size and format for all countries.

In the internal design of the database keys, search keys and matching logic, then country level customization of names and address processing is essential. The processes, which build keys and perform matching, should be able to succeed with unformatted or partially formatted data. When the multinational system is implemented separately per country or regionally, then it will be beneficial to have key building, searching and matching algorithms that are tuned to each separate country or region's population of data.

If multiple character sets will be in use then character mapping algorithms, stabilization algorithms and tables for abbreviations, nicknames and other naming word rule bases will need to be externalized from the standard executable code. In some cases where multiple character sets and languages are in use in one country, translation rule bases will be also necessary.

These processes should be designed with a common interface such that implementing a new country requires only that new country-level modules and rule bases are plugged in.

Code Pages, Character Sets and other Encoding Issues

This subject is not for the faint hearted; nothing in this area is as simple as we would all like it to be. Massive advances in character display technology, standards, tools and protocols have occurred over time. However the globalization of systems and databases has increased the frequency with which these standards are being mixed together.

Some examples of real world problems will suffice to raise the awareness of important issues.

It is true that accents on characters make them sound different but in most countries the error rate and variation in the use of accented characters is very high.

It is true that today's keyboard and code pages support accented forms, however many users still key the countries old conventions where two adjacent characters are used instead, or simply leave the special characters out.

We have found that databases in some countries suffer from non-standard versions of the local codepage standard. Fixing this still means that old data has different characters.

Moving data between tools sometimes converts characters without your knowledge. Some tools convert from EBCDIC to BCD and then back losing information. Some processes convert ASCII to EBCDIC and back inconsistently.

One terminal in a network set up with the wrong Code Page can cause database maintenance errors.

In a site in Chile we saw a large database where some terminals were using a USA English code page, others with a European Spanish code page, and others with a Latin America code page. This led to users continuously correcting and re-correcting the accented characters in a name and still each user was unable to see a correct form of the data. The net result is a very corrupt customer file.

DBCS encoding for Japan and China suffers from having several standards. This leads to increased complexity when sharing or comparing data from different sources.

The fact that people sharing data around the world can not read the same character sets as each other leads to names and addresses necessarily being recorded twice, once in a local form and also in an international form. In some cases this leads to the wrong form being used in the two fields, or even unrelated names being used in each field.

There are mixed protocols for handling foreign words, such as in Israel where sometimes Hebrew phonetic forms for a foreign name are used rather than the original Roman characters, or in Japan sometimes using Romanji and at other times using Katakana for a foreign word.

Different code pages and data entry conventions involving foreign data increase the complexity and error in identity data and this in turn increases the complexity of the algorithms needed to overcome the error and variation.

Unicode Issues

Unicode provides a technically more competent way of implementing international systems, and simplifies the storage, transfer and display of multi-lingual data. However, Unicode in itself does little to address the problems of searching and matching identity data.

Unicode does not know

- that BILL is a form of WILLIAM
- that ЛЁША is a form of ALEKSEI
- that ناصر is the Arabic form of MOHAMMED
- that 有 is essentially just "noise" in a Chinese company name
- that Ann Jakson could be a form of Anne Jackson-Brown

While it may be natural to think that Unicode can help unify data across countries and languages, Unicode does not help find and match identity data even within one language, let alone between languages. Unicode can actually lead to an increase in variation of the identity data stored in a database if the data is allowed to be captured and stored in a variety of character sets.

Thus, the bilingual Greek/English data entry operator in England opening an account for a Greekborn British national (who has provided their Greek name on the application form), enters it in Greek because the system allows it. Worse, part or all of the name may even look like English (e.g. the name POZANA) and be stored as though it were an English name.

In the majority of systems, data entry should be restricted to the character set of the primary locale and converted to Unicode by the system. And it is essential that this locale information be kept and stored so that it is available for use by localized data matching algorithms. Conversion to and from Unicode will require that it be done consistently. Conversion of old data to Unicode will still inherit all the error and variation in the old character forms. Users will still enter new data with the old character conventions, and of course continue to make mistakes.

Transliteration Realities

In most computer systems the term transliteration is used in the context of converting from a non-Latin alphabet to the Latin alphabet, or Romanization. In the real world, however, transliteration can occur between any two alphabets.

For example, a United States organization with offices in the US and Japan decides that all of its Japanese customer data should be captured in Japanese and in Romanized form to maintain a single language view of the corporate databases. A bank in Saudi Arabia captures customer data in Arabic for local needs, and in English to satisfy needs for inter-bank wire transfers and compliance regulations.

Transliteration may be done formally (conforming to a documented standard – although there will often be a number of standards used by different groups or organizations); or informally (by ordinary people in their normal day to day work, adding personal interpretations to the mapping choices and frequently changing the rules and making mistakes.)

Different formal transliteration standards and informal transliteration may co-exist in the same system/database, and result in significant variation in the transliterated form. Transliteration also has an attribute of direction. Forward transliteration refers to transliteration from a name's original script to a target script (e.g. "Romanization" of an Arabic name from Arabic to English; "Arabicization" of an English name from English to Arabic.) Reverse transliteration refers to the transliteration of a name from its representation in a foreign script to its original script (e.g. "Romanization" of an Arabic name recorded in English back to Arabic; "Arabicization" of an English name recorded in Arabic back to English.)

In addition to data recorded in a local script, a system/database may contain data that has been the subject of any combinations of formal and informal, forward and reverse transliteration.

Transliteration and Data Matching

Transliteration can assist with data retrieval and data matching of identity data stored in foreign scripts, however, there are good and bad techniques.

Do not expect to achieve reliability and performance by transliterating multiple foreign scripts into a common character set and applying a localized matching algorithm to the result. There is too much conflict and compromise in the rules. Search and matching on data from different countries and languages should be handled by algorithms tuned for each country/language.

Even a technique that attempts to detect language source in transliterated data to choose strategies and algorithms has inherent problems. How does one safely choose the language source for the name "Mohammed Smith" or "CharlesWong"?

If original script and/or informally transliterated data is available, do not discard it; such data provides an additional source of information useful for search and matching.

The real value of transliteration and transliterated data is when it is used in conjunction with the source language. A solution that indexes, searches and matches on all available forms, uses this inherent redundancy to multiply the opportunity for success.

CHAPTER 6

Customer Identification Systems

This chapter includes the following topics:

- Overview, 50
- What Data to Use for Customer Look-up, 50
- Use of Full Name in the Customer Search, 51
- Responsibilities of the Customer Take-on Transaction, 51
- The Customer Take-on Transaction and Duplication, 52

Overview

This chapter provides a background to why Informatica's approach to identity search and matching supports strong customer identification systems.

What Data to Use for Customer Look-up

Customer look-up is expected to be both quick and accurate.

In some systems, frequently the search will use an id-number, which is ideal for quick and accurate retrieval. In other systems identity numbers are just not available or the business does want to make its customer feel like a number or an account.

When an id-number is not available, the search will need to be driven by some other piece of identifying data.

One of the challenges for the system designer is to decide which attribute or attributes are the best to use for this identity search.

Given a choice of name, birth date, telephone numbers or an address, how does one determine the best?

In an ideal world, one would try combinations of each attribute over a period of time and measure the system's results and the business benefits. In the real world, the decision often has to be made without empirical evidence.

Because dates suffer from the fact that a valid variation in any component creates a completely different but valid date, a search driven by a date is going to fail when one or more of the components are wrong.

Except where property addresses are the foundation of "customer" (for example, electricity and water companies), then addresses suffer from the fact that customers move regularly. A search driven by address is therefore going to fail when an address change has not been notified to the system.

Except when telephone numbers are the foundation of "customer" (e.g. telephone companies or utility and emergency services), then telephone numbers suffer from the fact that customers move and change them, use home, work, mobile and public numbers. A search driven by telephone numbers is therefore going to fail when the number has not previously been notified to the system. And like dates, errors in the number, or variations in format make indexing with such numbers quite unproductive.

Names avoid the pitfalls of dates, phone numbers and addresses. Unlike dates or telephone numbers, if a character in a name is different, then it still has a good chance of being identified because systems can compensate for variation and error in names. And unlike addresses and phone numbers, names tend to remain more stable over time.

Use of Full Name in the Customer Search

An important characteristic of a customer name search transaction is that the average customer actually wants to be identified and will provide a full name when requested.

In the majority of cases, that name will be given correctly and will match the data on file. If the search takes too long however, both the customer and the system resource manager will generally complain.

Assuming the tuning of the system and database is addressed, the response time of a name search is dependent upon the commonality of the name, the volume of data on file, the richness of the file name and the design of the keys.

If 1% of the customer data is about SMITH, a key built from family name alone in a database of 1,000,000 records could return 10,000 records for the SMITH search. A key built from family name + initial might reduce that volume to 500 records, but that is still too many. In addition, 1% of the customer searches will probably be about SMITH and so the problem gets worse.

If the customer take-on system only captured family name, or family name and initial, then these difficult to use results are the best one could expect.

Provided the customer take-on system captures the full name, and given that we are expecting the average customer to provide their full name for future access, the name search should be able to take advantage of this to search a much narrower set of records.

This requires the operator to understand that using the full name will improve the response time. Such a system must also allow the widening of the search in case the match could not be found at the initial full level of detail.

Responsibilities of the Customer Take-on Transaction

Modern customer systems generally have access to complete person details, to large amounts of data storage and to application environments which accept variable size fields.

There is no reason why these systems should ever discard or truncate data as they did in the past.

One major responsibility of these systems is therefore to capture as much data about the person as possible within the boundaries of privacy laws and good customer relations.

A customer take-on application also has the responsibility of verifying the integrity of the person's details. This involves all kinds of edit checking, and at least a check to see if the person is already known to the customer system.

It may also be in the organization's interest to check other data sources for such information as:

- has this person applied before and been rejected;
- does the customer have a poor credit history;
- has the person been linked with fraud.
- is the person on a identity watch/alert list.

The most reliable piece of information to use to perform such searches is the person's name. It is generally the most stable, and can sustain the most variation without losing its essential identity.

The type of name search performed on each data should be allowed to differ due to the varying risk associated with missing a match.

Using the other identifying person data, such as birth date and address for confirmation (but not in search keys) these searches should be able to return a short list of highly likely candidate matches.

The Customer Take-on Transaction and Duplication

When a Customer Take-on System cannot find a match, there is a good chance that the operator will NOT perform any further searches, and simply add the "new" customer as a new record.

Even when the system finds an existing record, if that record is not identical or not easily visible in the list, a new record will often be added.

The important consequence of missing a match, if there was one, is not the duplication in itself, not the extra disk space that duplicate records use, nor the increase in candidates returned in future searches, but that the new customer record will be "unaware" of the existing one. Therefore, in future transactions it will often be random as to which customer duplicate will be used or updated. Such unlinked duplication is a major risk to the integrity of the database. It is a risk to the business processes which expect to find only one record per customer, or at least to find all records relating to a customer together.

Duplication can be tolerated provided that the duplicate records are linked. Resolving duplication with merge/purge can cause data corruption and data loss.

Provided that duplicate records are linked, and systems are built to recognize the links, the decision to merge or purge duplicates becomes one of housekeeping rather than absolute necessity.

The real problem with duplication is when systems which use the data cannot resolve it, resulting in duplicate or unintended mail and even duplicate product being sent to customers, as well as a distorted view of the customer base.

CHAPTER 7

Identity Screening Systems

This chapter includes the following topics:

- Overview, 53
- Characteristics of a Screening Application, 53
- Identity Data in Screening Systems, 54
- How do you Prove that you have not Missed Any Records?, 55
- The False Hit Problem, 55

Overview

An important use for identity searching and matching in today's systems is the vital role it plays in identity screening.

Identity screening is used in a variety of systems including:

- Visa issue and Border control;
- Anti-Money Laundering (AML) Compliance and Know Your Customer (KYC) programs;
- Passenger screening;
- Pre-employment screening;
- Credit or Consumer screening;
- Marketing List suppression.

Characteristics of a Screening Application

Screening applications are about minimizing risk. The nature of the risk may be small or large. . . wasting the cost of mailing, damaging a relationship, doing bad business, doing catastrophic business, allowing illegal activity, putting someone's life in danger.

The screening data will often include both alert lists and "cleared" lists (identities that should be expediently cleared). Making a false match to a cleared list is potentially as dangerous as missing a match in an alert list.

Regardless of the level of risk, there are some common elements of identity search in most screening applications.

- It involves a search where a "no match" is normal;

- A search where a "no match" is a good thing;
- A search where if an "alert" match could possibly be considered as true the system must report it; and/or deny the transaction or record from further processing;
- It must be designed and tested so that nothing relevant is ever missed.
- It must minimize false matches to avoid unnecessary and possibly expensive investigation, or false clearance in the case of a false match to a cleared list.

There are four points in a system where important screening needs to be done:

- To stop the transaction or raise an alert - involves a real-time screen of a transaction against an alert list;
- To discover historical matches (e.g. when better matching algorithms have been implemented), a periodic batch screen of alert list data should be performed against the database;
- At the time of adding new alert list entries, a screen of the new alert list entry against the database is required.
- Batch suppression of records from participating in a business process (e.g. marketing list suppression).

In addition, if a serious alert is raised, a common investigative requirement will be to find any other data in the organization's database(s) that could possibly be related to the identity that triggered the alert. Government and Industry investigation units need to be able do this across data sourced from multiple organizations. In this situation, because it is possible that the identity data involved is fraudulent or may have been manipulated, a thorough identity search must be used.

Identity Data in Screening Systems

Alert list data used by screening applications has different characteristics than typical customer or marketing data:

- It is generally of poorer quality (while some entries may be captured from official documents, many others are based on intelligence or third-party reports);
- An entry generally has fewer identifying attributes (only name may be present);
- The data is less complete (if an address attribute is present, the data may be missing or of little value; date of birth if present may be only an estimate of age);
- It will usually contain multi-national data;
- The multi-national data will be biased to a handful of countries;
- The skew in country/culture origin of the alert identities will be different than the skew in the transactions to be screened.

A specific problem in the banking industry is the need to screen financial transactions for AML Compliance. Such data (e.g. wire transfers, S.W.I.F.T., ATM etc) is often complex, only partially formatted and the identification details may only comprise a subset of the information.

In addition, it is common that, while the volume of alert/cleared list data is low, the volume of the transactions to be screened is high and be constrained by response time or throughput expectations.

Identity screening systems that use negative alert data require different strategies than other systems.

How do you Prove that you have not Missed Any Records?

There is no certification body for search and matching tools.

Testing these tools requires that, not only are you confident that all the data found is relevant, you must also be confident that very seldom is relevant data missed.

In systems that screen transactions against files of alert lists or other such negative data, the very normal and common expectation is that extremely few matches will be found. Designing testing strategies to prove that nothing relevant was missed, when the normal result is to not find anything, requires a lot of experience and skill in the testing of this class of system.

Many products that find "duplicates" in files have been purchased because of the high volume of duplicates discovered, when the critical criteria may still be "how many duplicates remain undiscovered!"

Failing to find anything is clearly a desirable and acceptable result of the process, but only if it's true that nothing was missed.

By using software that has been used for long periods of time by organizations that have more critical needs and higher risk that your own, it is possible to be more confident that nothing is being missed.

The only way to be sure is by using software that allows controllable "overmatching". If the software can be controlled such that overmatched results can be made visible and matches can be intelligently ranked in relative order of relevance of match, it is now possible to audit the quality of work. Such "overmatching" is the only possible way to expose undiscovered "undermatching".

The False Hit Problem

While not missing an important match is critical, false hits are also a primary concern in many screening systems. They are potentially a drain on investigator time and damaging to client relationships.

While controllable overmatching is essential for testing and audit, an operational identity screening system must be capable of minimizing the false matches. In doing this it will need at times to cope with single word names, greater than normal noise, severe spelling errors, missing supporting data, foreign names, foreign character sets, fraudulent manipulation and more.

Such a system should be capable of finding matches such as:

```
SEARCH: TONY DONG-SUNG GYUNG
ALERT:KYEONG, ANTHONY

SEARCH:INVOICE NO V-8021~TOSONI/CHAN-SHEI HAN
ALERT:SHEIHAN

SEARCH:ABDULLAH ABDULAZIZ ABDULLAH AL MUSA
ALERT:ABD A/AZIZ A. ALMOUSA
```

While avoiding false matches such as:

```
SEARCH :HERR FRANCOIS RIENERT / IM BUEHL 181
ALERT:HERRI BAHALUNA

SEARCH:FIDUCIARY BANK INTERNATIONAL OF NY
ALERT:BANAKAAT-JORDAN INTERNATIONAL INC.
```

CHAPTER 8

Fraud and Intelligence Systems

This chapter includes the following topics:

- Overview, 56
- Identity data in Fraud and Intelligence Systems, 56
- What Search Strategy to Use, 57
- How Well do these Systems have to Match?, 57

Overview

This chapter provides a background to why Informatica's approach to identity search and matching supports strong fraud & intelligence systems.

In data used by Law Enforcement, Intelligence, Fraud and Security systems there is a growing need to support better reliability and availability, more data integration, increasingly diverse data sources and larger volumes of data.

Computer systems must make sure that the highly valuable data that is stored in these systems can in fact be found, despite its error and variation. Similarly the value of the high-end tools of criminal and fraud investigation that provide "link analysis", "data clustering", or "visualization" can be significantly improved if they make use of the very best search and matching algorithms.

Identity data in Fraud and Intelligence Systems

Many aspects of Fraud, Audit, Enforcement, Prevention and Investigation systems depend upon data about the names, addresses and other identification attributes of people and organizations.

All such identification data suffers from unavoidable variation and error. Often the data is out of date or incomplete. Often the entity committing the fraud or perpetrating the crime is in fact trying to defeat existing matching algorithms, by subjecting the identification data to deliberate, abnormal or extreme variation.

In systems which support intelligence and investigation work, databases of potentially relevant incidents and known perpetrators are maintained such that suspicious activity or new incidents can be linked or matched against them, or new patterns discovered.

Such databases require sophisticated indexing and search techniques that cope well with poor quality data, and provide timely and accurate results.

What Search Strategy to Use

Some solutions to the searching and matching requirements of such systems require skilled investigators who know when and how to vary a search or change the search data to cause the system to work more successfully. Boolean based and wild-card searches are an example of these.

A far better solution uses automated search strategies that satisfy all permutations and variations of the search. . . the real solution needs to be designed to find all the candidates regardless of the way the search data was entered, regardless of the quality of the data stored in the database, and regardless of the experience of the user.

Such search strategies must of course provide real-time searching of all name and identity data. On-line usage must satisfy the officer's or investigator's need for fast response without any loss of quality of search.

While diligent investigators can use sophisticated search tools well, it is not possible for the average user to spend day after day simply browsing historical data and do a good job selecting candidate matches; even the diligent user can get ineffectual at the job if it is a continuous activity.

To better automate the searching, matching and linking process, it is necessary that computer systems are designed to "mimic" the very best users when choosing amongst the possible matches. In the same way as human operators use names, addresses, dates, identity numbers and other data, the system must be able to use matching algorithms that effectively rank, score or eliminate the candidates.

How Well do these Systems have to Match?

When your CIS, CRM, Campaign System, or Call Center system fails to find a customer record that exist, you have an unhappy customer, or a lost opportunity to make profit. In this case, failing to find records that are present has a relatively small penalty.

Software that is good enough for "Duplicate Discovery" in marketing systems, or data warehousing systems will frequently leave undiscovered duplicates in the system the penalty is small enough for organizations to tolerate some failure.

When an insurance company fails to find out that it is doing business with a known perpetrator of fraud; when a Government welfare agency fails to discover that an address has been used for multiple fraudulent welfare applications; when a police officer fails to find out that the person in the car he/she just stopped is a serious threat, the penalties are likely to be large.

CHAPTER 9

Marketing Systems

This chapter provides a background to why Informatica's approach to identity search and matching supports strong marketing systems.

Different Uses of Names and Addresses in Marketing Systems

Marketing systems use the names and addresses of people and contacts in a variety of ways.

- In matching and deduplication applications. For example, to dedupe a prospect list against itself; to dedupe a new prospect list against customer data, existing prospect data, fraud data or 'do not mail/opt off' data.
- To reach prospects via direct mailings.
- To achieve cheap mailing rates by using Post Office preferred addressing.
- In scripts for telemarketing campaigns.
- To personalize letters, address labels and other marketing collateral to support a "friendly relationship"
- In campaign preparation. For example to group prospects by household or location.
- To match incoming phone calls against campaign files.
- To support statistical analysis of campaigns. For example, to reconcile new customers by location against prior geographically based marketing campaigns.

Conflicting Needs of Name and Address Data

Marketing systems have conflicting needs in the way that name and address data is captured, stored and used.

In many marketing systems, this conflict has not been recognized, leading to a bias in one area and a less than satisfactory solution in another.

For example, the address most useful for reaching the prospect or customer and fostering a good relationship is the one the prospect or customer provides; the address most useful for achieving a cheap mailing rate is the one the Post Office provides.

Data that is parsed and scrubbed as it is captured into a system to support postal enhancement and personalization should not be relied upon for the development of a match-code for matching and online enquiry.

Incorrect parsing destroys valuable data. Original data must be retained to support high-quality matching. Even if a match-code process that relies on cleaned and formatted data is used for the marketing system, it should never be used for systems where missing a match is critical (e.g. fraud, audit and intelligence systems).

INDEX

A

address data 45
Addresses 25
AML Compliance 54
Anti-Money Laundering 53

B

build keys 9

C

Code Page 47
Custom Population 11
Customer Identification 16
Customer look-up 50
customer name search 51

D

database size 19
dedupe 58
Duplicate Discovery 57
duplicate records 52
duplicates 55
duplication 52

E

EBCDIC 47
Edit Rule Wizard 11
Edit RuleWizard 41
exhaustive search 42

F

false hits 55
foreign data 20
foreign key 14
Foreign name 45

I

identification data 15, 56
Identity screening systems 54

K

Key Levels
  Extended 27
  Limited 27
  Standard 27
key-building algorithms 41
Keys
  Extended 9
  Limited 9
  Standard 9

M

Marketing Lists 16
Marketing systems 58
Match Decision 10
Match Level 44
Match levels 11
Match Levels
  Conservative 40
  Loose 40
  Typical 40
Match Purposes 10
matching 17
matching tools 55
Missed name 44
multinational system 46

N

name change transaction 18
name key 24
negative data 21

O

On-line usage 57
Organization Names 25
Overmatch 44
overmatching 55

P

partitioning search keys 21
Person Names 25
Population Override Manager 11
Population rule-set 41
Production data 45

R

Romanization 49

S

Screening applications 53
search keys 24
Search Levels
  Exhaustive 27
  Extreme 27
  Narrow 27
  Typical 27
search strategies
  Exhaustive 10
  Extreme 10
  Narrow 10
  Typical 10
search techniques 56
SSA-NAME3 CJK-SUPPORT 12
SSA-NAME3 Key 14
Standard Population 11
Standard Populations 24, 25

T

telephone book
  search 18
transliteration 49
Transliteration 49
truncate data 51

U

Undermatch 44
Unicode 48
Unicode Support 12